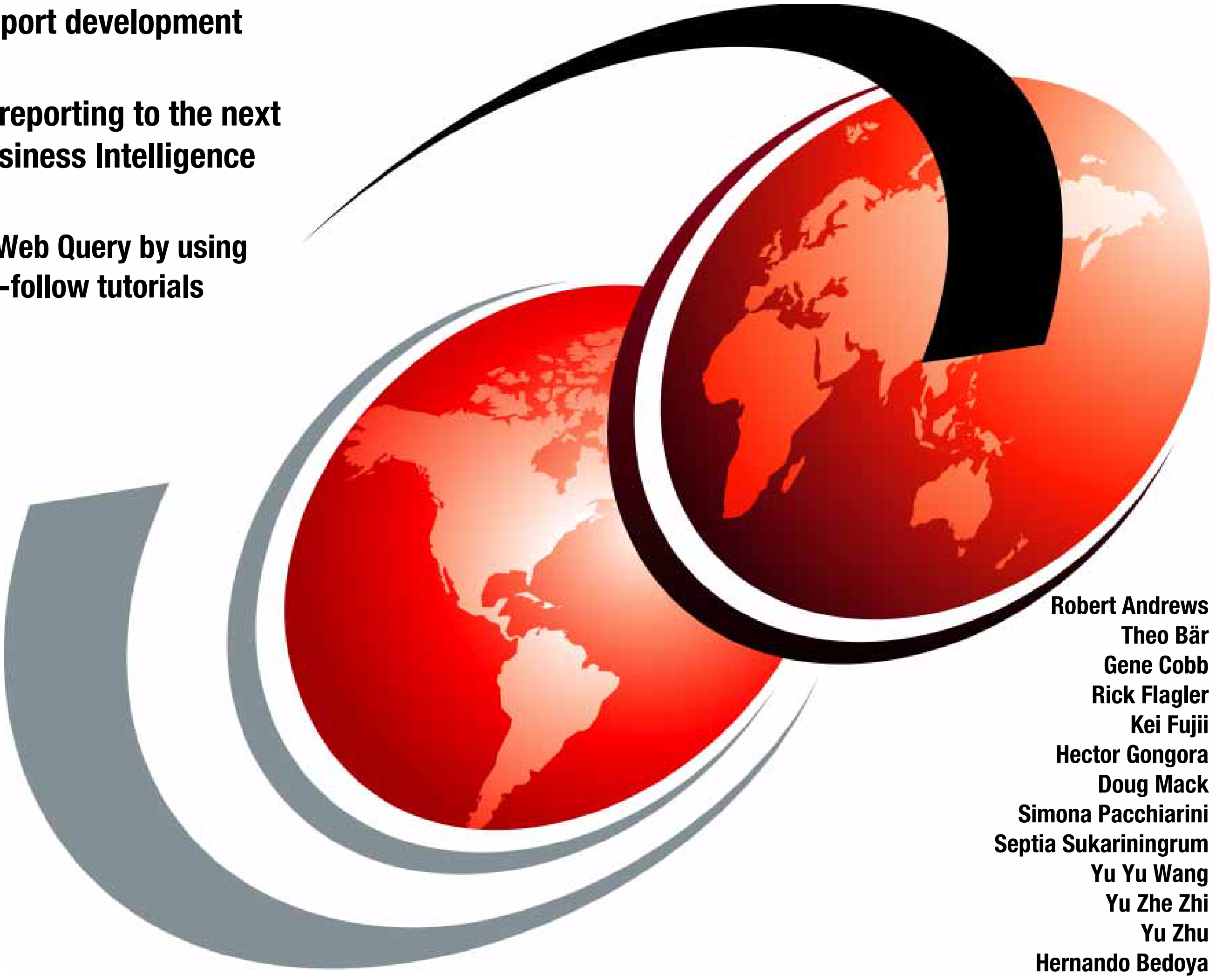
IBM

IBM DB2 Web Query for i Version 2.1 Implementation Guide

Follow the best practice guides to simplify report development

Take your reporting to the next level of Business Intelligence

Learn DB2 Web Query by using the easy-to-follow tutorials



Robert Andrews
Theo Bär
Gene Cobb
Rick Flagler
Kei Fujii
Hector Gongora
Doug Mack
Simona Pacchiarini
Septia Sukariningrum
Yu Yu Wang
Yu Zhe Zhi
Yu Zhu
Hernando Bedoya

Redbooks

ibm.com/redbooks

International Technical Support Organization

IBM DB2 Web Query for i Version 2.1: Implementation Guide

January 2014

SG24-8063-00

Note: Before using this information and the product it supports, read the information in “Notices” on page xiii.

First Edition (IBM DB2 Web Query for i Version 2.1: Implementation Guide)

This edition applies to IBM i 7.1 (product number 5770-SS1).

Contents

Notices

Disclaimer: The company, Century Electronics, used in this publication was chosen as a fictitious name. It is used for instructional purposes only. This name is part of the toolkit supplied by Information Builders and was selected to assist readers wanting to explore DB2 Web Query for IBM i powered by Information Builders.

Trademarks

Preface

Business Intelligence (BI) is a broad term relating to applications designed to analyze data for purposes of understanding and acting on the key metrics that drive profitability in an enterprise. Key to analyzing that data is providing fast, easy access to it while delivering it in formats or tools that best fit the needs of the end user.

At the core of any business intelligence solution are end user query and reporting tools that provide intuitive access to data supporting a spectrum of end users from executives to "power users," from spreadsheet aficionados to the external Internet consumer.

IBM® DB2® Web Query for i offers a set of modernized tools for a more robust, extensible and productive reporting solution than the popular Query for IBM System i® tool (also known as Query/400). IBM DB2 Web Query for i preserves investments in the reports developed with Query/400 by offering a choice of importing definitions into the new technology or continuing to run existing Query/400 reports as is. But it also offers significant productivity and performance enhancements by leveraging the latest in DB2 for i query optimization technology.

The DB2 Web Query for i product is a Web-based query and report writing product that offers enhanced capabilities over the IBM Query for iSeries® product (also commonly known as Query/400). IBM DB2 Web Query for i includes Query for iSeries technology to assist customers in their transition to DB2 Web Query. It offers a more modernized, Java-based solution for a more robust, extensible, and productive reporting solution.

DB2 Web Query provides the ability to query or build reports against data stored in DB2 for i (or Microsoft SQL Server) databases through browser-based user interface technologies:

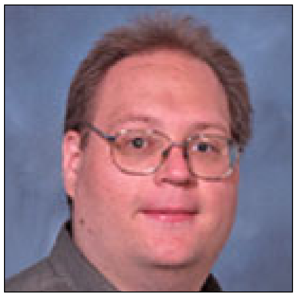
- Build new reports with ease through the web based, ribbon-like Info Assist tool that leverages a common look and feel that can extend the number of personnel that can generate their own reports.
- Simplify the management of reports by significantly reducing the number of report definitions required through the use of parameter driven reports.
- Deliver data to end users in many different formats, including directly into spreadsheets, or in boardroom-quality PDF format, or viewed from the browser in HTML.
- Leverage advanced reporting functions such as matrix reporting, ranking, color coding, drill-down and font customization to enhance the visualization of DB2 data.

DB2 Web Query offers features to import Query/400 definitions and enhance their look and functionality. It enables you to add OLAP-like slicing and dicing to the reports or to view reports in disconnected mode for users on the go.

This IBM Redbooks® publication provides a broad understanding of the new DB2 Web Query product. It entails a group of self-explanatory tutorials to help you get up to speed quickly. Overall, this book is designed for IT users. You can use Part 2, "Tutorials for DB2 Web Query" on page 161, as stand-alone tutorials for anyone who is developing their own queries.

Authors

This book was produced by a team of specialists from around the world working at the International Technical Support Organization (ITSO), Rochester Center.



Robert Andrews is an Advisory Software Engineer working for the STG Lab Services team in the DB2 for i Center of Excellence. He was the technical head for support for the DB2 Web Query product and currently provides custom consulting services helping clients make the most of DB2 Web Query and DB2 for i. Since joining IBM in 2001, Robert has co-authored many IBM Redbooks publications and his work has been published in trade publications with a focus on SQL, journaling, and communications.

He can be reached at robert.andrews@us.ibm.com.

Theo Bär is an IBM Business Partner and Consultant. He teaches classes to IBM i clients. He has been consulting and writing extensively on DB2 Web Query for i. He is the author of a German book on DB2 Web Query, and is also the author of multiple articles for the German “Midrange Magazin.” He worked for IBM Germany as a Systems Engineer from 1977 to 1982. Theo is a specialist in IBM i, SQL, RPG, EGL, and Profound Logic (PUI). He holds a diploma in economical engineering and mathematics from the universities of Karlsruhe and Heidelberg.

He can be contacted at Theo.baer@edv-baer.com.

Gene Cobb is a DB2 for i Technology Specialist in the IBM Systems Software DB2 for i Development team. He joined IBM in 1988 and spent over 10 years as an application consultant in the IBM STG Lab Services group, formerly the Client Technology Center (CTC) in Rochester, Minnesota. His current responsibilities include the promotion, education, enablement, and technical support of IBM DB2 Web Query, the strategic query and reporting solution for the IBM i platform. He speaks and writes regularly about DB2 for i and DB2 Web Query topics and is a co-author of the IBM Redbooks publication *Getting Started with DB2 Web Query for IBM i*.

He can be reached at cobbg@us.ibm.com.

Rick Flagler is an Information Technology Consultant and Adjunct Professor at Keene State College in Keene NH. He has been working on IBMi and predecessors for 30+ years. He held senior IT positions with The Timken Company and MPB Corporation, leading teams of developers and business analysts, installing ERP systems from major vendors such as SAP and Oracle/JD Edwards. His experience includes project management, database, and user interface development. He specializes in transforming data to knowledge and developing business metrics using DB2, SQL, IBMi, DB2 Web Query, Profound Logic (UI), ROBOT, SEQUEL, Microsoft Access, and Office.

Rick launched a grant-funded program to teach midrange development at Keene State College, a division of the University System of New Hampshire, where he has been an Adjunct Professor of Computer Science for 10 years. He lectures on database, user interface, and programming topics related to IBMi. He has held developer positions in the Insurance and Manufacturing industries as well as doing system integration and consulting. Rick holds a B.S. in Business Administration with emphasis on Computer Science from Nichols College.

He can be contacted at Rick.Flagler@ne.rr.com.

Kei Fujii is an Advisory IT Specialist for IBM i and working in the Systems Technical Sales division in IBM Japan. He has seven years of experience with IBM i systems in pre-sales and post sales support activities for clients as a technical expert. His areas of expertise are IBM DB2 Web Query for i, Business Intelligence, Hardware Configuration, and Capacity Planning. Kei has recently joined IBM Power Systems™ Technical Sales and is responsible for ATS and FTSS.

Hector Gongora is a System i and Database Administrator at Banco LAFISE in Nicaragua. His current responsibilities includes database performance and administration as well as System i high availability administration. Before joining Banco LAFISE, he worked for an IBM business partner as a System i Technical Support Engineer.

He can be reached at hgongora@bancolafise.com.ni.

Doug Mack is a DB2 for i and Business Intelligence Consultant in the IBM Power Systems Lab Services organization. Doug's thirty plus year career with IBM spans many roles, including product development, technical sales support, Business Intelligence Sales Specialist, and DB2 for i Product Marketing Manager.

Doug is a featured speaker at User Group conferences and meetings, IBM Technical Conferences, and Executive Briefings.

Simona Pacchiarini has been working for IBM Italy since 1989. Over the years she has participated in Redbooks publication projects on client connectivity, Windows cooperation, and database, first on the IBM AS/400® and now on IBM i. She is currently working in STG LAb Services in Italy, assisting customers on DB2 Web Query for i projects, DB2 for i performance, and IBM i system performance studies.

She can be contacted at simona_pacchiarini@it.ibm.com.

Septia Sukariningrum is an Advisory Technical Sales Specialist - Power Systems in IBM Indonesia. She has six years experience in the IT field, and started her career with IBM in April 2007. She has worked extensively in IBM i Capacity Planning, IBM i Performance Analysis, IBM i Availability, and IBM PowerVM®. She is now an IBM Certified Technical Sales Specialist for Power Systems with Power7 and IBM i. Septia holds a degree in Informatics Engineering from Bandung Institute of Technology.

Yu Yu Wang is a Software Engineer in IBM China Systems & Technology Lab. She joined IBM in 2007 and spent three years in IBM DB2 for i test work. Now she is responsible for Web Query development. She has experiences both on DB2 for IBM i and Web Query products.

She can be reached at wangyuyu@cn.ibm.com.

Yu Zhe Zhi is a Software Engineer in IBM China Systems & Technology Lab. His current responsibilities include the DB2 for IBM i test and DB2 Web Query for IBM i test. He has spent the past year working with DB2 Web Query for IBM i HotFix testing and the GA testing of this new 2.1.0 release.

He can be reached at zhiyuzhe@cn.ibm.com.

Yu Zhu is a Software Engineer working for IBM China Systems and Technology Lab. He joined IBM in 2010 and is mainly focus on DB2 Web Query development. He also has some experiences on DB2 for IBM i test and DB2 Web Query test. He is involved in the DB2 Web Query 2.1.0 development as well.

He can be reached at yuzhubj@cn.ibm.com.

Hernando Bedoya is a Senior IT Specialist at STG Lab Services and Training in Rochester, Minnesota. He writes extensively and teaches IBM classes worldwide in all areas of DB2 for i. Before joining STG Lab Services, he worked in the ITSO for nine years writing multiple IBM Redbooks publications. He also worked for IBM Colombia as an IBM AS/400 IT Specialist doing presales support for the Andean countries. He has 28 years of experience in the computing field and has taught database classes in Colombian universities. He holds a Master's degree in Computer Science from EAFIT, Colombia. His areas of expertise are database technology, performance, and data warehousing.

He can be contacted at hbedoya@us.ibm.com.

Thanks to the following people for their contributions to this project:

Linda Robinson
Ann Lund
International Technical Support Organization, Rochester Center

Jerry Evans
Charles Farrell
Cindy Mestad
Steven Ransom
Jim Bainbridge
Robert Bestgen
IBM STG Lab Services Rochester

Kathryn Steinbrink
IBM Development Rochester

Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at this website:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:

  ibm.com/redbooks

- Send your comments in an email to:

  redbooks@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

Stay connected to IBM Redbooks

- Find us on Facebook:

  http://www.facebook.com/IBMRedbooks

- Follow us on Twitter:

  http://twitter.com/ibmredbooks

- Look for us on LinkedIn:

  http://www.linkedin.com/groups?home=&gid=2130806

- Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

  https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm

- Stay current on recent Redbooks publications with RSS Feeds:

  http://www.redbooks.ibm.com/rss.html

Part 1

Background, installation, and setup

In this part of the book, we introduce DB2 Web Query for i and provide guidance on how to install it. We also introduce and provide background information about IBM DB2 Web Query for i, along with installation and setup instructions for the product.

This part includes the following chapters:

1

Product overview and architecture

Today's information overload requires companies to analyze data quickly and easily in formats that suit the needs of different types of users. Executives want quick access to track key performance indicators. Financial analysts want to analyze corporate performance to evaluate profitability or meet compliance guidelines. Business managers require up-to-the-minute sales and inventory information to ensure that customer demand and sales quotas can be met. The information must be delivered quickly and easily, with robust security in place, and distributed to a multitude of different client technologies, including browsers and spreadsheets, or integrated into line-of-business (LOB) applications.

In this chapter, we introduce IBM DB2 Web Query for i (DB2 Web Query). This comprehensive Java-based business intelligence tool provides graphical, Web-based query and report-writing functions.

We include the following key topics in this chapter:

- An overview of DB2 Web Query
- DB2 Web Query product structure
- DB2 Web Query architecture

1.1 Product background

Companies worldwide invest millions of dollars in operational applications to improve the way they conduct business. While these systems provide significant benefits, clients often struggle to meet the needs of various users with analysis of the data collected and stored in those systems.

Back in the 1980s, IBM provided a utility to view data stored in S/36, S/38, and AS/400 databases. Now commonly referred to as Query/400, this product continues to this day to be a heavily utilized utility for viewing data, creating printed reports, or extracting data for purposes of moving the data into a more modern visualization or analysis tool (such as spreadsheets). Many of the aforementioned operational applications still embed this decades-old technology into their applications for reporting purposes. It is truly amazing that after more than 25 years, this utility is still so heavily used today by I/T shops running IBM i on Power Systems to access data in DB2 for i.

Meanwhile, over those same decades, DB2 technology for processing data analysis requests has evolved FAR beyond what was available back then. And, unfortunately, Query/400 has not stayed up on a par with the numerous inventions within DB2 for i to improve performance and management of the query environment. So while still heavily used, it is not just the archaic front end reporting writing interface that has fallen behind, but also the back end processing by DB2 of queries submitted through the Query/400 tool.

On top of that, in today's highly competitive economic environment, the company that can analyze data in operational system databases in a more fluid and understandable way can make more intelligent and timely decisions. The net result is business intelligence, applying analytics to the data to provide insight for better business decisions.

Improved data analysis can provide many benefits, including these:

- Ability for end users to get the data when they need it, and in the forms in which they need it, through a self service model
- Allowing executives to monitor the state of the business through Key Performance Indicator dashboards
- Spot trends and exceptions in the data with real time analytical processing
- Automation of the creation and delivery of static reports in PDF, spreadsheet, or formats that allow data to be further analyzed in smart mobile devices

While most companies have the need for better analytics, not all companies have business analytics skills, large IT staffs, or a budget to pursue a large scale, enterprise analytics solution. This may apply to some of the companies who use IBM i today; the majority of which are in the category of “small and medium” businesses.

A few years ago, IBM introduced the IBM DB2 Web Query for i analytics solution to match the unique requirements of small and midsize companies. The solution is very affordable for companies with smaller analytics budgets. In addition, the solution is easy to configure, deploy, and use, making it easier to leverage existing skills, hardware, and fit within operational processes.

IBM DB2 WebQuery for i provides core analytics and BI platform capabilities including queries, reporting, OLAP support, and dashboards. Application integration interfaces are available to develop customized reporting applications or integrate the analytics into day to day operational applications. Mobile support enables end-users to view reports and information from their mobile devices.

IBM DB2 WebQuery for i has incorporated a significant number of integration points to leverage existing IBM i environments and to simplify the management of deployments. For example, operating system (CL) commands are provided to start, stop, and work with the reporting environment, create and refresh metadata, and invoke reports automatically through an existing IBM i based job scheduler. DB2 WebQuery for i integrates with IBM i security, work management, and software fix and license upgrade processes. It enables the BI software to become part of the normal IBM i management processes and policies. This capability can provide some interesting ways to allow reports to incorporate a fuzzy text search function for XML, Microsoft Word, or other document formats stored in an IBM Power System running IBM i.

Integration with DB2 for i provides an extensive array of capabilities to build more complex reports. The ability to define DB2 for i programs (called Stored Procedures), SQL Views, and DB2 User Defined Functions extends DB2 Web Query's capabilities. Combine the no-charge IBM OmniFind® product (5733-OMF) to create a search engine function for end users through DB2 Web Query. Add images to your reports representing documents such as a product catalog, and add links to allow an end user to drill down on the image to open that catalog.

Many IBM i companies store the majority of their operational data in DB2 for i, in some cases in multiple databases that span multiple servers or system partitions. A fundamental design point of DB2 Web Query is to leverage DB2 for i without having to move data around the enterprise. However, it is often a requirement to merge data from multiple databases into a single report, including data from non DB2 for i data sources.

Query/400 was very limited in that it only allowed you to build reports against a single DB2 for i database. DB2 Web Query expands that substantially by allowing you to access not only data in multiple DB2 for i databases, but also data in remote Microsoft SQLServer databases in the enterprise (all seamlessly to the end user). With the latest version of DB2 Web Query covered here, you can also integrate spreadsheets into the reporting environment and combine data from multiple sources into a single report. IBM continues to look at additional data sources to add to the DB2 Web Query family to meet the requirements of a single reporting tool for the IBM i centric company!

And that evolution of DB2 Web Query continues. In April 2012, IBM announced a restructuring of the product to simplify packaging and pricing, and to provide more flexibility for those wanting to support BI in the cloud. New "Express" and "Standard" editions simplify the individualized choices of features and products prior to this. New core-based pricing should be very attractive for those customers on larger systems that are leveraging IBM PowerVM virtualization capabilities. Additional enhancements include new role-based security, enhanced smart-phone device support including iPad and Android applications, and an enhanced user interface for DB2 Web Query authors and end users.

1.1.1 Taking advantage of DB2 for i analytics processing power

High-performance analytics processing is a key requirement for small and midsize companies, just as it is with large enterprises. Processing the data and gleaning new insight in an expedient manner maximizes the ability to make better business decisions sooner. A fundamental step in processing analytical workloads is the database's ability to process the query requests that the DB2 Web Query tools generate. IBM DB2 Web Query takes advantage of the advanced query indexing and optimization technologies, and self adapting self learning faculties within DB2 for i. These advanced features of DB2 for i can significantly improve performance, in many cases without any administrative intervention.

New with IBM i 7.1 is the DB2 for i Adaptive Query Processing, or AQP. AQP builds on top of the leading edge SQL Query Engine (SQE) to improve performance without requiring database administration.

Built into DB2, AQP enables the DB2 for i query optimizer to make real-time adjustments to its execution plan of a query, such as changing the join order or utilizing a new index, real-time while the query is running. This is a great example of an underlying performance enhancement that DB2 Web Query can leverage and Query/400 cannot.

Many analytical queries result in the summarization of transaction or other detail level data. A common technique to support these types of queries might be to programmatically build summary tables ahead of time and then have the reports access only the pre-aggregated summary tables. DB2 for i supports multiple aggregation techniques that provide a level of automation. Materialized Query Tables are summary tables that can be automatically recognized and available to a query being processed by DB2. New with DB2 for i 7.1 is the ability to include aggregate data directly within an Encoded Vector Index (a unique DB2 for i query acceleration indexing technology).

1.1.2 Optimizing the analytics with a dedicated, optimized server environment

For some companies, even small and medium businesses, it might be advantageous to isolate the analytics and reporting environment from the database production environment. These two run-time environments have very different workload characteristics, and the isolation of them can provide better system resource management and ultimately lead to better run-time performance. IBM Power Systems offers PowerVM virtualization capabilities to create multiple logical partitions on the same system. This feature allows the analytics environment to run in a separate partition on the same system where the database environment exists. A second option is to create the analytics environment on a second IBM Power System. That second system could be partitioned and used for other related or unrelated workloads. The second system could be leveraged for performing backups, recovering from disasters, or for development and test.

IBM recently introduced a new solution specifically to address this latter approach. The new IBM i for Business Intelligence (BI) solution provides several critical capabilities. First, it provides the data analytics environment on IBM i by including DB2 for i 7.1 and DB2 Web Query Standard Edition. Secondly, the solution includes an IBM POWER7® based system intended for housing data sourced from the database production system for use within the analytics environment. Thirdly, data transportation software is included to simplify the creation of an Operational Data Store for analytics. Lastly, installation services included in the package allow you to be up and running in a matter of days.

As opposed to a single-purpose appliance, the IBM i for BI solution can also be used for more than just the analytics. You can extend the solution to provide the ability to do backups, perform disaster recovery, or provide a development and test environment. The offering is available in several pre-configured sizes that can be expanded over time. The benefits of the IBM i for BI solution include reduced pricing, a second server for multiple purposes, quick implementation, and an optimized (tuned) environment for analytics, all of which meet the needs of small and midsize businesses.

1.2 Product structure

With the new V2.1 version of DB2 Web Query for i, IBM introduced a significantly more simplified product structure. Now a customer has a choice of one of two packages, referred to as “editions.” DB2 Web Query Editions are simple, to order bundles that provide a foundation for core DB2 Web Query functions for either getting started (Express) or rolling out the business intelligence application across your enterprise (Standard).

DB2 Web Query Express Edition provides an entry level software bundle that includes report authoring tools, Query/400 import function, and analytic reporting options for a small number of users. With the Express Edition, you can generate high quality PDF reports for external consumption, or provide OLAP reports for analysts interested in understanding trends or finding exceptions in the data to uncover what is driving results. Also, you can leverage the spreadsheet pivot table support and ability to integrate ad-hoc reporting directly into Microsoft Excel spreadsheets. For those with mobile devices who need to analyze data while on the road, the Active Technologies provide analytics for the users disconnected from the server.

DB2 Web Query Standard Edition combines all of the Express features plus more, providing a robust offering that can support up to thousands of users running reports. With Standard Edition, ample user licenses are included to support a typical roll out of developers and administrators, report authors, and a virtually unlimited number of run-time users and report consumers.

With Standard Edition, you can build more sophisticated dashboards for executives monitoring key performance indicators. Use Standard Edition to create an enterprise report distribution model with abilities to schedule report execution and distribution through e-mail or save the reports for later viewing. Optionally, leverage the data adapter for Microsoft SQLServer included in Standard Edition to be able to build reports including data from any of those databases in your network.

Business Intelligence is becoming part of many end user's daily activities. Integrating analytical function into line of business applications provides the end user with the experience that makes it simpler for them to turn data into information as part of their normal activities allowing for real time decision making. The facilities required to execute DB2 Web Query functions from a customized application are included in Standard Edition. Many clients will value the ability to create a URL interface to execute parameterized reports and surface that URL in an existing or new web-based application.

The new packaging also defines three types of user licences. An individual (named) “Licensed User” is required for anyone with administrative or report authoring or scheduling authorization. A “Workbench User” license is required for any Licensed User who also requires advanced metadata enhancement capabilities or the ability to build dashboards with the HTML layout tool. A new license type is the Run Time Group License. Similar in concept to the previous version's use of Run Time groups, this new license type is dedicated to supporting a group of runtime users, with each group supporting literally 100's of 1000's of users (but they can only RUN reports). How many Run Time Group licenses may be required probably depends less on how many users you want to have the ability to run reports, and more on how you want to organize your users into groups and associate them with a certain set of reports/folders they can work with.

Licensed Users and Workbench User Licenses can be added to both Express and Standard Edition. Run Time Group licenses can be added only to Standard Edition. At some point in the future, these editions could be extended or modified based on continual evolution of the product.

Two new product IDs are created to support the new structure. Express Edition is represented in IBM's ordering system as 5733-WQE, and Standard Edition is 5733-WQS. However, the Product ID that is actually installed on the system is 5733-WQX (and control of whether you are able to use Express or Standard Edition is done through License Management.

1.3 Metadata

One of the most important features of DB2 Web Query is the ability to define in one central place the particular meaning of data such that you do not have to define rules of data processing in each and every report. The concept of metadata is one that is considered a best practice by all the enterprise Business Intelligence tool providers. Why should a product targeted for small and medium businesses not adhere to this same best practice?

Through DB2 Web Query's metadata layer, authors do not have to worry about how to join files, or decompose dates through complex date conversion functions. Multi-dimensional relationships can be built into the metadata so OLAP reports can easily drill down based on how the business views its dimensions. Calculated fields can also be defined once into the metadata, rather than having to do this in every single report, assuring consistency and a “single version of the truth” for those definitions agreed to by the business.

The benefits of a metadata layer are many, including extending the community of report authors to more than just the I/T database experts. Tools provided to understand impact analysis of database changes, and automatic update of that metadata eliminate much of the inherent challenges that were associated with a heavy Query/400 reporting environment. Misinterpretations of data and inconsistent standards of the meaning of data can be catastrophic to businesses.

Metadata and synonyms: The terms *metadata* and *synonyms* are used interchangeably throughout this book. They both refer to the DB2 Web Query data abstraction layer.

1.4 Architecture

Although DB2 Web Query runs natively on the IBM i platform, it comprises multiple application tiers that service different components. The overall architecture contains each of the following components:

- HTTP clients
- Web tier
  - Application server
  - Web server
- Reporting Server
- Data adapters
- Relational Database Management System (RDBMS) data

  DB2 for i is the integrated database for the IBM i platform. Other source databases can be used with the installation of the appropriate add-on adapters that are discussed earlier in this chapter.

Figure 1-1 shows the overall architecture of the DB2 Web Query system.

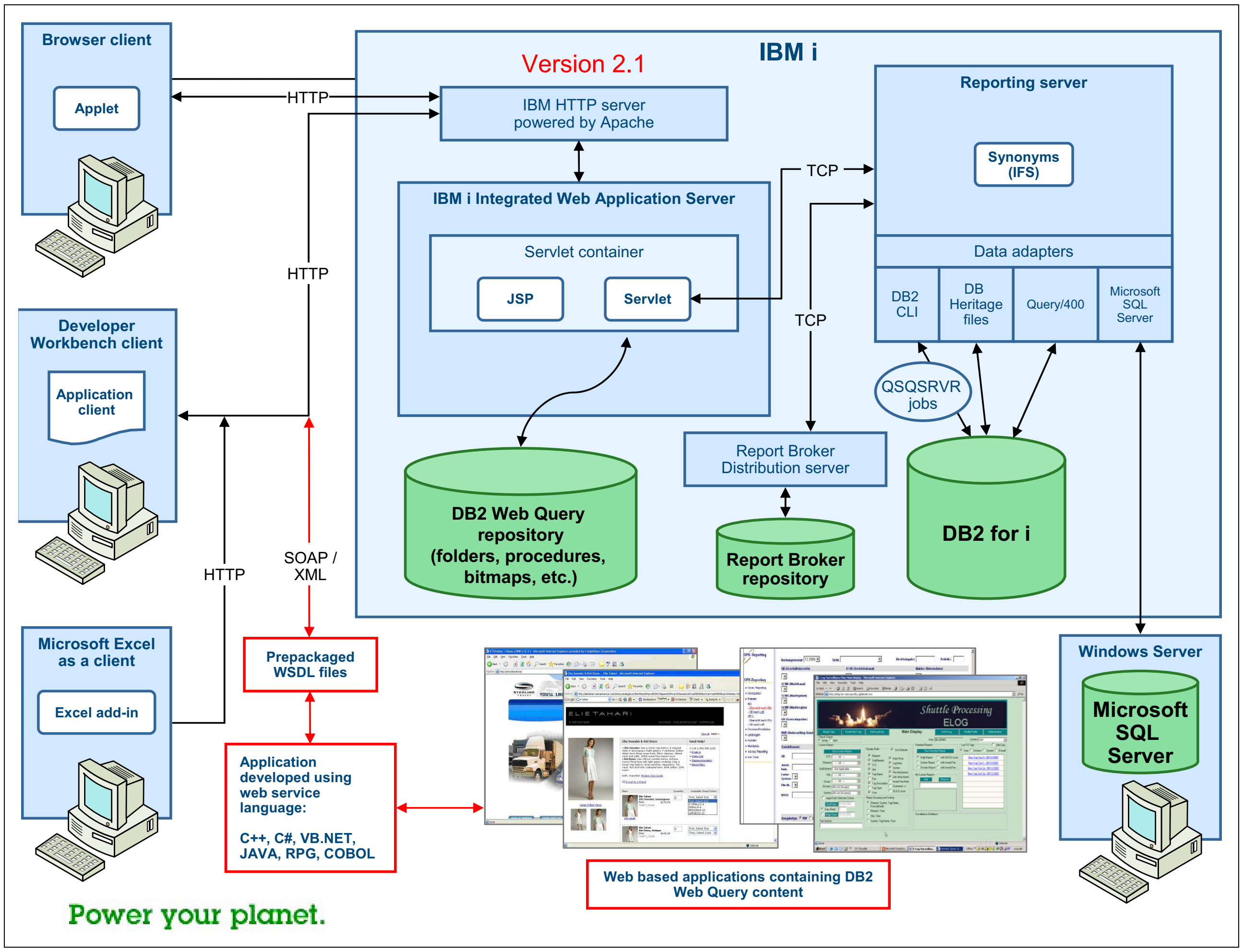

Browser client
Applet
HTTP
IBM i
Version 2.1
IBM HTTP server powered by Apache
IBM i Integrated Web Application Server
Servlet container
JSP
Servlet
TCP
Reporting server
Synonyms (IFS)
Data adapters
DB2 CLI
DB Heritage files
Query/400
Microsoft SQL Server
TCP
QSQSRVR jobs
Developer Workbench client
Application client
HTTP
DB2 Web Query repository (folders, procedures, bitmaps, etc.)
Report Broker Distribution server
Report Broker repository
DB2 for i
HTTP
SOAP / XML
Microsoft Excel as a client
Excel add-in
Prepackaged WSDL files
Application developed using web service language:
C++, C#, VB.NET, JAVA, RPG, COBOL
Web based applications containing DB2 Web Query content
Windows Server
Microsoft SQL Server
Power your planet.

Figure 1-1   DB2 Web Query base architecture

To better understand the architecture, it is helpful to understand the end-to-end process and how a report request flows through the architecture. The following actions describe the life cycle of a typical DB2 Web Query request:

1. A user requests a report for execution from a Web browser.
2. The Web server receives the request, processes the parameters, and routes it to the Reporting Server via the DB2 Web Query servlet.
3. The Reporting Server processes the request and passes it to the appropriate data adapter.
4. The data adapter generates the appropriate database request and submits the request to the DB2 for i database engine.
5. The Reporting Server receives the result set from the database, formats the report, and returns the formatted report to the Web server via the DB2 Web Query servlet.
6. The Web server delivers the report to the user's Web browser for display.

1.4.1 Web browser clients

Both DB2 Web Query developers and users use a Web browser as the interface to the product. The browser clients communicate via HTTP to the application server. At the time that this version of the book was published, the following Web browsers were generally supported:

- Internet Explorer v9 (32-bit)
- Internet Explorer v8
- Firefox v17, v16, v15, v14
- Safari v5.1.5
- iOS v5
- Chrome v21
- Opera v12.02

Specific browser and version support is dynamic and varies across the assorted DB2 Web Query components. Because this information is constantly changing and must be kept updated, specific support matrices are not provided in this publication. This information is provided in the DB2 Web Query Summary of New Features (SNF) document that is updated for every release and group PTF. This document is stored on the DB2 Web Query IBM developerWorks® WIKI. The SNF documents can be directly accessed by opening the following URL:

`http://ibm.co/db2wqnewfeatures`

Locate and download the SNF document titled **R210 New Features**. Browser support matrices are located in the Web Browser Support section of this document.

1.4.2 Web server

Web servers generally serve as the static content repository and handle HTML, GIF, CGI, and other traditional Web content and processing. The IBM HTTP Server is used to fulfill this role for the DB2 Web Query product.

The Web server content for DB2 Web Query is grouped into contexts (aliases). This information is stored in the both the IBM i integrated file system (in the `/qibm/proddata/qwebqry/base80/ibi_html` directory) and the DB2 Web Query database repository. Within these two data sources are the following items:

- Common templates and forms
- Common images, scripts, and style sheets
- Published forms and launch pages

1.4.3 Application server

An *application server* is a component-based product that resides in the middle-tier of a server-centric architecture. It handles requests from Web clients that require Java and non-traditional processing and provides middleware services such as security and state maintenance. DB2 Web Query runs on the IBM i™ Integrated Web Application Server and uses the servlet container component to process these client requests. Servlet containers are also referred to as *servlet engines*.

The DB2 Web Query application server has the following attributes:

- Is a J2EE-compliant Web application
- Runs on a *servlet container*, a Java-based container that serves servlets, JavaServer Pages (JSPs), and connectors

- Does not require a full J2EE application server
- Supports servlets
- Standardizes deployment schemes using archives or packages
- In some cases, groups multiple applications into Web Application Bundle (WAB) files

  Applications are grouped into a WAB file that is served under the context root of `/ibi_apps` and contains the following components:

  - DB2 Web Query servlet:
    - DB2 Web Query API
    - Managed Reporting API

      Communicates with the DB2 Web Query Repository
  - DB2 Web Query home page controller and JSPs
  - DB2 Web Query administration console
  - DB2 Web Query and Managed Reporting drivers
  - Communication Drivers to DB2 Web Query server
  - Custom DB2 Web Query exit (plug-in)
  - Custom servlet filter (plug-in)

1.4.4 Reporting server

The DB2 Web Query Reporting Server is a C-based application that resides on the IBM i platform and is responsible for managing data access, processing the business logic, and generating the fully styled output. It comprises the following components:

- Reporting Engine of DB2 Web Query product
- Data adapters repository
- Metadata or synonym repository

1.4.5 Data adapters

An *adapter* is a program that enables DB2 Web Query to access a data source. Data adapters are responsible for generating the appropriate request to submit to the database engine. DB2 Web Query comes with three different adapters. The adapter that you choose to create your metadata or synonym is transparent to the users who will create reports. The format of the request depends on the adapter that is used:

- The *DB2 CLI adapter* generates appropriate SQL statements to submit to the DB2 for i database engine.

  We recommend this adapter for use by DB2 Web Query. Select this adapter if you want to write a report on a single member physical file. The report can include an SQL table or a data description specifications (DDS)-created physical file. This adapter is also used for object types of alias, stored procedures, and materialized query tables (MQTs). The DB2 CLI adapter generates SQL that takes full advantage of the latest DB2 enhancements that are found in the SQL Query Engine (SQE).
- The *DB2 Heritage File adapter* generates the appropriate OPNQRYF command to handle reports that are based on data in a file that has multiple members or multiple record formats.

  This command does not take advantage of the new enhancements to the DB2 optimizer and uses the Classic Query Engine (CQE) under the covers.

- The *Query/400 adapter* retrieves the associated Query/400 query.

  Choose this adapter when you want to create metadata on an already existing Query/400 query. The QRYDFN object must exist on the IBM i platform. This adapter sends the RUNQRY command to the server, which also uses the older CQE.

Table 1-1 summarizes the adapters and how they correspond to the data types.

Table 1-1 Data adapters

Adapter	Data type	Command sent to IBM i® machine
DB2 CLI	Single member physical file; alias, stored procedure, or MQT	CLI API
DB2 Heritage File	Multiple member file, multi-record format files	OPNQRYF
Query/400	QRYDFN object	RUNQRY

These three adapters allow you to query DB2 for i data sources and are included in the DB2 Web Query Express Edition. An additional data adapter is bundled in the DB2 Web Query Standard Edition: the SQL Server adapter allows you to connect to Microsoft SQL 2000, 2005, and 2008 servers. Metadata is created the same way as in DB2. Once created, you can create reports solely against the SQL Server data or even join tables between an SQL Server and DB2 on i. Just learn one set of tools and use them for all of your reporting needs.

In addition, a JD Edwards application adapter can be purchased as a separate option from the Standard Edition package. This adapter will automate some of the transformations necessary to display JD Edwards information in a meaningful way.

1.5 Request for changes on Web Query

The IBM DB2 Web Query team is always looking for ways to improve the product. If you have a suggestion for a new feature or would like a change to an existing feature, let us know. You can voice your opinion and ideas to us via the Web. You can use the standard Request for Design change form. This form can be used to requests changes to all IBM products around IBM i. For Web Query, be sure to select the DB2 Web Query option from the problem area drop-down menu. The following address is used to get to the Web form:

https://www-912.ibm.com/r_dir/ReqDesChange.nsf/Request_for_Design_Change?OpenForm

2

Installation and server operations

In this chapter, we explain the installation of DB2 Web Query, 5733-WQX, using the RSTLICPGM command. We outline the PC and IBM i prerequisites for installing DB2 Web Query. We also describe which IBM i system jobs will run and their functions in DB2 Web Query. In addition, we discuss other system objects that are created upon installation of DB2 Web Query.

Note: For the most current and complete information about the installation instructions, see the *IBM DB2 Web Query for i Install Instructions* at this website:

`http://ibm.co/db2wqinstallation`

2.1 Installation and setup

The licensed program product number for DB2 Web Query for i is 5733-WQX, which replaces 5733-QU2. The 5733-QU2 and 5733-WQX products can coexist and run concurrently, as long as 5733-WQX is in the trial period. After 5733-WQX is licensed, you can run either product, but not both at once.

2.1.1 Installing DB2 Web Query

Important: Before you start the installation, you must obtain the most current information about the prerequisites and required program temporary fixes (PTFs). You can find this information in an Info APAR, which is available at this website:

`http://ibm.co/db2wqinstallation`

To install DB2 Web Query, you must follow the up-to-date instructions on this website. You will find links to the Installation Instructions, Upgrade Instructions, and Info APARs.

While the exact steps are documented on the web, the basic flow of the process is as follows:

1. Install prerequisite programs, if missing.
2. Install prerequisite PTFs and PTF Groups, if missing.
3. Install the DB2 Web Query product and any options purchased.
4. Install the PTF Group for DB2 Web Query.
5. Configure a language if not running in English.

It is important that you follow all the steps as documented in the Installation Instructions and Info APARs. As noted, a new Group PTF for Web Query will be released about every three months. Refer to the Info APARs about updated Group PTFs and try to stay as current as possible. If you run into a problem and call support, you will be asked whether you are at the most current Group PTF for Web Query.

IBM i system objects

Upon installing 5733-WQX, the following integrated file system directories are created:

- /QIBM/PRODDATA/QWEBQRY
- /QIBM/USERDATA/QWEBQRY

Table 2-1 lists the system objects that are created.

Table 2-1 System objects

Object name	Object type
QWEBQRYX	*LIB
QWEBQRY	*LIB
QWQREPOS	*LIB
QWQCENT	*LIB
QSYS/QWQADMIN	*USRPRF
QSYS/QWQ0000000	*AUTL
QWEBQRY/QWEBQRYJOB	*JOBD
QUSRSYS/QWQADMIN	*MSGQ

This is the DB2 Web Query-related user profile:

- QWQADMIN: This profile has functionality on the server side and has authority to start and stop the Reporting Server. The profile is automatically created.

2.1.2 Authorizing and verifying users

The users must be licensed to use DB2 Web Query. For each user that will use the product, you must allocate a license. There are multiple types of licenses available including Web Query Developer Users, Web Query Developer Workbench Users and Web Query Runtime Groups. However, every user must be pre-authorized. In order to add or remove licenses from user profiles, use the Security Center. For details on the various licenses and the use of the Security Center see Chapter 6, “Security Center: Setting up users” on page 171.

You can view the currently licensed users by running the following command:

```
WRKLICINF PRDID(5733WQX)
```

Then take option 8 next to the feature 5104, 5105, and 5106.

2.1.3 License keys

After you install DB2 Web Query, you are allowed a grace period in which to try the product. During this grace period, you have access to all the features that are provided in the base product as well as all of the IBM add-on components. However, after the grace period expires, license keys are required to continue using both the base product and the add-on components.

To obtain your DB2 Web Query license keys, you must work through your normal product ordering channels (for example, a System i business partner). The system serial number is required to generate your specific license keys.

Each license key is used to activate either the base product feature (5050), the Express Edition feature, the Standard Edition feature, or one of the add-on features. To enter your license keys and activate the specific features, use the ADDLICKEY command.

Note: You do not have to wait for the grace period to expire before you enter your license keys. You can do this at any time.

2.1.4 Dynamic Language Switching

If you are running a system that may need to be viewed in multiple languages, the Dynamic Language Switching option is for you. This is included in the base product. It allows you to set a list of languages that a user can select from at login time. This feature will, by default, update the interface of the program to match the language selected.

To use this feature, follow the instructions in the installation document on the DB2 Web Query wiki at this website:

http://ibm.co/db2wqinstallation

2.1.5 Sample database

Throughout this book, we use a sample database called the Century database. This database will be available after installing DB2 Web Query on IBM i. The library is named QWQCENT. This allows you to follow along figure by figure to see exactly what you will get in the book. You can use your own data if you like, but we make this sample database available to you.

2.2 Requirements

In this section, we cover the PC and IBM i minimum requirements to install and run DB2 Web Query.

2.2.1 PC requirements

No client is required to be installed on your PC to enable DB2 Web Query. You only need a browser. For more information on the supported browsers, see "Web browser clients" on page 10.

In regard to PC memory requirements, in general, queries usually run best on PCs that have 1 GB of RAM or more.

2.2.2 IBM i requirements

Your IBM i environment must have 6.1 of IBM i or later.

2.2.3 Developer Workbench requirements

Although the Developer Workbench client is not required for DB2 Web Query report development and execution, you might still find that it provides additional development features that make it worthwhile. If you choose to install it on your PC, use the hardware and software requirements listed on the wiki at this website:

`https://www.ibm.com/developerworks/mydeveloperworks/wikis/home?lang=en#/wiki/W516d8b60d32c_4fc5_a811_5f3d840bf524/page/DB2%20Web%20Query%20for%20i%20Wiki`

Note: You must be an administrator to the Windows machine to run the Developer Workbench installation.

Provided third-party component

The third-party component Java 2 SDK is provided for use with Developer Workbench. A Java SDK is required for WebFOCUS features such as servlet connectivity, graph generation, and online analytical processing (OLAP). If it is not present on your machine, you have the option to install it with Developer Workbench.

If Java SDK 1.4.1 or later is not installed on your machine, you *must* allow Developer Workbench to install the SDK.

2.3 Web Query administrative commands

The base component of the licensed program provides multiple IBM i commands to help with DB2 Web Query administrative tasks. By default, some of the commands to work with Web Query are shipped as *PUBLIC *EXCLUDE. However, any user that is a DB2 Web Query administrator (a member of the QWQADMIN group profile) or is authorized to use the commands via *ALLOBJ special authority or explicit authority can use these commands. When DB2 Web Query is started, a prerequisite check is performed. If required products or PTFs are missing, DB2 Web Query will not start. The job log will provide complete details on what was missing to allow you to correct this issue. Here we describe some commands:

- STRWEBQRY

  Use the STRWEBQRY command to start all the jobs that are required to run Web Query.

- ENDWEBQRY

  Use the ENDWEBQRY command to end the Reporting Server.

- WRKWEBQRY

  Use the WRKWEBQRY command to start, stop, and display the status of DB2 Web Query. Each Web Query port and its status is displayed, as well as installed product and prerequisite information.

- CRTWQSYN

  Use the CRTWQSYN command to create and refresh DB2 Web Query synonyms. You can also use the metadata console from the browser as well as Developer Workbench to create synonyms.

- WRKWQRTE

  Use the WRKWQRTE command to create dynamic Runtime Environments and assign users to them. The feature allows users to dynamically change their library list in a quick and easy manner. For more information on dynamic Runtime Environments, see the following document on the DB2 Web Query Wiki:

  https://www.ibm.com/developerworks/mydeveloperworks/wikis/home?lang=en#/wiki/W516d8b60d32c_4fc5_a811_5f3d840bf524/page/Dynamic%20Runtime%20Environments

2.4 DB2 Web Query server jobs

Typically, the interface that DB2 Web Query uses to access the DB2 for i database is the SQL call-level interface (CLI). CLI is a callable SQL programming interface that is available in DB2 on the IBM i operating system. CLI consists of application programming interfaces (APIs) that are used to connect to the server and start dynamic SQL statements. The CLI is a subset of Open Database Connectivity (ODBC). The system job that processes CLI is QSQSRVR.

For more information about the CLI, see *System i Database DB2 UDB SQL call level interface (ODBC)* at this website:

http://publib.boulder.ibm.com/infocenter/iseries/v6r1m0/topic/cli/rzadp.pdf

When DB2 Web Query is active, the following jobs run on the IBM i server in the QWEBQRY21 subsystem:

- EDAPTH: One job that helps with the workspace process and runs under the user profile QWQADMIN.
- EDAPLOG: One job that contains startup information and runs under the user profile QWQADMIN.
- EDAPGWY: Three listener jobs, one each for HTTP, TCP, and Java. These jobs receive incoming requests and hand off work to the TSCOM3 jobs. All run under the user profile QWQADMIN.
- TSCOM3: By default, four of these jobs are running. These jobs accept the request from the EDAPGWY job and translate the DB2 Web Query request into SQL. One TSCOM3 job does not correspond to one user. One TSCOM3 job can service several users. For this reason, the design of these jobs contributes to the scalability of the product. They are referred to as *agents*. and all run under the current user's user profile.
- JSCOM3C: One job that services Java processes and runs under the user profile QWQADMIN.
- WQLWI80: This job is for integrated application server jobs. Four of these jobs should be running whenever DB2 Web Query is active (three run under the user profile QTMHHTTP, the other under profile QWQADMIN).
- QSQSRVR: This IBM i native prestart job handles SQL requests that are made over CLI. This job does the actual database work of optimization, execution, and returning the result. By default these jobs run under the user profile QUSER, but when DB2 Web Query report (based on a DB2 CLI synonym) is run, a CLI connect event occurs with one of these jobs. At this point, the user profile running the job is swapped to the user profile that initiated the DB2 Web Query report request.

In addition, the following jobs are started to support the DB2 Web Query report scheduling and distribution feature (known as Report Broker):

- STRBROKER: One job that runs under QWQADMIN and invokes QSHELL to run a script to start the Distribution Server.
- QP0ZSPWP: Two jobs that run under user profile QWQADMIN. One job is the JVM and the other is the console window that the Distribution Server starts up.

Another way to look at the jobs is via their ports. When running, DB2 Web Query will be active on ports 12331 to 12339. Port 12339 is used for DB2 Web Query Report Broker. Therefore, if you do not have that product installed, this port will not be active. At the time of publication, port 12337 is not used. The WRKWEBQRY command can be used to display the status of these ports. Alternatively, you can look at active ports by using the NETSTAT command and taking option 3. Once the display is up, use F14 to display port numbers. As you scroll down, you will see that the local port column is ordered descending. Scroll down until you reach a local port of 12331 to see these connections.

2.5 Running QU2 and WQX concurrently

Customers that migrate from Web Query 1.1.x (product id 5733QU2 or QU2) to version 2.1 (5733WQX - WQX) will be able to run both versions concurrently under the following conditions:

- A permanent WQX license key is NOT applied.
- The trial period for WQX has not ended. The trial period initiates the first time WQX is started using STRWEBQRY command. It ends 70 days after it is initiated.

When the WQX trial period is over, permanent license keys must be entered to continue usage of the product. At this point, the customer will not be able to start both QU2 and WQX at the same time. They can, however, start one or the other, just not both at the same time. Here are the scenarios when trying to start both:

- If both QU2 and WQX are down, QU2 is started, and an attempt is made to start WQX, then the request to start WQX fails.
- If both QU2 and WQX are down, WQX is started, and an attempt is made to start QU2, then the request to startQU2 fails.

When the trial period has expired and permanent 5733WQX license keys have been applied, administrators will notice that version 2.1 allocates port 11331 (instead of port 12334 as described above). This port allocation is what prevents both versions from running concurrently.

Table 2-2 summarizes the ports used by the two products.

Table 2-2 Usage of the ports by the products

5733QU2 (version 1.1.x)	5733WQX (version 2.1.x
11331	12331
11332	12332
11333	12333
11334	12334 (during trial period) 11331 (after trial period)
11335	12335
11336	12336
11338	12338
11339	12339

3

Defining metadata

This chapter explains what metadata is and why you should consider it as an advantageous feature, something that can reduce the complexity of your query and reporting environment and make life easier for your report developers.

Before you can create a single report or graph in DB2 Web Query, you must create metadata (also known as synonyms) over the data source. The metadata requirement of the DB2 Web Query product may be foreign to you. Many IBM i customers are simply not accustomed to working with a product that requires metadata and consequently view it as an unnecessary burden—an extra layer of complexity that they have to create and maintain. Typical questions include these:

- What exactly is metadata?
- Why do I need it?
- Can I not just query my files directly?
- What happens if the structure of my underlying files changes?

In this chapter, we cover the details of the metadata and how it relates to DB2 Web Query for i.

3.1 What metadata is

Metadata is simply data about data. Whenever you issue the DSPFD or DSPFFD commands, what is generated and displayed on your screen is in fact metadata; information such as record lengths, record formats, field names, data types, field attributes, and field lengths. DB2 for i also maintains system catalogs, which are files that store information about each of the objects in your database. They are effectively a materialized metadata repository that is kept up to date and can be queried to collect a wide variety of information about your database.

In much the same way, DB2 Web Query metadata is a materialized repository that contains information about your database files. In other words, DB2 Web Query can potentially read all of your database, but you have to give it information on where the data is located and how it is structured. Before you can create a report or graph in DB2 Web Query you must first create metadata (also referred to as synonyms) over the data sources. You can create a synonym over the following database objects:

- Tables/physical files
- SQL views
- DDS logical files
- Stored procedures
- Materialized query tables

In general, whenever you create a synonym over an object, database or flat file, two stream files are created in the Integrated File System (IFS) in a directory that has the same name as your folder in DB2 Web Query `/qibm/UserData/qwebqry/apps/<myfolder>`. The two files that are created are called a master file and an access file:

- Access file: This file identifies the object name and type on which the metadata is created. The access file has an extension of .acx. The access file consists of multiple keyword-value pairs that are separated by commas. Attributes included are the name of the synonym segment (`T1_ORDERS`), the underlying table that is referenced (ORDERS in library QWQCENT), the system to connect to in order to find the data source object (*LOCAL), and the number of key fields (0).

  If the table has referential integrity relationships with other tables, the relationships are listed in the .acx file. If the synonym refers to more than a table (cluster or join synonym) all the tables are referenced with the information described above. If the object is a QRYDFN, it says QIQRY. If the object type is a table, it says TABLENAME. Figure 3-1 shows an access file for table QWQCENT/ORDERS. If the table is linked with Referential Integrity relationship with other tables, the existing relationships are explicitly stated in the .acx file (Figure 3-2).

```
Browse : /qibm/UserData/qwebqry/apps/simona/cen_orders.acx
Record :       1   of       1 by  18                      Column :    1
Control :

...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+.
************Beginning of data**************
SEGNAME=CEN_ORDERS, TABLENAME=QWQCENT/ORDERS, CONNECTION=*LOCAL, KEYS=0, $
************End of Data********************
```

Figure 3-1 Access file for the QWQCENT/ORDERS table

```
Browse : /qibm/UserData/qwebqry/apps/simona_10/cen_orders.acx
Record :       1   of       7 by  18                      Column :    1
Control :

...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+..
************Beginning of data**************
SEGNAME=CEN_ORDERS, TABLENAME=ORDERS, CONNECTION=*LOCAL, KEYS=0, $
 FOREIGN_KEY=Q_QWQCENT_ORDERS_PROD_NUM_00001, PRIMARY_KEY_TABLE=INVENTORY,
   FOREIGN_KEY_COLUMN=PRODUCTNUMBER, PRIMARY_KEY_COLUMN=PRODUCTNUMBER, $
 FOREIGN_KEY=Q_QWQCENT_ORDERS_PLANT_CODE_00001, PRIMARY_KEY_TABLE=PLANT,
    FOREIGN_KEY_COLUMN=PLANTCODE, PRIMARY_KEY_COLUMN=PLANTCODE, $
 FOREIGN_KEY=Q_QWQCENT_ORDERS_STORE_CODE_00001, PRIMARY_KEY_TABLE=STORES,
    FOREIGN_KEY_COLUMN=STORECODE, PRIMARY_KEY_COLUMN=STORECODE, $
************End of Data********************
```

Figure 3-2 Access file for the QWQCENT/ORDERS table with RI

- Master file: This file identifies the fields of the table or, if the object is a QRYDFN or a Stored Procedure, the fields in the result set. The master file has an extension of .mas. The fields are described in terms of length and data type. This file is similar to the information in a DSPFFD file, but the information is described in a way in which DB2 Web Query understands. Figure 3-3 shows a master file for table QWQCENT/ORDERS.

  Similar to the access file, the master file uses a keyword-value pairing technique to define the column attributes. These include the name of the field, the ALIAS (the actual underlying field name in the file), the data type and length (for both how it is used in reports and actually defined in the file), and the title to be used in report column headings.

  For QRYDFN objects, there are three additional metadata files:

  – .fex
  – .txt
  – .inf

In DB2 Web Query V2R1M0, metadata files can be created in a “common folder” where they can be used by everybody who has access to the product or can be created in a “private” directory that is associated to a folder defined in the product. This has been done to achieve two purposes:

- Security: Synonyms associated to an application folder can only be modified by those who have development rights on the specific folder.
- Privacy: Synonyms associated to an application folder are presented only to those who can develop in the specific folder and are only usable by those who can use the folder and its contain.

An additional advantage of metadata is that it is an abstraction layer. You can build business logic into the metadata so that all reports and graphs that reference those synonyms have access to all of that logic. This greatly simplifies the environment for the report developers.

```
Browse : /qibm/UserData/qwebqry/apps/simona_10/cen_orders.mas
Record :       1   of      33 by  18                      Column :    1
70 by 131
Control :

...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+..
..8....+..
************Beginning of data**************
FILENAME=CEN_ORDERS, SUFFIX=DB2    ,
REMARKS='Orders table for Web Query QWQCENT DB', $
 SEGMENT=CEN_ORDERS, SEGTYPE=SO, $
  FIELDNAME=ORDERNUMBER, ALIAS=ORDERNUMBER, USAGE=A5, ACTUAL=A5,
    TITLE='Order,Number', $
  FIELDNAME=PRODUCTNUMBER, ALIAS=PRODUCTNUMBER, USAGE=A4, ACTUAL=A4,
    TITLE='Product,Number', $
  FIELDNAME=ORDERDATE, ALIAS=ORDERDATE, USAGE=YYMD, ACTUAL=DATE,
    TITLE='Order,Date', $
  FIELDNAME=STORECODE, ALIAS=STORECODE, USAGE=A6, ACTUAL=A6,
    TITLE='Store,Code', $
  FIELDNAME=PLANTCODE, ALIAS=PLANTCODE, USAGE=A3, ACTUAL=A3,
    TITLE='Plant,Code', $
  FIELDNAME=SALESREP, ALIAS=SALESREP, USAGE=A50, ACTUAL=A50,
    TITLE='Sales Rep', $
  FIELDNAME=QUANTITY, ALIAS=QUANTITY, USAGE=I11, ACTUAL=I4,
    MISSING=ON,
     TITLE='Quantity', $
FIELDNAME=LINETOTAL, ALIAS=LINETOTAL, USAGE=P22.2, ACTUAL=P11,
     MISSING=ON,
     TITLE='Revenue', $
    FIELDNAME=COSTOFGOODSSOLD, ALIAS=COSTOFGOODSSOLD, USAGE=P22.2,
ACTUAL=P11,
     MISSING=ON,
     TITLE='Cost of,Goods Sold', $
   FIELDNAME=RETURNS, ALIAS=RETURNS, USAGE=I11, ACTUAL=I4,
     MISSING=ON,
     TITLE='Returns', $
  FIELDNAME=WARRANTYEXP, ALIAS=WARRANTYEXP, USAGE=P22.2, ACTUAL=P11,
     MISSING=ON,
     TITLE='Warranty,Expenses', $
  FIELDNAME=SHIPPINGCOST, ALIAS=SHIPPINGCOST, USAGE=P22.2, ACTUAL=P11,
     MISSING=ON,
     TITLE='Shipping,Cost', $
 ************End of Data********************
```

Figure 3-3 DB2 Web Query master file example

3.1.1 Benefits of metadata

The primary objective of the metadata layer is simple: Improve the productivity of the DB2 Web Query report developers by providing an abstraction layer and burying the database complexity. Report developers can be more productive if you keep the data model simple. A simple and intuitive data model also means that you can extend the report developer community because it does not require an intimate knowledge of the database. By empowering more and more of your users, you can reduce their dependence on IT and the number of backlogged reporting requests.

But before this can happen, some work needs to be done to build up this abstraction layer. While in previous releases of the product, an additional tool, DB2 Web Query Developer Workbench, was needed, in DB2 Web Query V2R1M0, it is possible to create and edit synonyms via the web interface.

The web based metadata editor allows you to perform these tasks:

- Define database joins.
- Build virtual columns to centralize business logic.
- Standardize column formats.
- Convert and standardize date fields.
- Create filters.
- Create business views.

3.2 Creating metadata

DB2 Web Query metadata can be generated in multiple ways. Each technique is discussed next:

- Create your own metadata.

  Metadata creation wizards are available from both the Web browser ("Creating metadata with the web interface" on page 26) via the Web Query launch page and in the DB2 Web Query Developer Workbench tool; we discuss more on this tool later ("Creating metadata with Developer Workbench" on page 30). Alternatively, you can create metadata programmatically using the CRTWQSYN CL command ("Creating metadata with the CL command CRTWQSYN" on page 34). To create metadata, you must be a DB2 Web Query DBA as explained in Chapter 4, "Security Center" on page 149.

  Metadata can be created within your application folder (then it is available only in your folder) or in the Common application folder (metadata available from all applications).

- Create metadata through the use of third-party tools.

  Here are some examples of this approach:

  – The Databorough x-WebQuery product (`www.databorough.com`) generates the metadata based on the data model that it derives from both database definitions and business logic buried in application programs. It can also use this extracted information to generate the necessary dimensions for instant OLAP capabilities.

  – Information Builders's iWay Data Migrator (`www.ibi.com`) and Coglin Mill's RODIN DB2 Web Query Edition (`www.thinkrodin.com`) products generate the metadata in conjunction with building a data mart or data warehouse.

- Provided by ERP or services provider.

  ISVs have the ability to include DB2 Web Query content as part of their solutions package. If you purchase such a distribution, the metadata will be included in the package. Similarly, a services provider may have the expertise and tools to build the metadata for you.

- JDE Adapter provides metadata for JDE as is explained in Chapter 21, "Assignment #14: Creating JD Edwards reports" on page 693.

3.2.1 Creating metadata with the web interface

In most cases, you are going to create your own metadata using the provided metadata creation wizards. To do this, take the following steps:

1. Open a Web browser and enter the URL of the DB2 Web Query home page.
2. On the DB2 Web Query home page, expand the folder in which you want to work. Right-click the specific report folder and select **Metadata** (Figure 3-4).

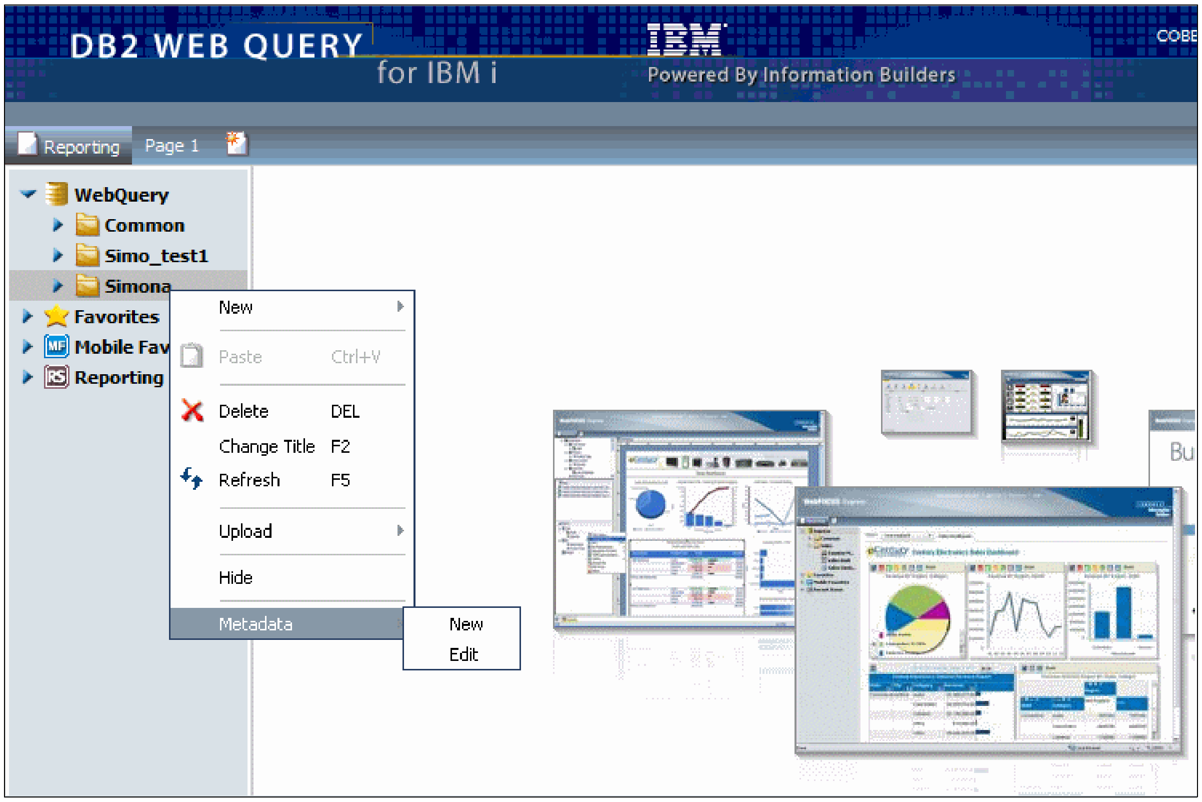
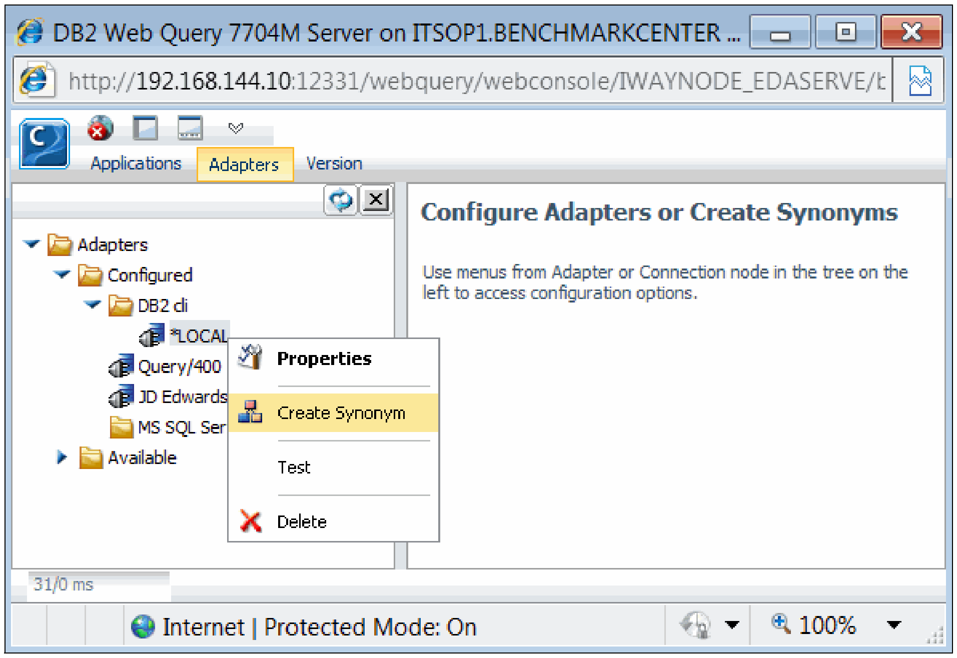


Figure 3-4 Metadata

3. In the new pane, on the left-hand side, select the adapter that you want to use; in this example, we want to use DB2 data on the local system, hence select **DB2 CLI**. You will see the *LOCAL connection appear below it. Right-click ***LOCAL** and select **Create Synonym**, as shown in Figure 3-5. If you have configured a connection to remote databases or installed other adapters, they will be presented in the list.

   To learn more about data adapters, see 1.4.5, “Data adapters” on page 11.

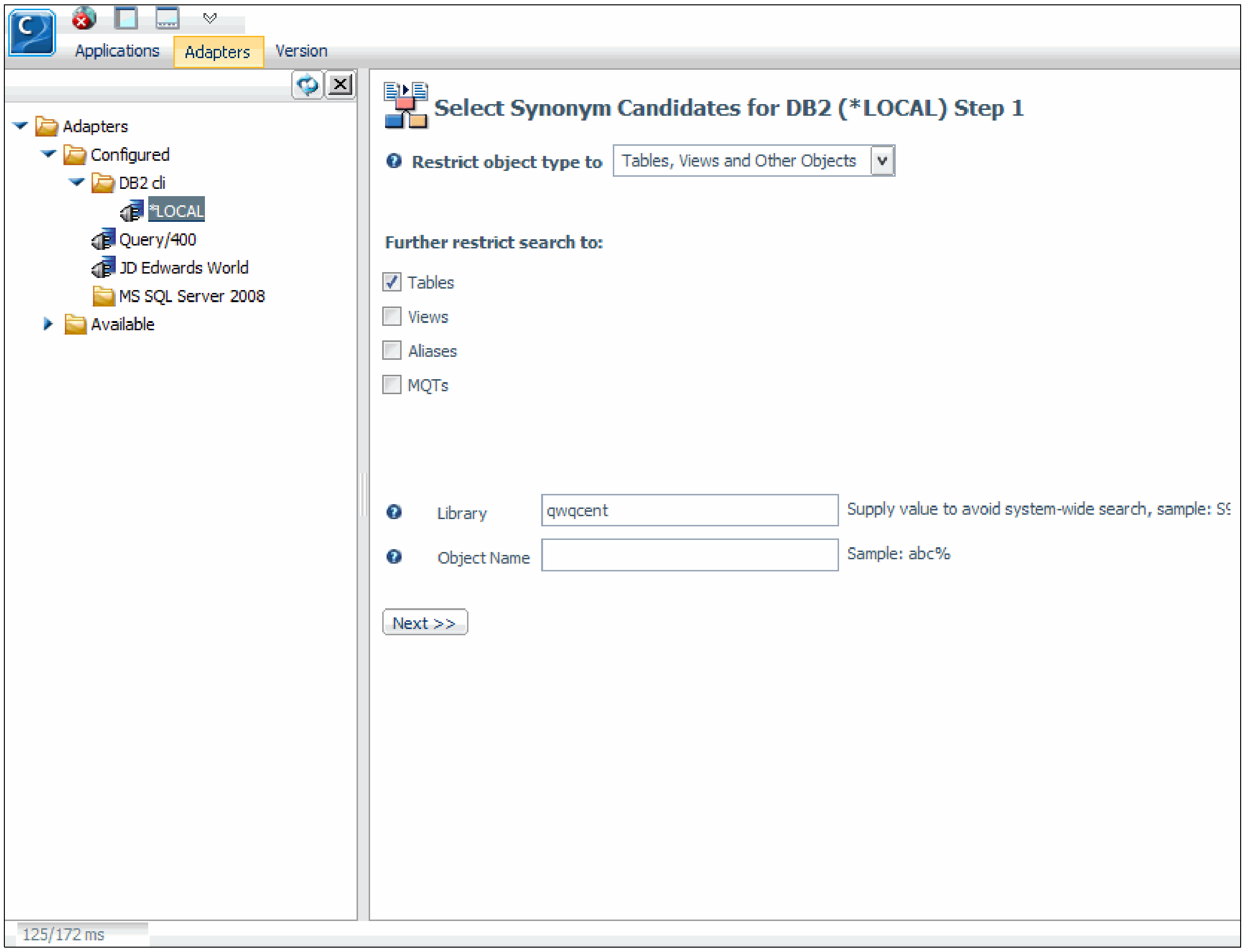

DB2 Web Query 7704M Server on ITSOP1.BENCHMARKCENTER ...
http://192.168.144.10:12331/webquery/webconsole/IWAYNODE_EDASERVE/t
Applications
Adapters
Version
Adapters
Configured
DB2 cli
*LOCAL
Query/400
JD Edwards
MS SQL Ser
Available
Properties
Create Synonym
Test
Delete
31/0 ms
Internet | Protected Mode: On
100%

Configure Adapters or Create Synonyms

Use menus from Adapter or Connection node in the tree on the left to access configuration options.

Figure 3-5 Create Synonym for DB2 CLI

4. In the Select Synonym Candidates for DB2 CLI pane (Figure 3-6), enter your collection/library name. Select the type of data that you want to query. In our example, we select **Tables** because want to create metadata on a table in QWQCENT. In the Library field, type `qwqcent`. Click **Next**.



Select Synonym Candidates for DB2 (*LOCAL) Step 1

Restrict object type to: Tables, Views and Other Objects

Further restrict search to:

- [✓] Tables
- [ ] Views
- [ ] Aliases
- [ ] MQTs

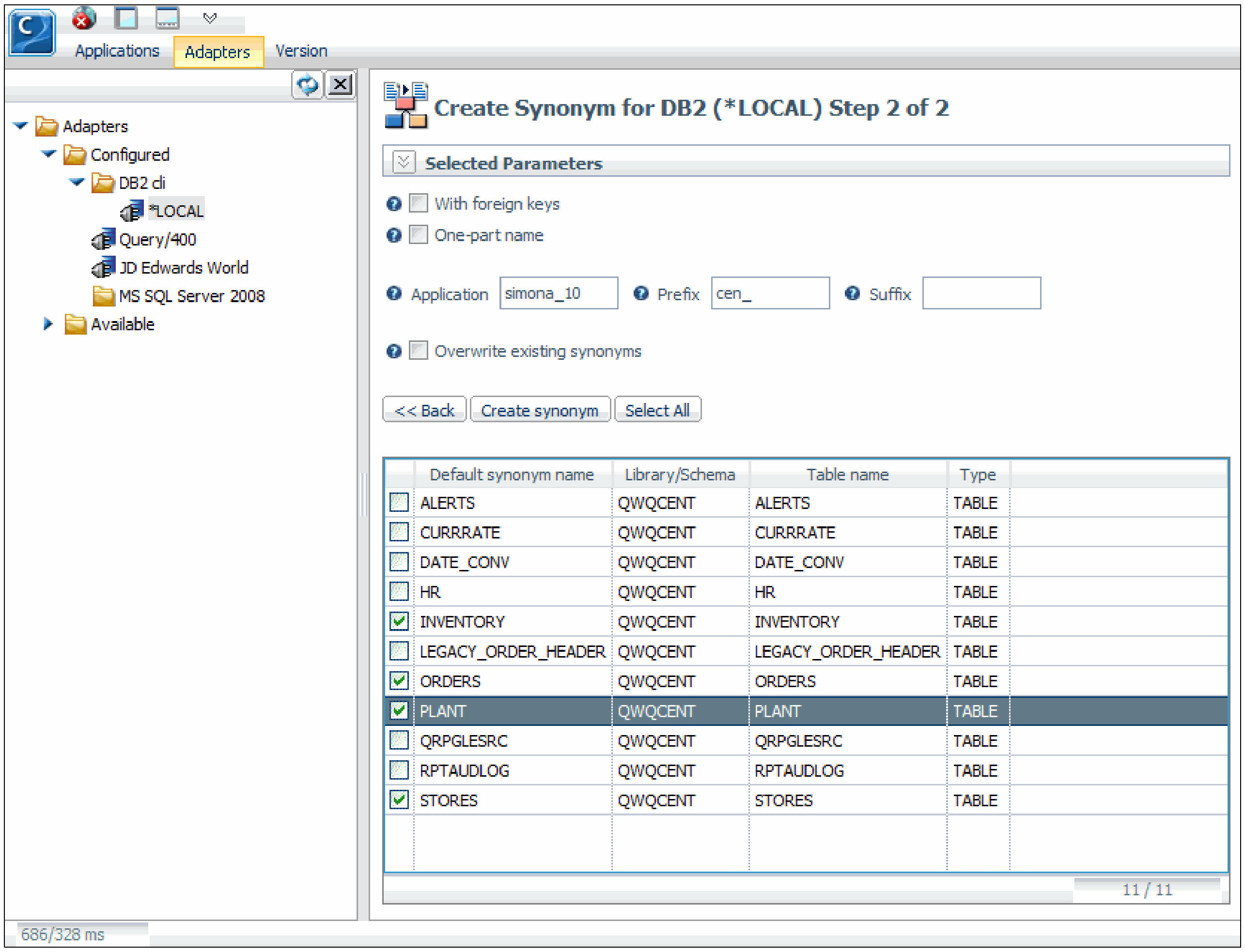
Library: qwqcent — Supply value to avoid system-wide search, sample: S

Object Name: — Sample: abc%

Next >>

Figure 3-6 Select Synonym Candidates

5. The Create Synonym for DB2 CLI pane is displayed (Figure 3-7). It shows all the different tables that reside on the QWQCENT schema. Select the table names for which you want to create reports.



Create Synonym for DB2 (*LOCAL) Step 2 of 2

Selected Parameters

- [ ] With foreign keys
- [ ] One-part name

Application: simona_10 Prefix: cen_ Suffix:

- [ ] Overwrite existing synonyms

	Default synonym name	Library/Schema	Table name	Type
☐	ALERTS	QWQCENT	ALERTS	TABLE
☐	CURRRATE	QWQCENT	CURRRATE	TABLE
☐	DATE_CONV	QWQCENT	DATE_CONV	TABLE
☐	HR	QWQCENT	HR	TABLE
✓	INVENTORY	QWQCENT	INVENTORY	TABLE
☐	LEGACY_ORDER_HEADER	QWQCENT	LEGACY_ORDER_HEADER	TABLE
✓	ORDERS	QWQCENT	ORDERS	TABLE
✓	PLANT	QWQCENT	PLANT	TABLE
☐	QRPGLESRC	QWQCENT	QRPGLESRC	TABLE
☐	RPTAUDLOG	QWQCENT	RPTAUDLOG	TABLE
✓	STORES	QWQCENT	STORES	TABLE

11 / 11

Figure 3-7 Selecting tables from a schema or library

You might want to include a prefix or suffix. In our example, cen_ is our prefix because the table resides in library QWQCENT. We leave the suffix blank, but the developer has the choice to use it. In the Application field, you find the name of the folder in which you are creating metadata.

If you want your metadata to have a different name than the underlying table, double-click the name and edit it.

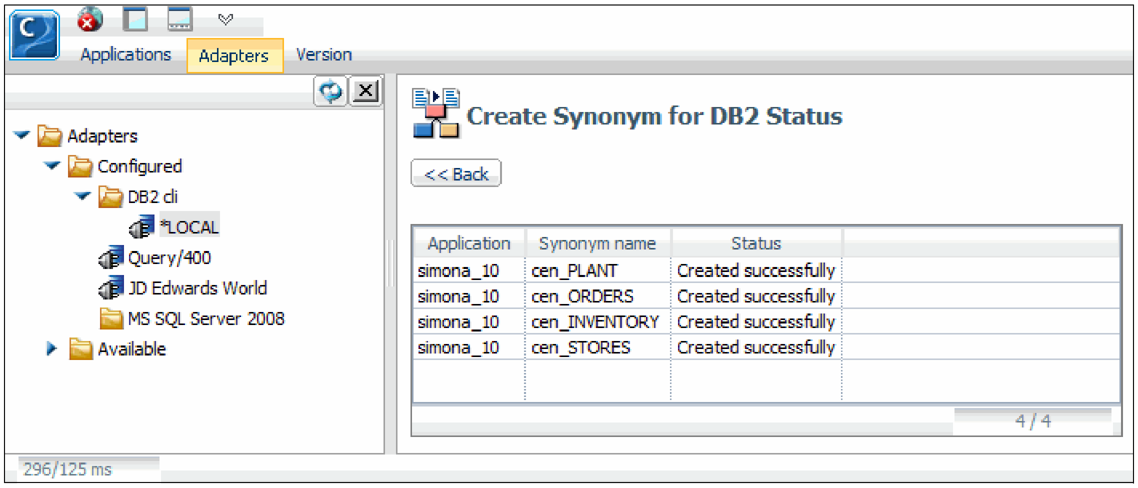
Click the **Create synonym** button.

Prefix and suffix (optional, yet recommended): The prefix and suffix are optional letters that you can add to your synonym name to provide extra meaning for you. Using a prefix or suffix is not required. It is up to the developer to decide whether it is necessary to use the prefix and the naming convention that is preferred.

We recommend that you use the library name or an abbreviated version of the library name for the prefix. This way, when you create your reports, you can search on all metadata that starts with the library name.

Keep in mind that all metadata is displayed in one box called Database Descriptions. The box is small and lists the metadata in alphabetical order, not according to library.

6. The metadata takes a few seconds to create, depending on how many items you selected. After the processing is done, in the Status column, you see the message `Created successfully`, as shown in Figure 3-8.



Create Synonym for DB2 Status

Application	Synonym name	Status
simona_10	cen_PLANT	Created successfully
simona_10	cen_ORDERS	Created successfully
simona_10	cen_INVENTORY	Created successfully
simona_10	cen_STORES	Created successfully

Figure 3-8 Synonyms created

7. The process of creating metadata is now complete. Close the message box.

3.2.2 Creating metadata with Developer Workbench

Metadata can also be created using the client-server development tool, Development Workbench.

1. Open Developer Workbench, select the system on which to work, and validate with your userid and password. Remember that the user must be part of the DevWorkbench group (use **Administration** → **Security Center** in the web interface to set up as documented in Chapter 4, “Security Center” on page 149.

2. Select **Data Servers** → **EDASERVE** → **Applications** and right-click the folder where you want to create the new synonym and select **New** → **Synonym** as shown in Figure 3-9.

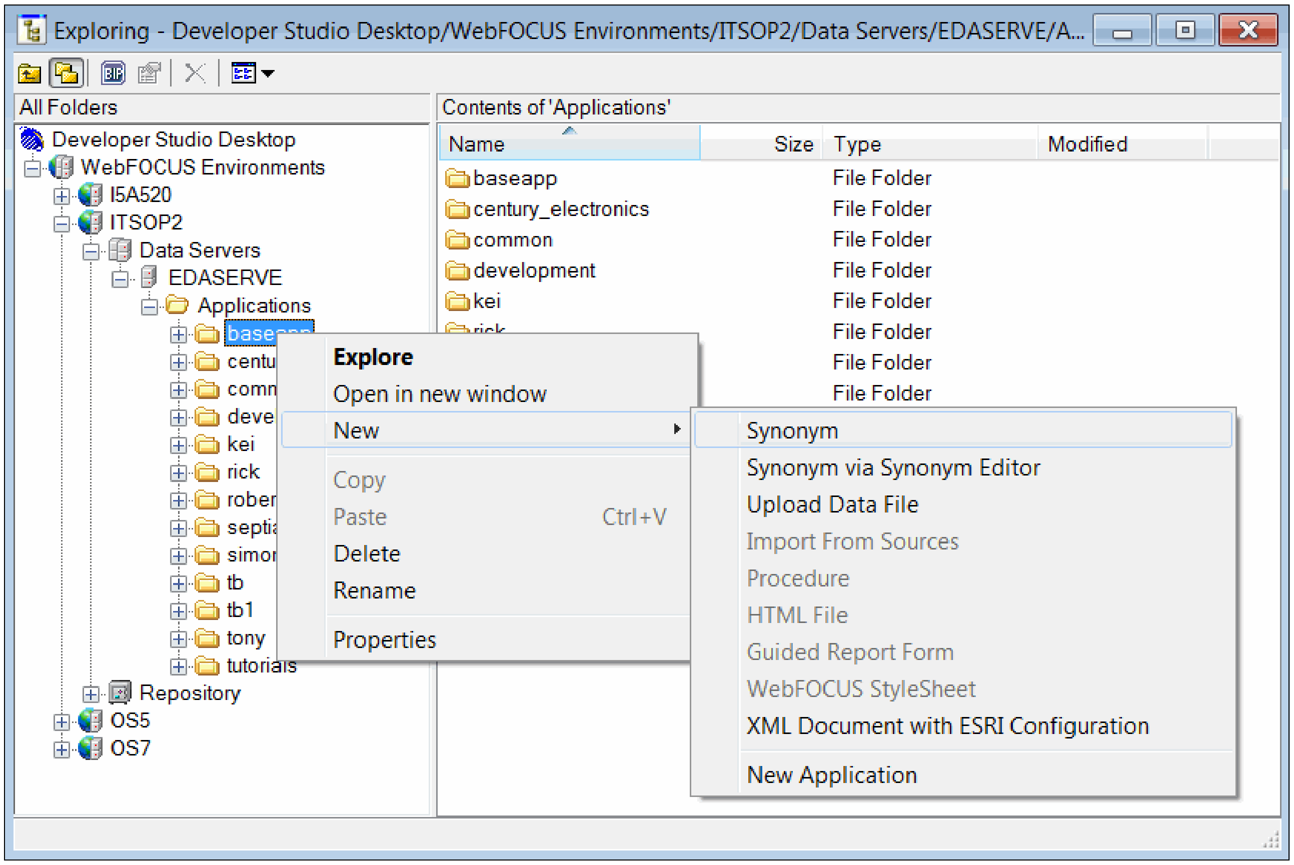


Figure 3-9 Create Synonym with Developer Workbench - Step 1

3. You are presented with the Select Adapter window, select ***LOCAL** and click **OK** (Figure 3-10).

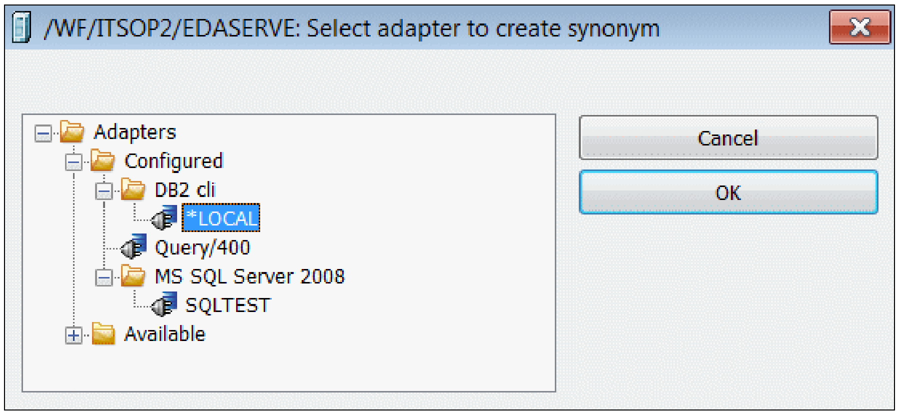
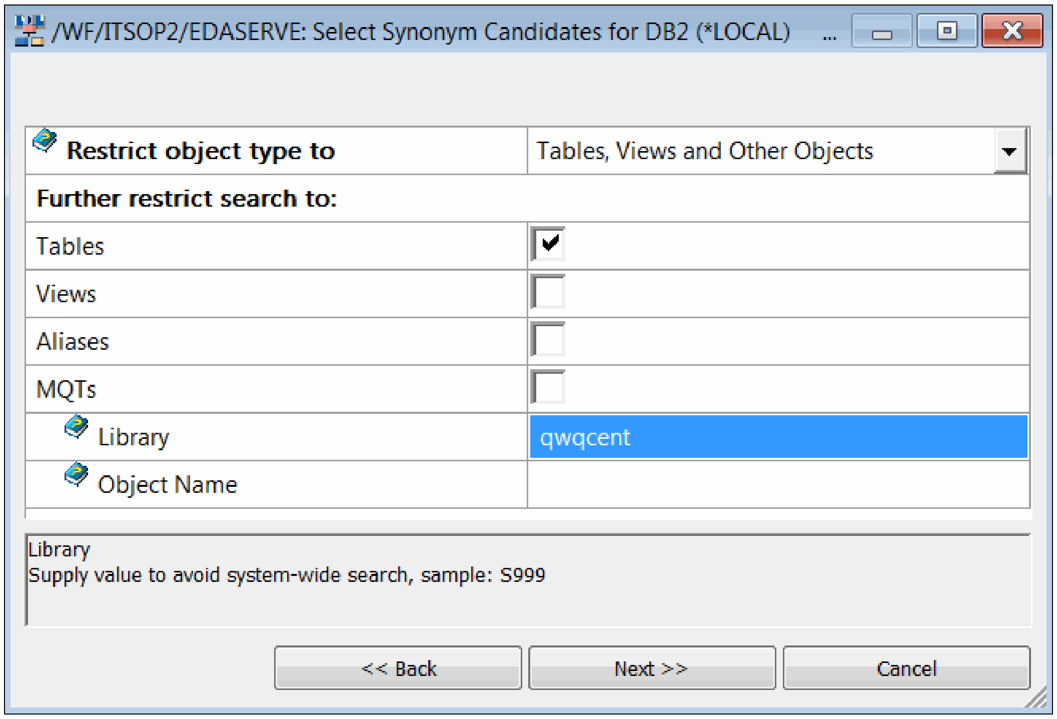


Figure 3-10 Create Synonym with Developer Workbench - Step 2

4. In the “Select synonym candidate” window, check “**Tables**”, position in the “**Library**” field and type in the name of the library where to look for tables to be referenced (**QWQCENT** in our example), then click **Next** as shown in Figure 3-11.

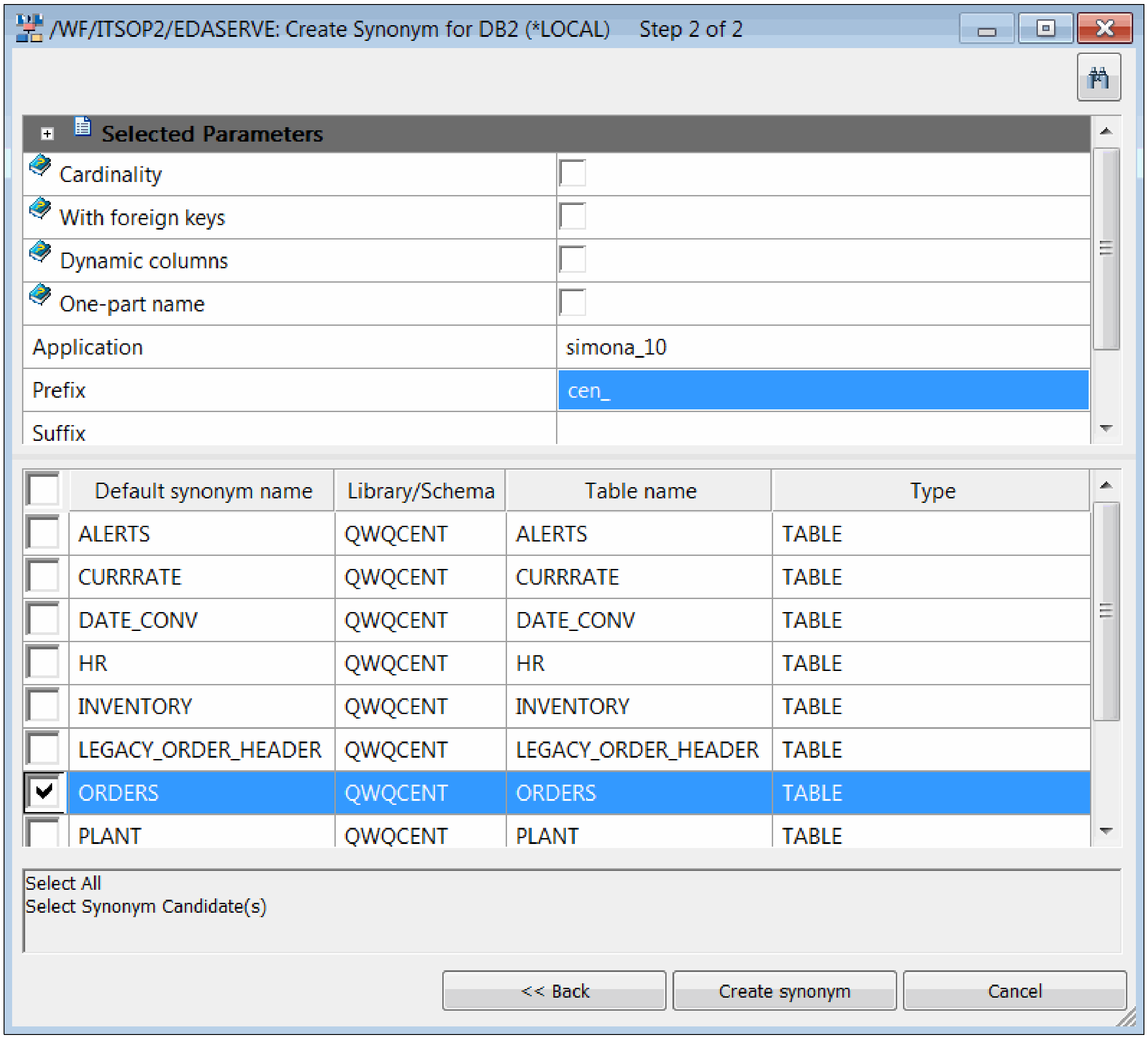

/WF/ITSOP2/EDASERVE: Select Synonym Candidates for DB2 (*LOCAL) ...

Restrict object type to	Tables, Views and Other Objects
Further restrict search to:	
Tables	✓
Views	
Aliases	
MQTs	
Library	qwqcent
Object Name	

Library
Supply value to avoid system-wide search, sample: S999

Figure 3-11 Create Synonym with Developer Workbench - Step 3

5. You are presented with a panel where you can choose the table(s) for which to create synonyms. The same considerations detailed in 3.2.1, “Creating metadata with the web interface” on page 26 apply here. In this example, we use prefix **CEN_** and select table **ORDERS** as shown in Figure 3-12. When you are done, select **Create synonym**.


/WF/ITSOP2/EDASERVE: Create Synonym for DB2 (*LOCAL) Step 2 of 2

Selected Parameters

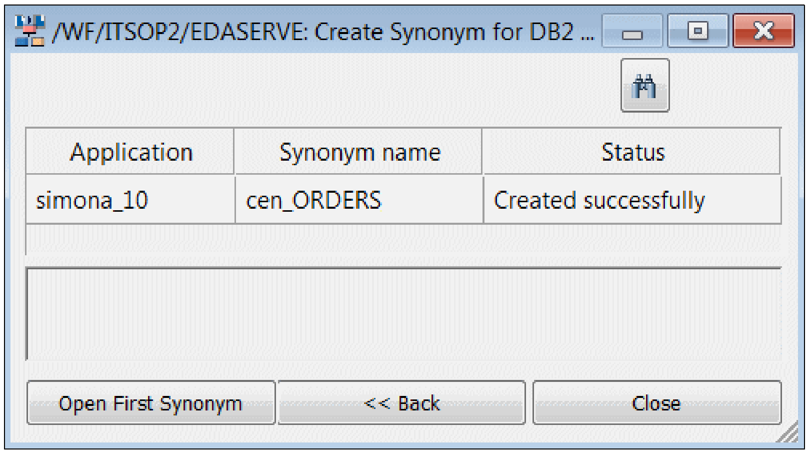
Cardinality	☐
With foreign keys	☐
Dynamic columns	☐
One-part name	☐
Application	simona_10
Prefix	cen_
Suffix	

☐	Default synonym name	Library/Schema	Table name	Type
☐	ALERTS	QWQCENT	ALERTS	TABLE
☐	CURRRATE	QWQCENT	CURRRATE	TABLE
☐	DATE_CONV	QWQCENT	DATE_CONV	TABLE
☐	HR	QWQCENT	HR	TABLE
☐	INVENTORY	QWQCENT	INVENTORY	TABLE
☐	LEGACY_ORDER_HEADER	QWQCENT	LEGACY_ORDER_HEADER	TABLE
✓	ORDERS	QWQCENT	ORDERS	TABLE
☐	PLANT	QWQCENT	PLANT	TABLE

Select All
Select Synonym Candidate(s)

Figure 3-12 Create Synonym with Developer Workbench - Step 4

6. You are presented with a message confirming the synonym creation (Figure 3-13). Select **Close**.



Application	Synonym name	Status
simona_10	cen_ORDERS	Created successfully

Figure 3-13 Create Synonym with Developer Workbench - Step 5

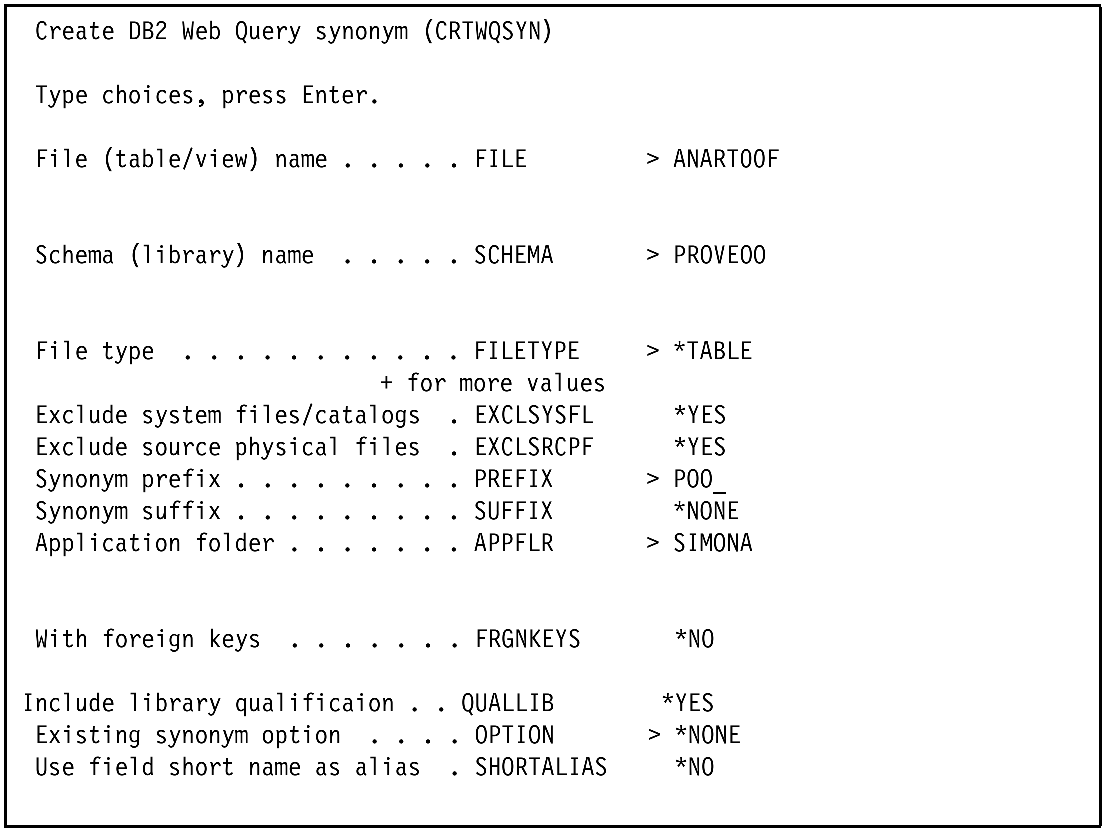
3.2.3 Creating metadata with the CL command CRTWQSYN

As an alternative to the product's browser interface, it is now possible to use the **CRTWQSYN** command in the library QWEBQRY to create metadata. This command can be used from a 5250 interactive session, inserted in programs (to automate and schedule a mass refresh action) or used with Job Scheduler. See Figure 3-14 for an example of using the CRTWQSYN command to create a new synonym.

The CRTWQSYN CL command has the following parameters:

- FILE - Enter the name of the database file object to create the DB2 Web Query synonym over.
- SCHEMA - Enter the name of the database schema (library) which contains the files that you want to create DB2 Web Query synonyms over.
- FILETYPE - When *ALL is specified for the file parameter, specifies what types of files are to be searched for and included in the synonym creation process. Up to four file types can be specified.
- EXCLSYSFL - When *ALL or a generic* value is specified for the file parameter, specifies whether to exclude system files and catalogs when locating and creating synonyms for the files in the specified schema.
- EXCLSRCPF - When *ALL or a generic* value is specified for the file parameter, specifies whether to exclude source physical files when locating and creating synonyms for the files in the specified schema.
- PREFIX - Specifies the prefix to append to the beginning of the synonym name. If you have tables with identical table names, assign a prefix or a suffix to distinguish them. For example, if you have identically named human resources and payroll tables, assign the prefix HR to distinguish the synonyms for the human resources tables. Note that the resulting synonym name cannot exceed 64 characters.
- SUFFIX - Specifies the suffix to append to the end of the synonym name. If you have tables with identical table names, assign a prefix or a suffix to distinguish them. For example, if you have identically named human resources and payroll tables, assign the prefix HR to distinguish the synonyms for the human resources tables. Note that the resulting synonym name cannot exceed 64 characters.

- APPFLR - Specifies the name of the application folder where the synonym is to be created.
- FRGNKEYS - Specifies whether the synonym should include foreign key relationship definitions. If specified, the synonym being created will automatically include every table related by a foreign key to the table specified in the FILE parameter. The resulting multi-table synonym describes all table foreign key relationships.
- QUALLIB - Specifies whether to generate a library qualified table name in the DB2 Web Query synonym Access file (.acx) for the specified table.
- OPTION - Specifies what action to take if the specified synonym name (including prefix and suffix) already exists in the specified application folder.
- SHORTALIAS - When generating the table's column level information in the Master (.mas) file, specifies whether to use the table's system (short) name for the synonym's alias. This allows the DB2 Web Query report developer to see and use both the SQL (long) name of the column as well as the system (short) name when using the report development tools.



```
 Create DB2 Web Query synonym (CRTWQSYN)

 Type choices, press Enter.

 File (table/view) name . . . . . FILE         > ANARTOOF


 Schema (library) name  . . . . . SCHEMA       > PROVEOO


 File type  . . . . . . . . . . . FILETYPE     > *TABLE
                           + for more values
 Exclude system files/catalogs  . EXCLSYSFL      *YES
 Exclude source physical files  . EXCLSRCPF      *YES
 Synonym prefix . . . . . . . . . PREFIX       > POO_
 Synonym suffix . . . . . . . . . SUFFIX         *NONE
 Application folder . . . . . . . APPFLR       > SIMONA


 With foreign keys  . . . . . . . FRGNKEYS       *NO

Include library qualificaion . . QUALLIB        *YES
 Existing synonym option  . . . . OPTION       > *NONE
 Use field short name as alias  . SHORTALIAS     *NO
```

Figure 3-14 Create a new synonym using CL command CRTWQSYN

3.3 Deleting metadata

When metadata is no longer needed, you may want to delete it.

You do not have to delete it explicitly if you simply want to recreate it (there is an option to overwrite existing synonyms in the creation wizard) or if you want to update the metadata to reflect a change in the underlining files. See “Refreshing metadata when IBM i database object structure changes” on page 39.

There are three ways of deleting metadata:

- Using the browser interface
- Using the Developer Workbench tool
- Deleting corresponding files in the IFS

3.3.1 Deleting metadata with the web interface

To delete the metadata, proceed as follows:

1. On the folder where the metadata is located, right-click, **Metadata** → **Edit** (Figure 3-15).

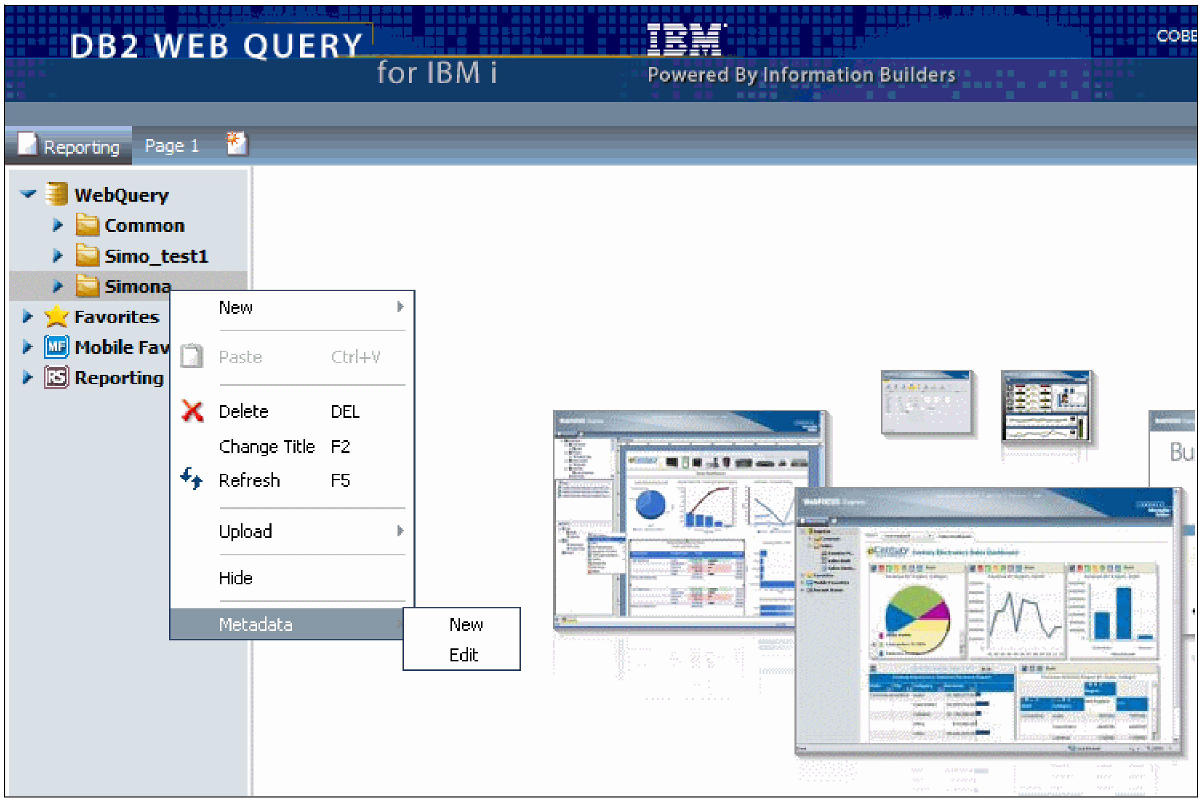


Figure 3-15 Deleting metadata with web interface - Step 1

2. In the metadata pane, right-click the metadata you want to delete and select **Delete** (Figure 3-16).

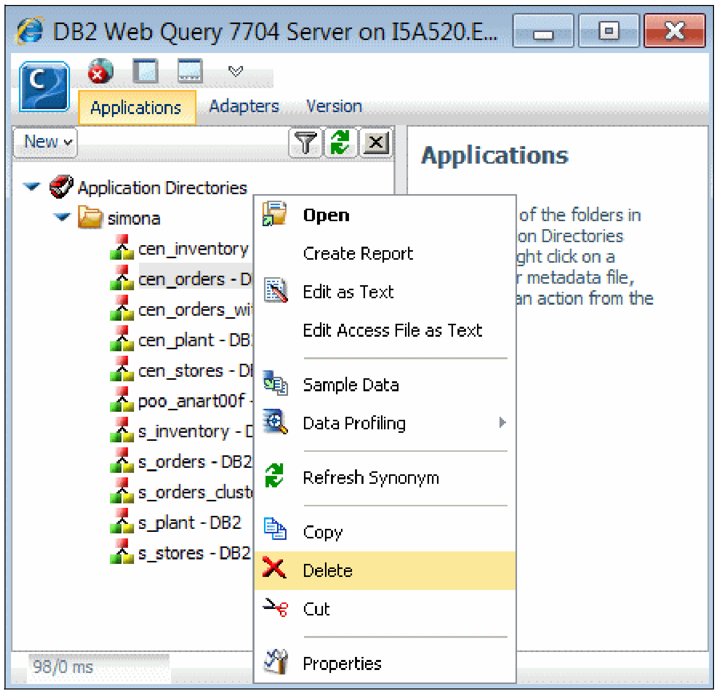


Figure 3-16 Deleting metadata with web interface - Step 2

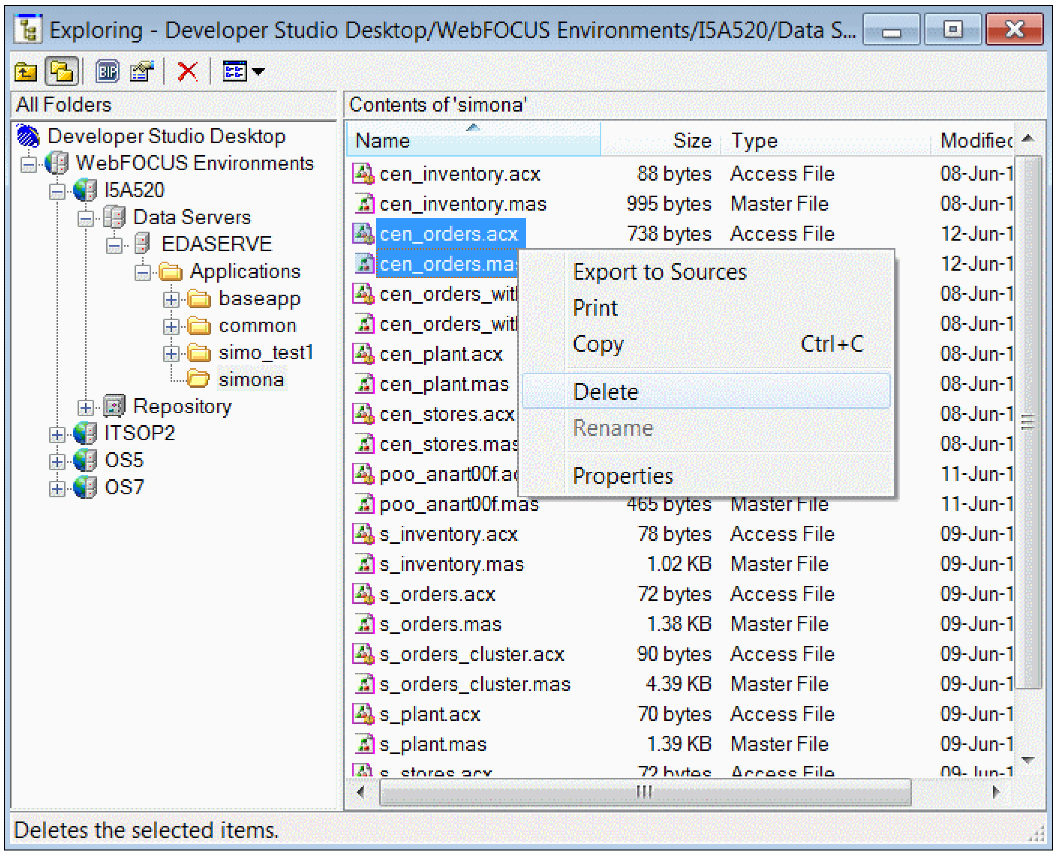
3. You will receive a message asking you to confirm deletion. Click **OK**.

3.3.2 Deleting metadata with the Developer Workbench tool

An additional client server tool, Developer Workbench, can be used to create, edit, and delete synonyms:

1. Open Developer Workbench; select the system on which to work, and validate with your userid and password. Remember that the user must be part of the DevWorkbench group (use **Administration** → **Security Center** in the web interface to set up).
2. Select **Data Servers** → **EDASERVE** → **Applications** → ***<yourfolder>*** and in the right-hand side of the panel, select the metadata that you want to delete, as shown in Figure 3-17. Right-click the files highlighted and select **Delete**.

Note: A metadata synonym is made up of two files, a .acx file and a .mas file. When using Developer Workbench to delete a synonym, only the .mas file needs to be selected; both .acx and .mas files will be deleted. If you select both, you may get an error message, however, the synonym gets correctly deleted.

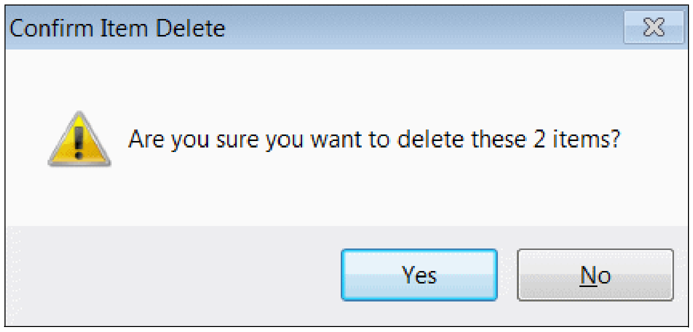


Name	Size	Type	Modifiec
cen_inventory.acx	88 bytes	Access File	08-Jun-1
cen_inventory.mas	995 bytes	Master File	08-Jun-1
cen_orders.acx	738 bytes	Access File	12-Jun-1
cen_orders.ma:			12-Jun-1
cen_orders_wit			08-Jun-1
cen_orders_wit			08-Jun-1
cen_plant.acx			08-Jun-1
cen_plant.mas			08-Jun-1
cen_stores.acx			08-Jun-1
cen_stores.mas			08-Jun-1
poo_anart00f.ac			11-Jun-1
poo_anart00f.mas	465 bytes	Master File	11-Jun-1
s_inventory.acx	78 bytes	Access File	09-Jun-1
s_inventory.mas	1.02 KB	Master File	09-Jun-1
s_orders.acx	72 bytes	Access File	09-Jun-1
s_orders.mas	1.38 KB	Master File	09-Jun-1
s_orders_cluster.acx	90 bytes	Access File	09-Jun-1
s_orders_cluster.mas	4.39 KB	Master File	09-Jun-1
s_plant.acx	70 bytes	Access File	09-Jun-1
s_plant.mas	1.39 KB	Master File	09-Jun-1
s_stores.acx	72 bytes	Access File	09-Jun-1

Deletes the selected items.

Figure 3-17 Deleting metadata, Dev. Workbench - Step 1

3. You will receive a message prompting you to confirm delete. Select **Yes** (Figure 3-18).



Are you sure you want to delete these 2 items?

Figure 3-18 Deleting metadata, Dev. Workbench - confirm

3.3.3 Deleting files in the IFS

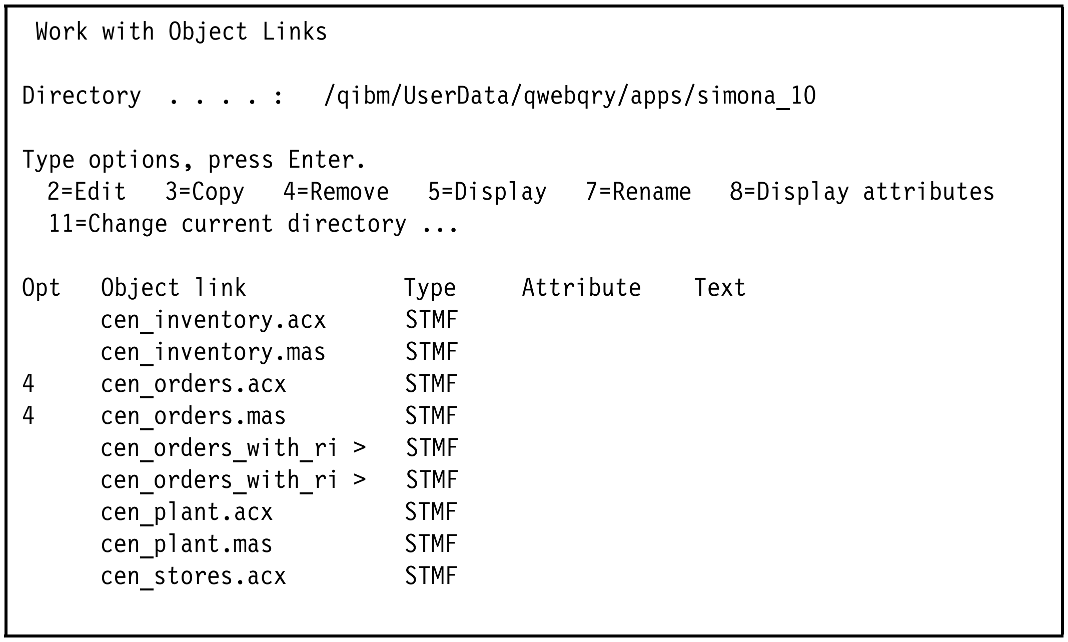
To delete the files, proceed as follows:

1. From the System i command line, enter the following command:

```
WRKLNK OBJ('/qibm/UserData/qwebqry/apps/<myfolder>')
```

2. Select option 5 for directory <*myfolder*>.

3. Browse the <*myfolder*> directory for the metadata that you want to delete. When you find it, select option 4 (Figure 3-19) and press **Enter** to delete.



```
 Work with Object Links

Directory  . . . . :   /qibm/UserData/qwebqry/apps/simona_10

Type options, press Enter.
  2=Edit   3=Copy   4=Remove   5=Display   7=Rename   8=Display attributes
  11=Change current directory ...

Opt   Object link           Type     Attribute    Text
      cen_inventory.acx     STMF
      cen_inventory.mas     STMF
4     cen_orders.acx        STMF
4     cen_orders.mas        STMF
      cen_orders_with_ri >  STMF
      cen_orders_with_ri >  STMF
      cen_plant.acx         STMF
      cen_plant.mas         STMF
      cen_stores.acx        STMF
```

Figure 3-19 Deleting metadata via 5250

Note: Remember that a metadata synonym is made up of two files, .acx and .mas. Both files have to be selected if you want to delete the synonym with the IFS interface or commands.

To delete files in the IFS, it is also possible to use CL command **DEL**. For instance, to delete metadata CEN_ORDERS in folder SIMONA_10 you would use the following command:

```
DEL OBJLNK('/qibm/UserData/qwebqry/apps/simona_10/cen_orders.*')
```

Using .* it is possible to catch both .acx and .mas files in one delete command.

3.4 Refreshing metadata when IBM i database object structure changes

The database structure can be subject to changes, for example, if the application is updated:

- New tables are defined
- Existing tables get deleted
- New columns are added to the table
- Columns get deleted because they are not used any more

What happens to metadata and the report that make inquiries on our data? This is a common concern for many customers, what happens when the metadata gets out of sync with the underlying data source? For example, if the database administrator alters a table and either adds a new column or removes an existing one, do existing reports still run successfully?

The answer to this question is that it depends. The synonym may need to be refreshed to reflect this change. If none of your reports reference a column deleted from the underlying file, a synonym refresh is not required. The same applies if you do not want to include the new column in any existing or new reports. However, if the changes do impact your reports, you must synchronize the synonym with the altered file. For this, you can use the new synonym interface provided by DB2 Web Query, Developer Workbench or CL command CRTWQSYN.

The refresh option is only supported in these environments:

- Synonyms based on DB2 CLI adapter:
  - The following adapters are NOT supported:
    - Query/400
    - JD Edwards
    - SQL Server
- *LOCAL databases only
  - No support for remote objects
- Supported file types:
  - Table
  - Physical File
  - SQL View
  - Logical file

Using the CRTWQSYN command, *ALL and generic (wildcard) search are supported in the FILE parameter. Useful for refreshing ALL files in a library (or all files that start with ORD*).

3.4.1 Refreshing metadata with the browser interface

1. In the folder where the metadata was created, select **Metadata** → **Edit** as shown in Figure 3-20.

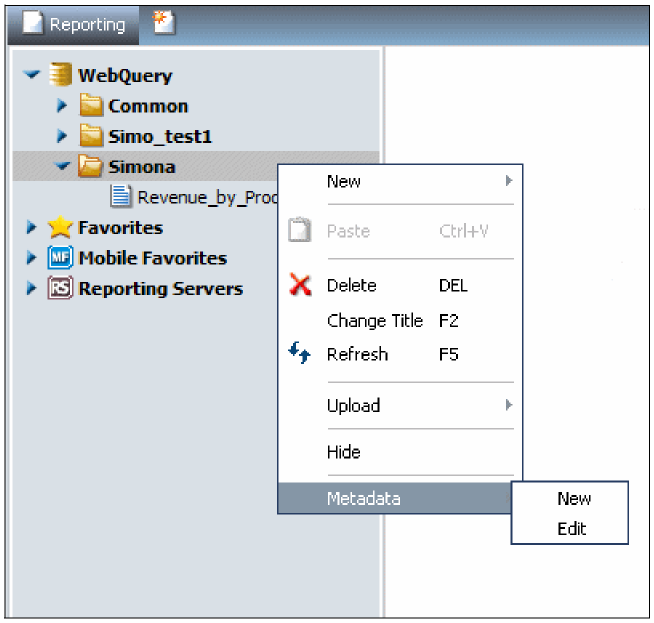


Figure 3-20 Refresh Synonym - web - Step 1

2. You are taken into the Metadata management window. Right-click the synonym that needs to be refreshed and select **Refresh synonym** as shown in Figure 3-21.

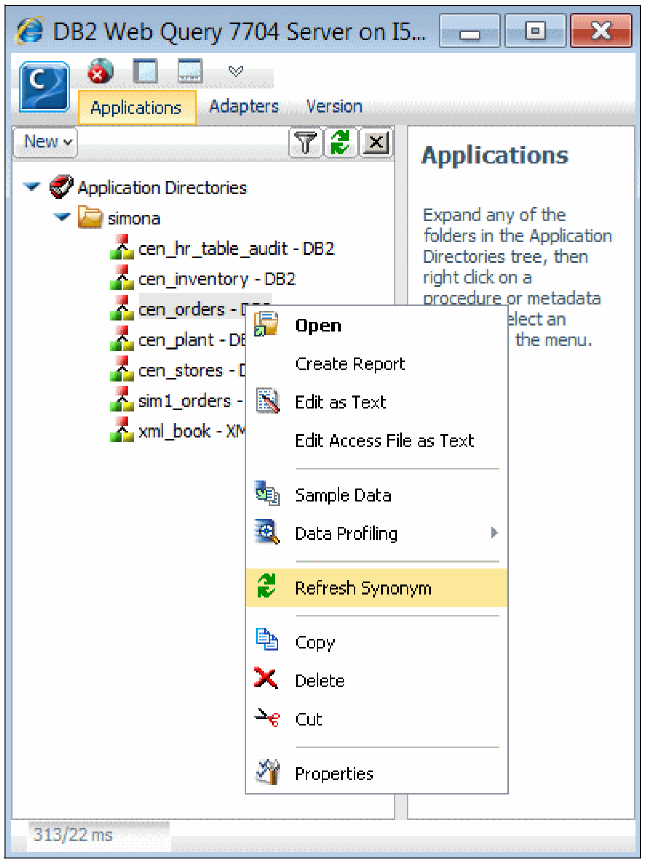


Figure 3-21 Refresh Synonym - web - Step 2

When this action is taken, the synonym is parsed and its contents are compared with the structure of the actual underlying data source. New columns are added to the synonym and deleted columns are removed. Any custom changes (joins, virtual columns, filters, OLAP dimensions, and so on) that were made to the synonym are preserved.

3.4.2 Refreshing metadata with Developer Workbench

An additional client server tool, Developer Workbench, can be used to create, edit, and delete synonyms:

1. Open Developer Workbench; select the system on which to work, and validate with your userid and password. Remember that the user must be part of the DevWorkbench group (use **Administration** → **Security Center** in the web interface to set up).
2. Select **Data Servers** → **EDASERVE** → **Applications** → ***<yourfolder>***, in the right-hand side of the panel, right-click the metadata that you want to refresh, and select **Refresh** as shown in Figure 3-22.

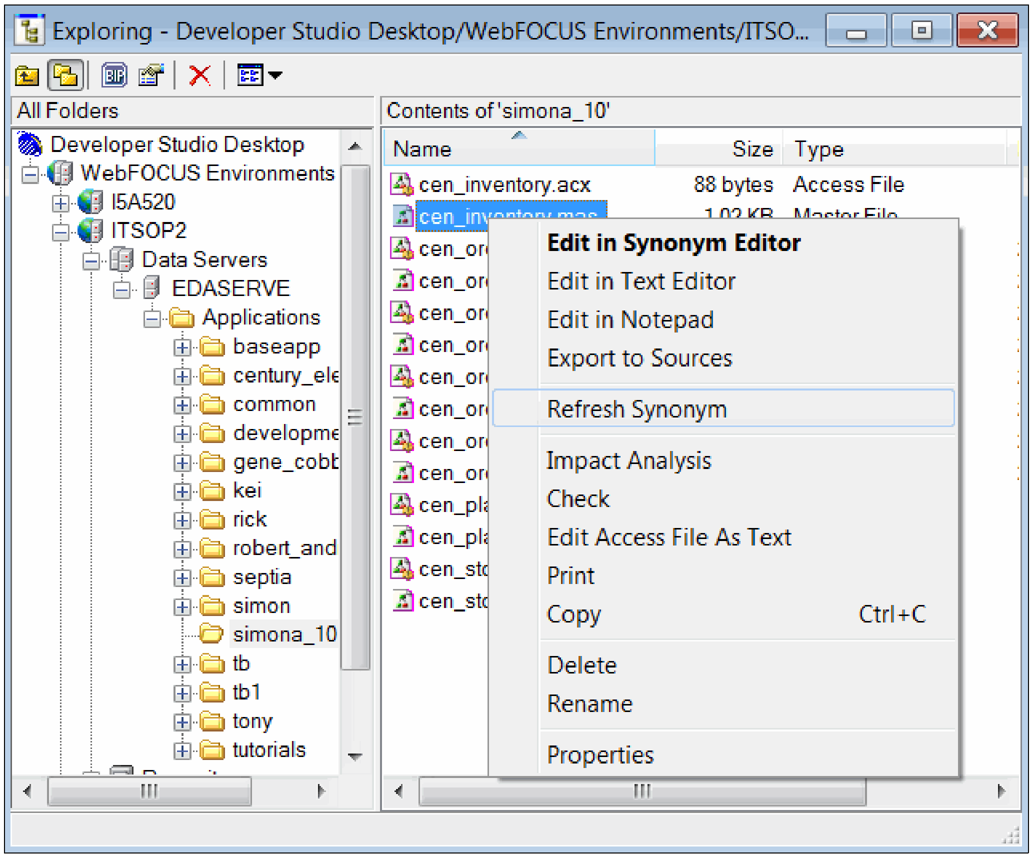
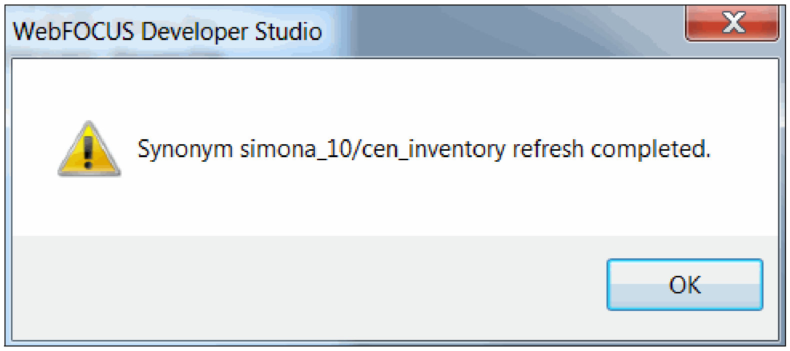


Figure 3-22 Developer Workbench - refresh synonym

3. You will be presented with a message box confirming that the synonym has been refreshed, as shown in Figure 3-23. Select **OK**.



Synonym simona_10/cen_inventory refresh completed.

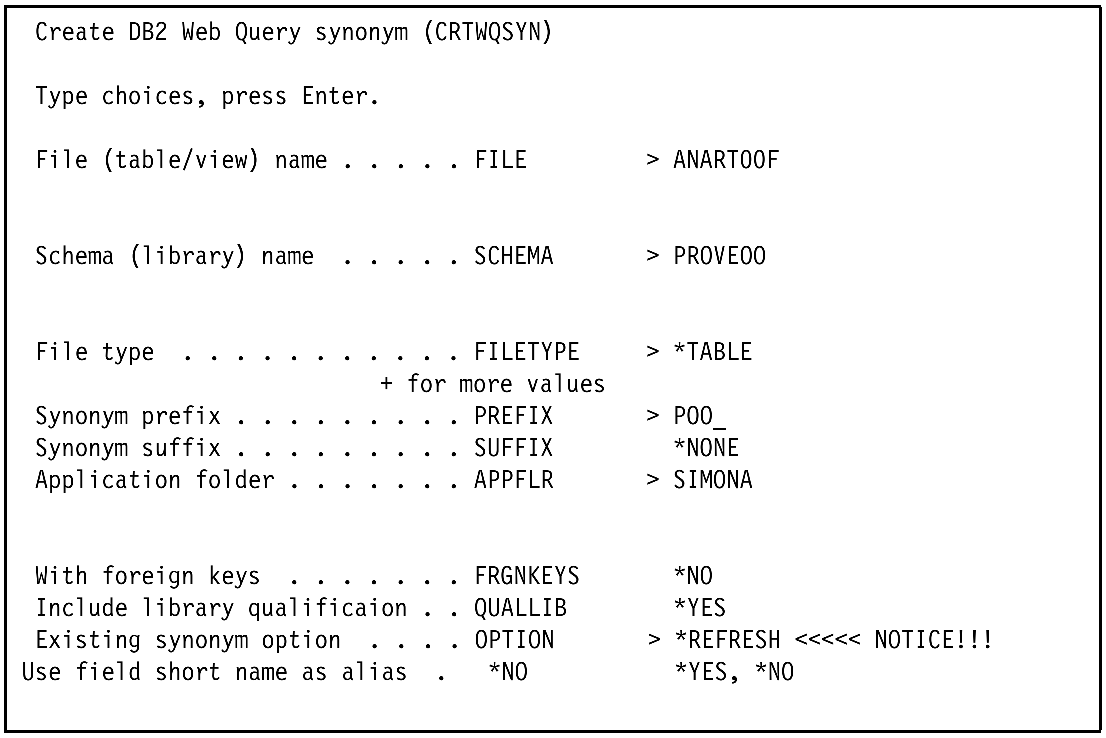
Figure 3-23 Developer Workbench - refresh completed

3.4.3 Refreshing metadata with the command CRTWQSYN

As an alternative to the product's browser interface, it is now possible to use the command CRTWQSYN in library QWEBQRY to perform metadata refresh (Figure 3-24). This command can be used from a 5250 interactive session, inserted in programs (to automate and schedule a mass refresh action) or used with Job Scheduler.

The CRTWQSYN CL command has the following parameters:

- FILE - Enter the name of the database file object to create the DB2 Web Query synonym over.
- SCHEMA - Enter the name of the database schema (library) which contains the files you want to create DB2 Web Query synonyms over.
- FILETYPE - When *ALL is specified for the file parameter, specifies what types of files are to be searched for and included in the synonym creation process. Up to four file types can be specified.
- EXCLSYSFL - When *ALL or a generic* value is specified for the file parameter, specifies whether to exclude system files and catalogs when locating and creating synonyms for the files in the specified schema.
- EXCLSRCPF - When *ALL or a generic* value is specified for the file parameter, specifies whether to exclude source physical files when locating and creating synonyms for the files in the specified schema.
- PREFIX - Specifies the prefix to append to the beginning of the synonym name. If you have tables with identical table names, assign a prefix or a suffix to distinguish them. For example, if you have identically named human resources and payroll tables, assign the prefix HR to distinguish the synonyms for the human resources tables.Note that the resulting synonym name cannot exceed 64 characters.
- SUFFIX - Specifies the suffix to append to the end of the synonym name. If you have tables with identical table names, assign a prefix or a suffix to distinguish them. For example, if you have identically named human resources and payroll tables, assign the prefix HR to distinguish the synonyms for the human resources tables. Note that the resulting synonym name cannot exceed 64 characters.
- APPFLR - Specifies the name of the application folder where the synonym is to be created.
- FRGNKEYS - Specifies whether the synonym should include foreign key relationship definitions. If specified, the synonym being created will automatically include every table related by a foreign key to the table specified in the FILE parameter. The resulting multi-table synonym describes all table foreign key relationships.
- QUALLIB - Specifies whether to generate a library qualified table name in the DB2 Web Query synonym Access file (.acx) for the specified table.
- OPTION - Specifies what action to take if the specified synonym name (including prefix and suffix) already exists in the specified application folder.
- SHORTALIAS - When generating the table's column level information in the Master (.mas) file, specifies whether to use the table's system (short) name for the synonym's alias. This allows the DB2 Web Query report developer to see and use both the SQL (long) name of the column as well as the system (short) name when using the report development tools.



```
Create DB2 Web Query synonym (CRTWQSYN)

Type choices, press Enter.

File (table/view) name . . . . . FILE         > ANARTOOF


Schema (library) name  . . . . . SCHEMA       > PROVEOO


File type  . . . . . . . . . . . FILETYPE     > *TABLE
                           + for more values
Synonym prefix . . . . . . . . . PREFIX       > POO_
Synonym suffix . . . . . . . . . SUFFIX         *NONE
Application folder . . . . . . . APPFLR       > SIMONA


With foreign keys  . . . . . . . FRGNKEYS       *NO
Include library qualificaion . . QUALLIB        *YES
Existing synonym option  . . . . OPTION       > *REFRESH <<<<< NOTICE!!!
Use field short name as alias  .   *NO          *YES, *NO
```

Figure 3-24 Refresh synonym using the CRTWQSYN CL command

Attention: The Refresh Synonym will not update the synonym for all database object changes. For example, if column attributes such as data type or length are altered in the database object, those changes are not applied to the synonym. For those types of changes, the synonym must either be manually edited or recreated.

3.4.4 What happens to metadata if the IBM i database object is deleted

If you delete the underlying database object or the QRYDFN object, the DB2 Web Query report cannot run based on the metadata alone. If you attempt to run the report in DB2 Web Query, you receive an error message. You have the option to recreate the table or QRYDFN exactly as it was before. Therefore, you do not need to recreate the metadata. Or if you create a new, yet slightly different object, you must create new metadata on that object. Or you can delete all reports and underlying metadata if this information is not needed any more.

3.5 Miscellaneous considerations about creating metadata

The DB2 CLI adapter can create metadata on five different IBM i5/OS object types:

- Tables
- Views
- Aliases
- Stored procedures
- MQTs

Tables, views, aliases, and MQTs are treated identically by DB2 Web Query. All of them can be used to filter, join, sort, define, compute, and access all the functionality of Report Assistant. After you create the metadata, you will be unable to tell which object type you are using in your report.

3.5.1 Stored procedure synonyms

A stored procedure is a program or procedure in a service program that can be called by an application using the SQL CALL statement. Stored procedures can be written in the SQL procedural language, or you can use existing programs or service program procedures (written in RPG, COBOL, JAVA, and so on) and register them to the database as stored procedures.

When it comes to DB2 Web Query data sources to base their reports on, many developers only use database objects like tables and views. But a very powerful, yet under-utilized feature of DB2 Web Query is its ability to use a stored procedure as a data source, provided that the stored procedure returns a result set. DB2 Web Query has the ability to capture that result set and use it as the source data for a report.

When a DB2 Web Query synonym is created over a stored procedure, the result set is used as the format for synonym. This means that all fields in the result set can be used as columns in the report. In addition, the input parameters of the stored procedures can be used as input parameters for the report. Consequently, you can pass parameter values from a report to the stored procedure, let the procedure use those values to perform the business logic, and return the results to the report.

This is a powerful technique because it gives the report developer programmatic control over what is returned to DB2 Web Query. Consider the following ways that a stored procedure could be used with the tool:

- Reuse and repurpose the business logic in your existing programs.You can take an existing RPG program (or one written in any language supported by the IBM i), make small modifications to return a result set, and register it to the database as a stored procedure. This means that many existing reports generated by RPG programs could be converted to work with DB2 Web Query. Comment out the header lines and change the details line to write to a result set, temporary file, or array, rather than a print file record. By combining DB2 Web Query with existing report programs, you can salvage proven business logic and provide a modernized output format for those *old* reports.
- Use native record level access (RLA), instead of SQL, if you prefer that method of data access.
- Provide the ability to call to other programs, commands, and system APIs as well as pull data from other system objects such messages queues, data queues, or data areas.

- Provide a way to dynamically change the library list by passing an input parameter to the stored procedure (and changing the library list based on that parameter value), then execute the appropriate process to return the result set.
- Adopt authority of the user profile that owns the underlying program or service program. This gives you the ability to restrict access to the database objects and only allow users to access the objects through the programs (stored procedures) with adopted authority.
- Provide auditing capability. The stored procedure can include logic to insert a row in an audit log table to record any report request. With so much attention given to security and auditing these days (that is, Sarbanes-Oxley), this can be very important consideration, especially for users who have access to sensitive information.
- Provide the ability to programmatically change attributes of the querying job. Here are some examples of what can be specified in a stored procedure to change the environment:
  - SET CURRENT DEGREE to enable Symmetric Multi Processing (SMP) and boost query performance of a long-running report if the requestor is an executive or other high-profile user
  - SET OPTION SRTSEQ to change the collating sequence of the report

To illustrate how a stored procedure could be used, let us assume that you need to create a report that returns rows from a sensitive table (the HR table) and you must add an auditing feature to this report. That is, each time the report is run, information such as the time stamp, name of the report, and requesting user profile must be logged to an audit table.

Note: This example uses material from the QWQCENT sample library. The requested tables and stored procedure may already be there.

To create a stored procedure that returns data from specific columns in the payroll table and satisfies the auditing requirement, take the following steps:

1. From an SQL interface such as System i Navigator Run SQL Scripts, create the audit table:

```
CREATE TABLE qwqcent.rptaudlog (rpttimst TIMESTAMP, rptname CHAR(25),rptusrprf
CHAR(10))
```

2. From same SQL interface, create the stored procedure:

```
CREATE PROCEDURE qwqcent.hr_table_audit (inplantcode CHAR(3))
   DYNAMIC RESULT SETS 1
   LANGUAGE SQL
   NOT DETERMINISTIC
   MODIFIES SQL DATA
   P1 : BEGIN
      DECLARE c1 CURSOR WITH RETURN TO CLIENT FOR
      SELECT lastname, firstname, status, payscale, salary
        FROM qwqcent.hr
      WHERE plantcode = inplantcode;
      INSERT INTO qwqcent.rptaudlog VALUES(CURRENT TIMESTAMP,
        'HR Salary Report', SESSION_USER);
      OPEN c1 ;
   END P1  ;
```

Several items must be pointed out for this store procedure:

- The cursor is declared with the “WITH RETURN TO CLIENT” clause specified. It is a good idea to specify this clause to ensure that the result set is returned to the client application.

"WITH RETURN TO CALLER" is the default, which would cause problems in the event that you had a nested stored procedure (a stored procedure called by another stored procedure) that was actually returning the result set.

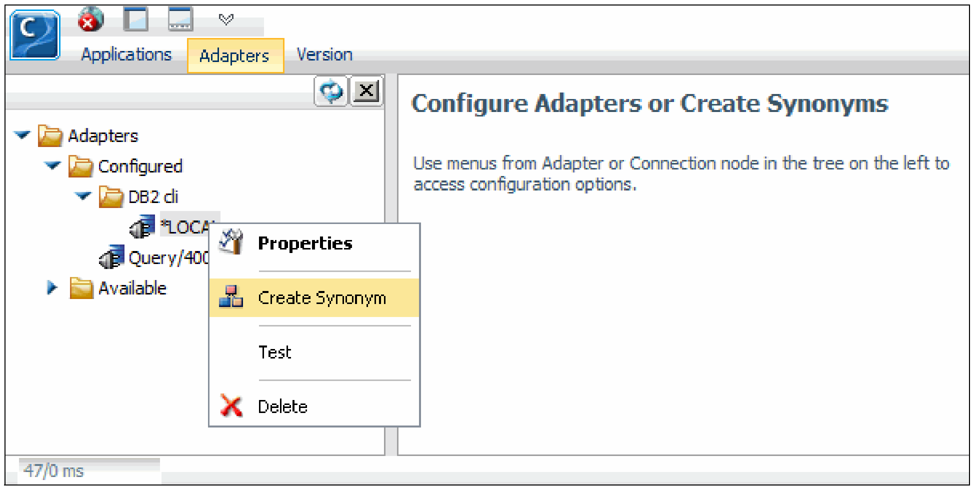
- The INSERT statement is specified to log the request. This satisfies the auditing requirement.
- The cursor is opened (and left open). This returns a result set to the client application (which is DB2 Web Query).

3. Check that your stored procedure is working:

```
CAll PROCEDURE qwqcent.hr_table_audit ('BOS');
```

The process for creating a stored procedure synonym is similar to that of creating a synonym over a table. Take the following steps:

1. Right-click the folder where you want to create the synonym and select **Metadata** → **New**. You are presented with the synonym wizard, select the **DB2 CLI adapter**, right-click ***LOCAL**, and select **Create synonym** as shown in Figure 3-25.

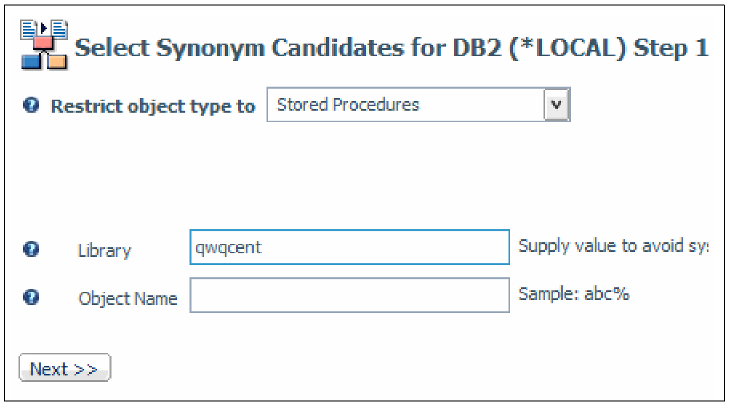


Configure Adapters or Create Synonyms

Use menus from Adapter or Connection node in the tree on the left to access configuration options.

Figure 3-25 Creating stored procedure synonym: Step 1

2. In the Select Synonym Candidates select **Stored Procedure**, type in the library name (`QWQCENT` in our example) and select **Next** as shown in Figure 3-26.



Select Synonym Candidates for DB2 (*LOCAL) Step 1

Restrict object type to: Stored Procedures

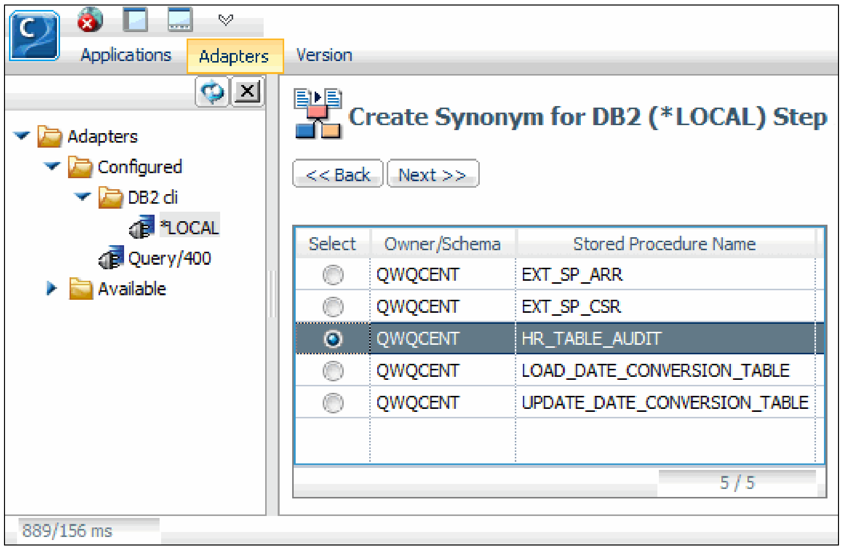
Library: qwqcent — Supply value to avoid sy

Object Name: — Sample: abc%

Next >>

Figure 3-26 Stored Procedure synonym, select candidate

3. From the list of stored procedures displayed, select **HR_TABLE_AUDIT**, as shown in Figure 3-27.

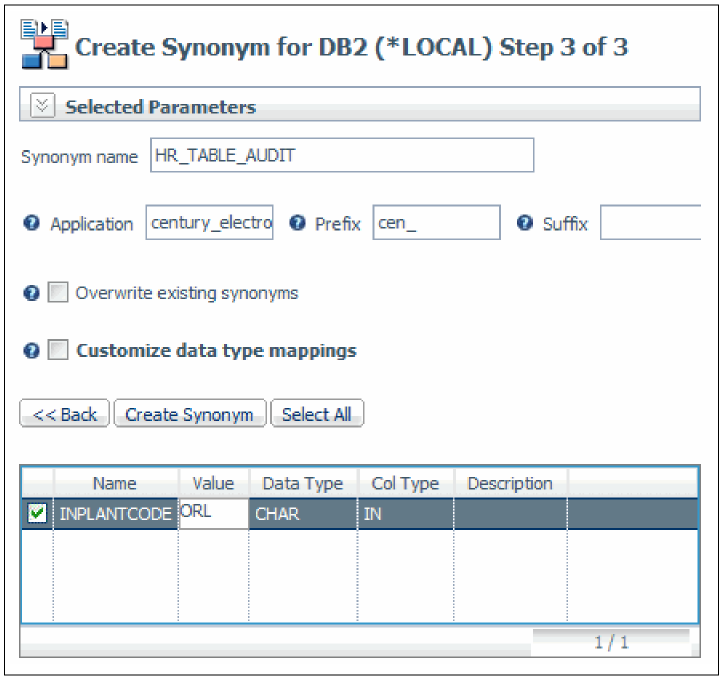


Select	Owner/Schema	Stored Procedure Name
○	QWQCENT	EXT_SP_ARR
○	QWQCENT	EXT_SP_CSR
●	QWQCENT	HR_TABLE_AUDIT
○	QWQCENT	LOAD_DATE_CONVERSION_TABLE
○	QWQCENT	UPDATE_DATE_CONVERSION_TABLE

Figure 3-27 Creating stored procedure synonym: Step 2

4. If the stored procedure has input parameters, you will be prompted to enter a valid value for the input parameter. As demonstrated in Figure 3-28, specify a valid input parameter value by taking these steps:
   a. Select/check the INPLANTCODE parameter.
   b. Specify ORL as the value for the input parameter. (This is a valid Plant Code value.)
   c. Select **Create Synonym**.

   Note: When the synonym is created, the stored procedure is called by DB2 Web Query so that it can receive the result set. This is because it must store the format of the result set in the synonym. Consequently, you must pass it a valid value for the input parameter at this step.


Create Synonym for DB2 (*LOCAL) Step 3 of 3

Selected Parameters

Synonym name	HR_TABLE_AUDIT
Application	century_electro
Prefix	cen_
Suffix	

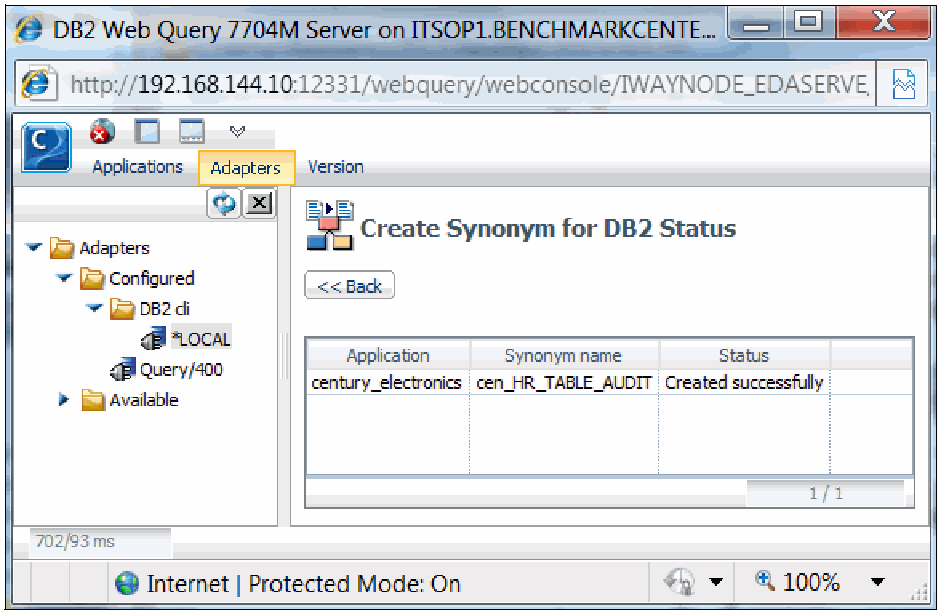
☐ Overwrite existing synonyms

☐ Customize data type mappings

	Name	Value	Data Type	Col Type	Description
✓	INPLANTCODE	ORL	CHAR	IN	

Figure 3-28 Creating stored procedure synonym: Step 3

5. As shown in Figure 3-29, a confirmation window is displayed to confirm that the stored procedure synonym was successfully created. Close this window.


DB2 Web Query 7704M Server on ITSOP1.BENCHMARKCENTE...

http://192.168.144.10:12331/webquery/webconsole/IWAYNODE_EDASERVE

Create Synonym for DB2 Status

Application	Synonym name	Status
century_electronics	cen_HR_TABLE_AUDIT	Created successfully

Figure 3-29 Stored procedure synonym created

For examples using stored procedure with DB2 Web Query, see Chapter 19, “Assignment #12: Stored procedures in action” on page 613.

Multiple result sets: DB2 Web Query supports the return of multiple result sets from the stored procedure. In this case, it assigns a distinct segment name to the fields of each separate result set.

3.5.2 Considerations with multimember files

SQL can only read data from the first member in a file. Perhaps you have run into this situation when writing SQL queries on multimember files before. It is possible to use an SQL alias to point to a member within a multimember file and then create a synonym against the SQL alias. This way, you will be able to use the DB2 CLI adapter to get to data in a member and the SQL statement issued will be eligible for SQE.

An alias is a permanent i5/OS object that *points* to a specific member in a file. It is easy to create, and after it exists, SQL treats the alias the same as it would a table. Simply substitute the alias name in any query where you would put a table name.

This is an example of a CREATE ALIAS command in SQL:

```
CREATE ALIAS QGPL/MYALIAS FOR QGPL/MULTI_MBR_FILE (MBR_NUM_2)
```

You can run this command in the Run SQL Scripts window of iSeries Navigator or in STRSQL from the command line. STRSQL has a prompt for CREATE ALIAS that is intuitive.

You can create metadata on an alias as easily as you can on a regular table. See “Creating metadata” on page 25, which explains how to create DB2 Web Query metadata.

3.6 Joining database objects

In DB2 Web Query, it is possible to define joins to other metadata (that refer to database objects) at the report level as it was done in Query/400. This approach is still working, but it is not very productive and it is error prone:

- If you need 2 tables to be joined in 150 reports, you have to repeat this step 150 times.
- In some of the previously mentioned 150 reports, the join definition may be set incorrectly, thus generating incorrect data in the report output.

To overcome these problems, the person in charge with defining and maintaining metadata (DBA role in the folder) may decide to define join relationships directly in the metadata, thus developing metadata that are comprehensive of all the joins that can be needed in the various reports. This is one of the great advantages provided by metadata in a reporting environment.

As an alternative, joins can be defined at the database level, creating SQL views to define the joins and then building metadata on the SQL view.

If the table(s) on which metadata is created have a referential integrity relationship defined, the metadata creation wizard can detect that there are “natural” joins available among the tables and have the capability to describe it in the metadata. Hence a single synonym will be created, referencing all objects and their relationships. This is described in “Benefit of using referential integrity in DB2 Web Query” on page 51.

If you do not have Referential Integrity defined, you can set up joins in metadata using either the web interface or the Developer Workbench tool.

To summarize the different techniques:

- Join defined at DB level with in SQL view + metadata on view
- Join defined in the DB2 Web Query metadata layer automatically enforcing Referential Integrity rules
- Join defined in metadata using:
  - Web metadata editor:
    - Reference to existing synonym
    - Segment manually

  - Developer Workbench tool:
    - Reference to existing synonym
    - Copy of existing synonym
    - Segment via metadata import
    - Segment manually

The various approaches, with relative advantages and disadvantages, are described in the following sections.

Notes:

- One of the main concerns people usually have with creating complex metadata with multiple join is about performance. This arises from the idea that access can be slow when joining multiple database object.
- Metadata is an abstraction layer, not the physical data.
- If in the report are included columns from just one segment, the subsequent SQL statement that is generated by DB2 Web Query will only reference that single object, even if the metadata used references many tables with their join relationship.
- Think of these join as "potential" joins that only become effective when they are really needed in reports.
- On the other hand, when a join is needed, it is already defined and ready to be used by everyone who develops reports.

3.6.1 Referential integrity

In this section, we explain the benefits to DB2 Web Query of having referential integrity defined in the database. We begin by briefly explaining referential integrity.

A feature of a relational database is that it must support the definition and enforcement of referential integrity. If you do not fully grasp the meaning of referential integrity, you are not alone.

A database can consist of one large fact table and several smaller dimension tables. The dimension tables each contain a primary key that is referenced by a foreign key in the fact table. For our example, the primary key on the inventory table is PROD_NUM, and the corresponding foreign key on the order table is PROD_NUM. An entry in the fact table is not allowed unless there is a matching key in the dimension tables. That is, you are not allowed to place an order for a product that is not confirmed to be in your inventory. The referential constraint does not permit an entry into the orders table unless the entry has a value for PROD-NUM in the inventory table. This prohibits unmatched, or orphan, entries in the Orders table.

Tip: You can also think of the small dimension tables as *parent* tables, and the large fact table as a *child* table. Do not let the part about the child table being larger confuse you. Think of it as a parent that has many children. The constraint does not let the child be born unless the parent already exists.

Benefit of using referential integrity in DB2 Web Query

When DB2 Web Query creates metadata, it recognizes files that are related through referential integrity and brings them all in together in one piece of metadata. When you write your report, you select a single synonym or table name that includes several files that are all related to each other. You do not need to manually define all the table joins. In order to have this convenience, you must have defined constraints on your tables.

If you do not currently have referential integrity in your database, we strongly recommend that you consider implementing it. Not only does referential integrity aid in more effective report writing in DB2 Web Query, but it also moves your business logic closer to the database level to reduce the programming effort. You do not need to understand SQL to add constraints to your database. You can do this easily by using iSeries Navigator. Referential integrity constraints can be added to files that are created with DDS and CRTPF, or they can be added to tables that are created with the CREATE TABLE statement.

Tip: If you want to add constraints by using the command line, use the ADDPFCST command. If you want to add constraints by using SQL, use the CREATE TABLE or ALTER TABLE statement.

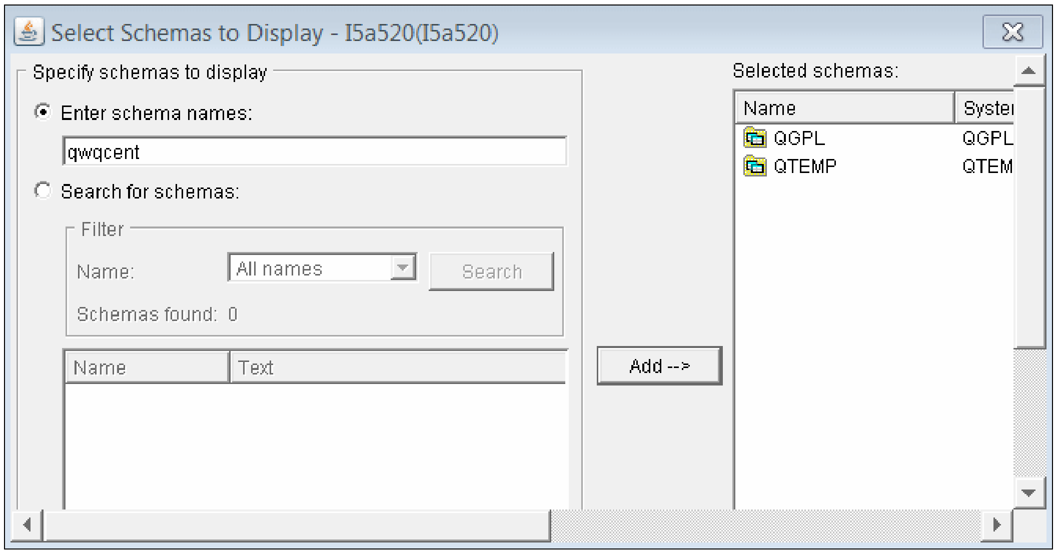
The following brief example demonstrates how to set up referential integrity on our sample QWQCENT database. The orders table is the fact table. The inventory, plant, and stores tables are the dimensional tables. For our example, we set up referential integrity between the orders table and the inventory table. An item must exist in the inventory table before it can have a child in the orders table.

First we must create a primary key in INVENTORY, then we can create the foreign key in ORDERS. The relationship between the two is called a *referential constraint*. The referential constraint starts at the child table and ensures that no child entry is inserted into the table unless an entry exists in the parent table.

Creating a primary key on INVENTORY field PROD_NUM

Open iSeries Navigator and follow these steps:

1. Expand the database container.
2. If the QWQCENT schema is not currently displayed, right-click **Schemas** and then **Select Database to display**.
3. In the Select Schemas to Display window (Figure 3-30), in the Enter schema names field, type `Century` and click **Add**. Then click **OK**.


Select Schemas to Display - I5a520(I5a520)

Specify schemas to display

(● Enter schema names: qwqcent

○ Search for schemas:

Filter

Name: All names

Schemas found: 0

Name	Text

Selected schemas:

Name	Syster
QGPL	QGPL
QTEMP	QTEM

Figure 3-30 Select Schemas to Display window

4. In the left navigation bar of iSeries Navigator, expand the **QWQCENT** schema and click **Tables**. In the right pane, right-click the **INVENTORY** table and select **Definition**, as shown in Figure 3-31.

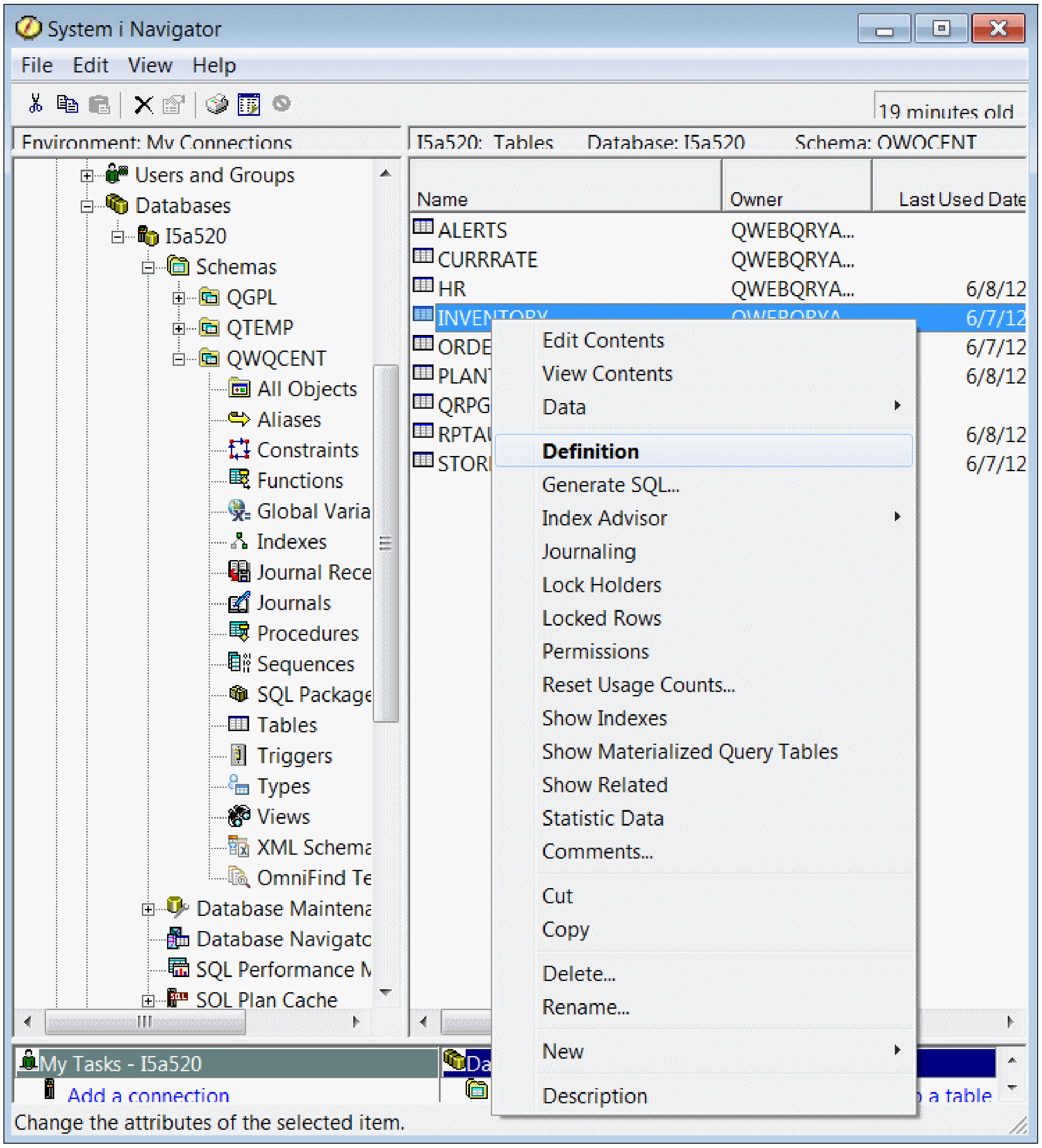
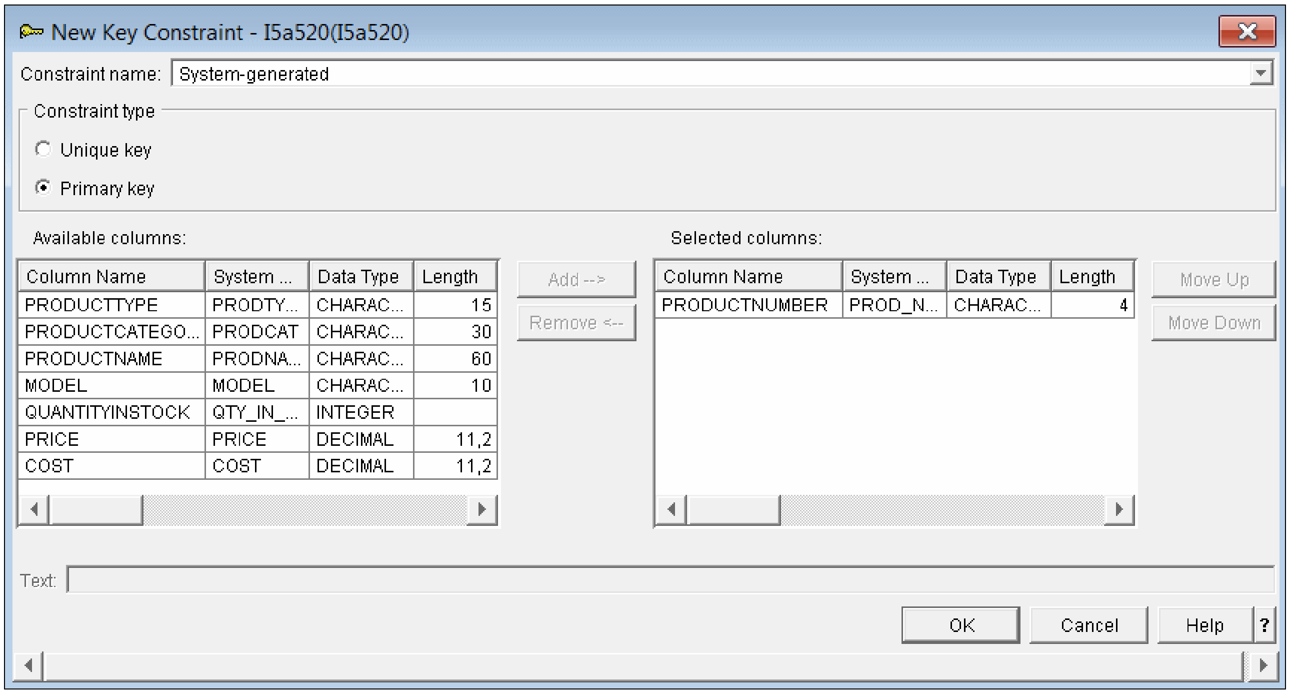


Figure 3-31 Selecting the Definition Option

5. In the next window, from the tabbed selections, click the **Key Constraints** tab and then click **Add**.

6. In the New Key Constraint window (Figure 3-32), complete these steps:
   d. Under Constraint type, select the **Primary Key** radio button.
   e. Under Available columns, select the **ProductNumber** field and click the **Add** arrow button. Now the ProductNumber field is displayed under Selected columns. Click **OK**.



Unique key [✓ Primary key]

Available columns:

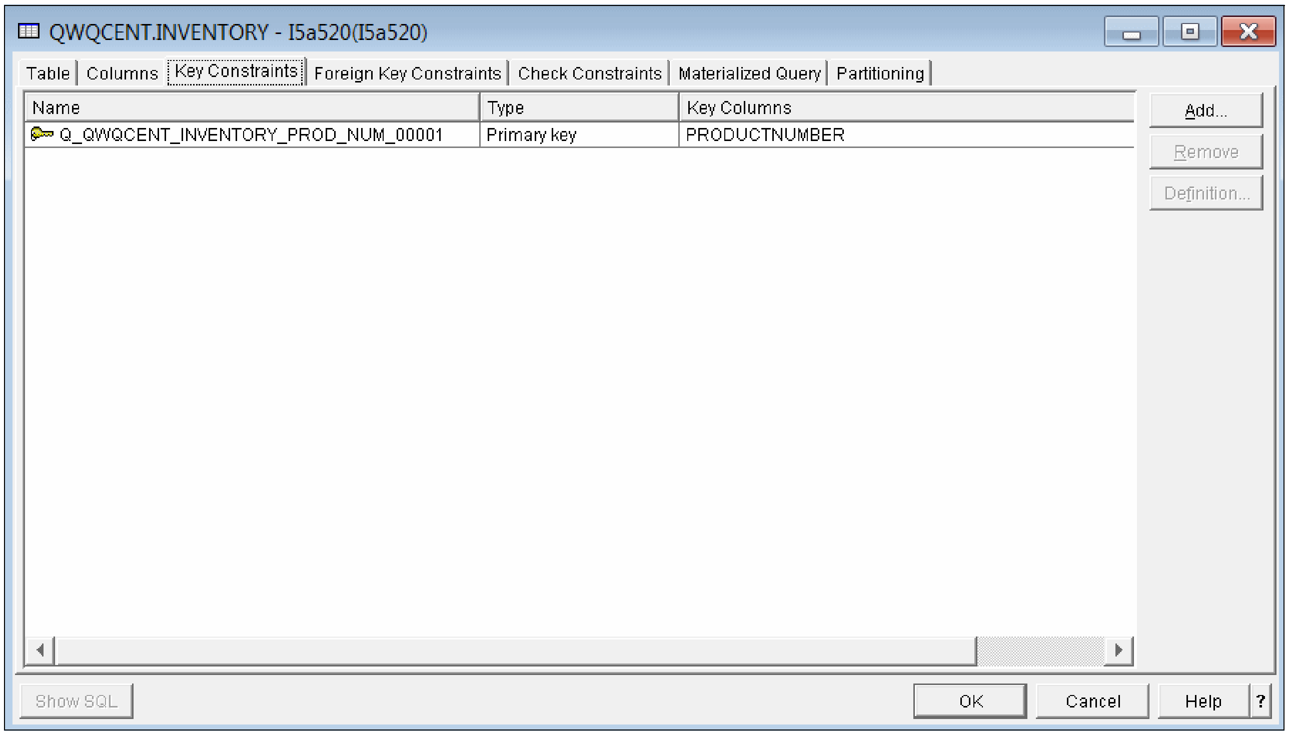
Column Name	System ...	Data Type	Length
PRODUCTTYPE	PRODTY...	CHARAC...	15
PRODUCTCATEGO...	PRODCAT	CHARAC...	30
PRODUCTNAME	PRODNA...	CHARAC...	60
MODEL	MODEL	CHARAC...	10
QUANTITYINSTOCK	QTY_IN_...	INTEGER	
PRICE	PRICE	DECIMAL	11,2
COST	COST	DECIMAL	11,2

Selected columns:

Column Name	System ...	Data Type	Length
PRODUCTNUMBER	PROD_N...	CHARAC...	4

Figure 3-32 Key Constraints window

7. In the window that contains the various tabs (Figure 3-33), click **OK**.

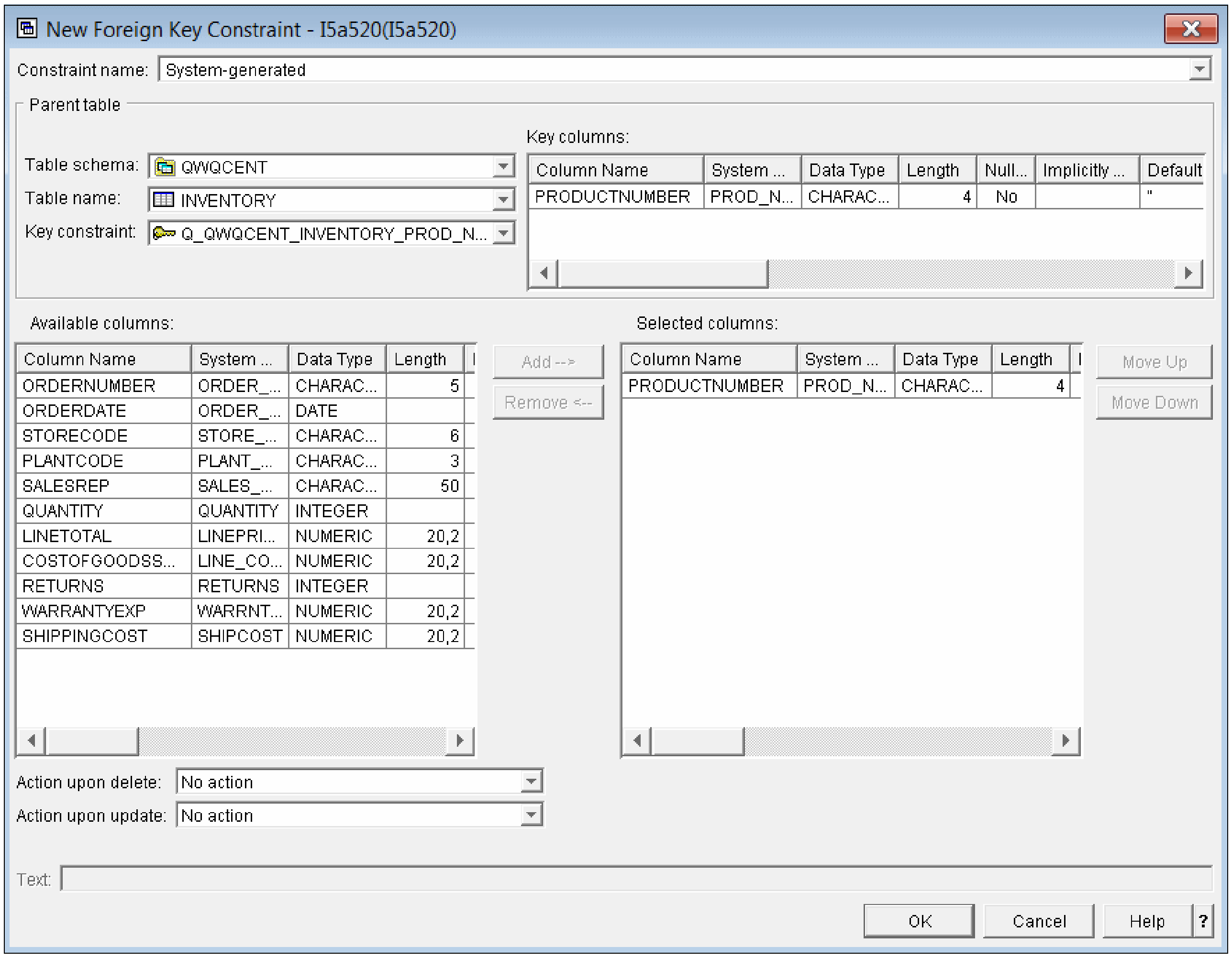


Name	Type	Key Columns
Q_QWQCENT_INVENTORY_PROD_NUM_00001	Primary key	PRODUCTNUMBER

Figure 3-33 Key Constraints tab showing the ProductNumber field

You are now finished creating the primary constraint on the Inventory table. Next, we go to the orders table and create a referential constraint. Remember that the orders table checks the inventory table to ensure that any new record also has a matching key value in Inventory:

1. Expand the **QWQCENT** schema and click **Tables**. Right-click the **Orders** table and select **Definition**.
2. In the next window, click the **Foreign Key Constraints** tab. Click **Add**.
3. In the New Foreign Key Constraint window (Figure 3-34) complete these steps:
   a. For table name, select **Inventory**.
   b. Under Available Columns, select **ProductNumber** and click **Add**. We know that this is the primary key on the inventory table because it is indicated under Key columns near the top of the window on the right. Click **OK**.


New Foreign Key Constraint - I5a520(I5a520)

Constraint name: System-generated

Parent table

Table schema: QWQCENT

Table name: INVENTORY

Key constraint: Q_QWQCENT_INVENTORY_PROD_N...

Key columns:

Column Name	System ...	Data Type	Length	Null...	Implicitly ...	Default
PRODUCTNUMBER	PROD_N...	CHARAC...	4	No		''

Available columns:

Column Name	System ...	Data Type	Length
ORDERNUMBER	ORDER_...	CHARAC...	5
ORDERDATE	ORDER_...	DATE	
STORECODE	STORE_...	CHARAC...	6
PLANTCODE	PLANT_...	CHARAC...	3
SALESREP	SALES_...	CHARAC...	50
QUANTITY	QUANTITY	INTEGER	
LINETOTAL	LINEPRI...	NUMERIC	20,2
COSTOFGOODSS...	LINE_CO...	NUMERIC	20,2
RETURNS	RETURNS	INTEGER	
WARRANTYEXP	WARRNT...	NUMERIC	20,2
SHIPPINGCOST	SHIPCOST	NUMERIC	20,2

Selected columns:

Column Name	System ...	Data Type	Length
PRODUCTNUMBER	PROD_N...	CHARAC...	4

Action upon delete: No action

Action upon update: No action

Text:

Figure 3-34 New Foreign Key Constraint window

You are now finished creating the referential constraint between the inventory and orders tables. You can now repeat this process with the two other dimension tables, stores and plant. The key fields on those tables are, respectively, Store_Code and Plant_Code. The orders table has the corresponding foreign key for each table as it did for the inventory table.

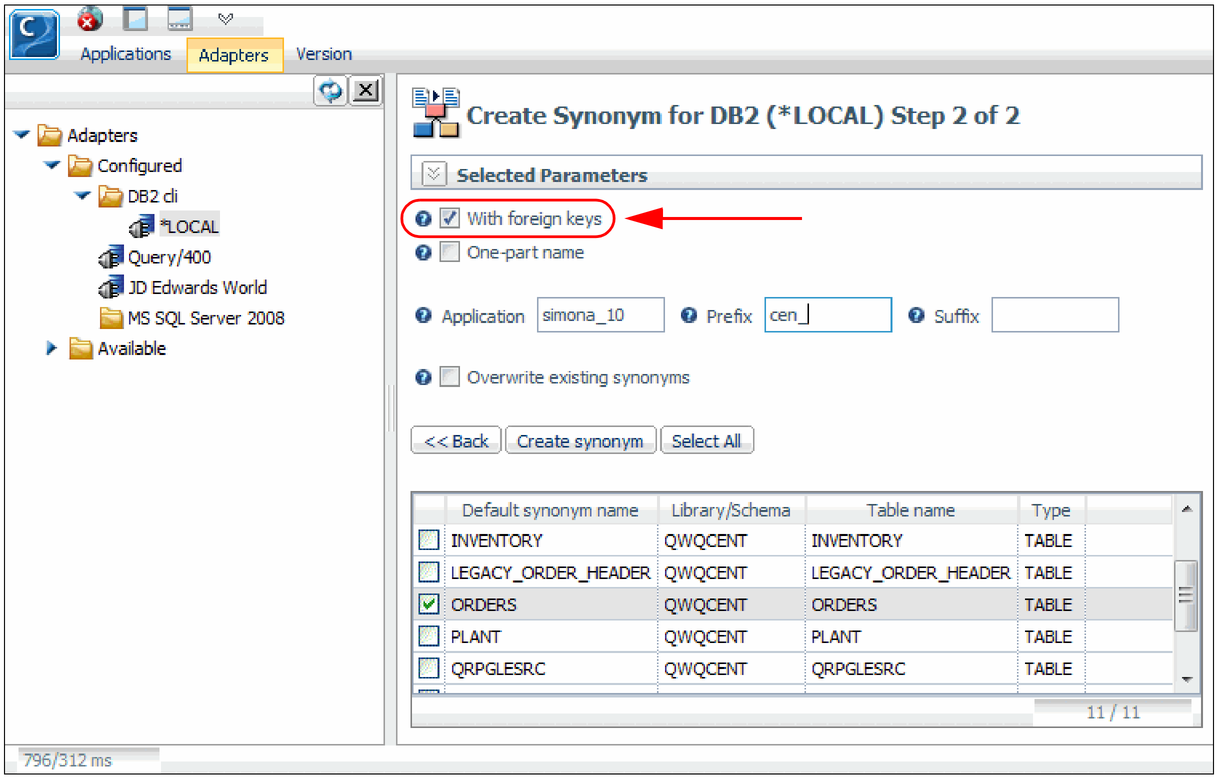
After you are done creating the other two referential constraints, your database is now ready for metadata creation.

3.6.2 Creating metadata on tables with referential integrity

The process to import a *cluster* of related tables into one piece of metadata is similar to creating a single piece of metadata. Refer to Chapter 3.2.1, “Creating metadata with the web interface” on page 26 for detailed instructions.

1. In the Create Synonym pane (Figure 35), notice the “With foreign keys” check box at the top of the page. This is the only difference in the process. You must check the **With foreign keys** check box, so that the related tables are included in the same piece of metadata.

   Then the only table that you have to select is the fact table, which in our example library is **ORDERS**. The dimension tables are automatically included. Then click **Create synonym**.



[✓] With foreign keys
[ ] One-part name

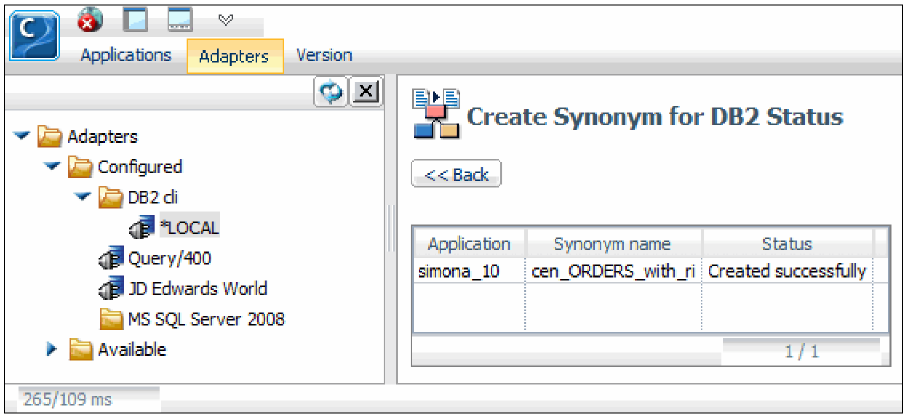
Application: simona_10 Prefix: cen_ Suffix:

[ ] Overwrite existing synonyms

	Default synonym name	Library/Schema	Table name	Type
☐	INVENTORY	QWQCENT	INVENTORY	TABLE
☐	LEGACY_ORDER_HEADER	QWQCENT	LEGACY_ORDER_HEADER	TABLE
✓	ORDERS	QWQCENT	ORDERS	TABLE
☐	PLANT	QWQCENT	PLANT	TABLE
☐	QRPGLESRC	QWQCENT	QRPGLESRC	TABLE

Figure 35 Creating metadata with referential integrity

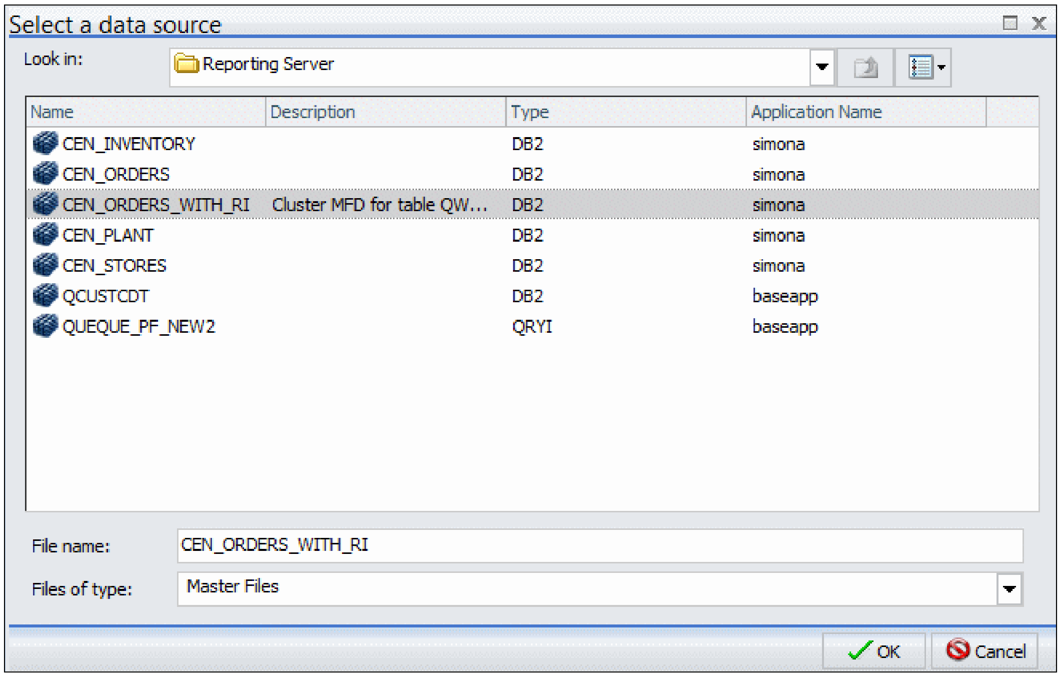
2. Next, the Status pane (Figure 3-36) is displayed, on which you see the status message, `Created successfully`. Notice that there is only one piece of metadata. You can close this panel by selecting **Finish**.



Application	Synonym name	Status
simona_10	cen_ORDERS_with_ri	Created successfully

Figure 3-36 Created successfully message

As we have already mentioned, this one synonym includes references to all of the tables connected by referential integrity relationship, specifying the join relationships. A synonym that includes joins can be referred to as a *clustered synonym*. When you first start to write the report based on the file cluster, you are presented with the metadata window: the related files are all under one name. The description says `Cluster xxx for table yourFactTable`, as in Figure 3-37. The word `Cluster` in your metadata indicates that referential integrity has been used to create that metadata. All the related dimension tables with referential constraints defined appear in the development tool, Info Assist.



Name	Description	Type	Application Name
CEN_INVENTORY		DB2	simona
CEN_ORDERS		DB2	simona
CEN_ORDERS_WITH_RI	Cluster MFD for table QW...	DB2	simona
CEN_PLANT		DB2	simona
CEN_STORES		DB2	simona
QCUSTCDT		DB2	baseapp
QUEQUE_PF_NEW2		QRYI	baseapp

File name: CEN_ORDERS_WITH_RI

Files of type: Master Files

Figure 3-37 Cluster: Metadata with referential integrity

3.6.3 Creating metadata over tables without referential integrity

While there are many advantages to implementing referential integrity, many existing DB2 for IBM i production databases currently do not have RI set up. This means that some activities have to be performed to define the relationship to set up the join between objects. The following three options are viable if tables must be joined together:

- Define join in SQL views
- Define join in the DB2 Web Query metadata layer
- Define join in the reports and graphs

Each of the above options is discussed in the following sections.

3.6.4 Defining joins in SQL views

An SQL view is a virtual table whose definition is based on a SELECT statement. In traditional i5/OS terms, they can be thought of as non-keyed logical files. Because there are no keys associated with views, there is no access path maintenance. This means that you can create as many views as you like and not worry about performance repercussions associated with access plan maintenance.

SQL views are can be used to join tables together, present only some columns, define derived columns, aggregate data and so on. This way, report development can be simplified and only desired data are presented for reporting, masquerading the database complexity and/or avoiding exposing sensitive data when unnecessary. Once the view is created, DB2 Web Query metadata can be created over the view (instead of the base tables). Reports using that synonym can access any of the columns from each of the join segments defined in the view. Using reports based on SQL views provides several distinct advantages to DB2 Web Query, including these:

- Encourages database optimization (*pushes* logic down to DB2)
- Provides a method for implementing more complex reporting requirements such as row and column-level security
- Provides additional join types, unions, intersects, excepts, Common Table Expressions, complex business logic (case statements, and so on)
- Is data-centric because business logic and rules can be defined in one place (at the database level) rather than the application level

In addition, SQL views can be accessed by other interfaces such as RPG programs (with embedded SQL or native Record Level Access), ODBC, JDBC, and so on.

In Example 3-1 we illustrate an SQL View that could be leveraged by DB2 Web Query.

Example 3-1 SQL View definition

```
CREATE VIEW ORDERSVIEW
  (ORDERNUMBER FOR COLUMN ORDER_NUM,
   PRODUCTNUMBER FOR COLUMN PROD_NUM,...)
AS SELECT a.*, b.*, c.*, d.* FROM orders a
INNER JOIN stores b
   ON a.storecode = b.storecode
INNER JOIN inventory c
   ON a.prod_num = c.prod_num
INNER JOIN plant d
   ON a.plantcode = d.plantcode
```

A synonym created over the example SQL view would contain all of the columns in the joined tables.

You create a synonym over an SQL view in the same way as you do with tables. See 3.2, “Creating metadata” on page 25 for detailed instructions.

3.6.5 Defining joins in DB2 Web Query synonyms

You can also define the relationship among files in the DB2 Web Query metadata. When this technique is employed, DB2 Web Query join syntax is stored in metadata and translated to an SQL statement with join syntax. Because the joins are defined at the synonym level, you only must do this in one place. Every report that uses this synonym will have access to the join logic already defined, thus it is not necessary to define joins in each report. This is an example of the power of the metadata abstraction layer.

Consider the following possibilities when using this technique:

- It can be accessed by all DB2 Web Query reports and graphs that use the synonym. It is not accessible from any other SQL interface.
- Simply defining the join in a metadata does not force a join at execution time if columns from the other table(s) are not used in the report.
- Performance could suffer if logic is not pushed down to database due to database optimization disablers. For more information about this, see 23.5.2, “Performing analysis and looking for optimization disablers” on page 753.

Before you begin this exercise, you need to have a basic understanding of the different types of joins that DB2 Web Query supports:

- Multiple (SEGTYPE=S0):

  Indicates that the segment has no key field and is therefore not sorted. Keys and sequences are defined in the access file. With the multiple segment type, each joined file is represented as an individual join segment. If no matching row is found based on the join criteria, no row is returned to the join. This prevents something called the multiplicative effect, but also does not allow a developer to write a report that sorts (or performs a combination of sorting and selecting) across the different segments.
- Unique (SEGTYPE=U):

  For each join definition. Every child segment becomes logically part of the parent. At least one row will be retrieved, regardless of whether a matching row was found. If there is a one-to-many relationship between parent and child, *all* children will be returned (despite the name *Unique*, which indicates a one-to-one relationship). Since the child becomes an extension of the parent, the result is one virtual segment. This provides the ability to sort and select fields across the underlying segments, but could also result in a report that delivers incorrect results due to the multiplicative effect.

For this exercise, we are not concerned about the multiplicative effect. Therefore, unique join types are defined.

Another very important concept is on how synonyms are “linked” one into the other. Basically four different approaches are available:

- *Reference to existing synonym* means that the content of the second (“joined”) metadata is just referenced in the main one (“joiner”). The “joined” synonym must still exist and can be updated/refreshed. These changes are reflected in the joiner (are or can? requires manual refresh?).

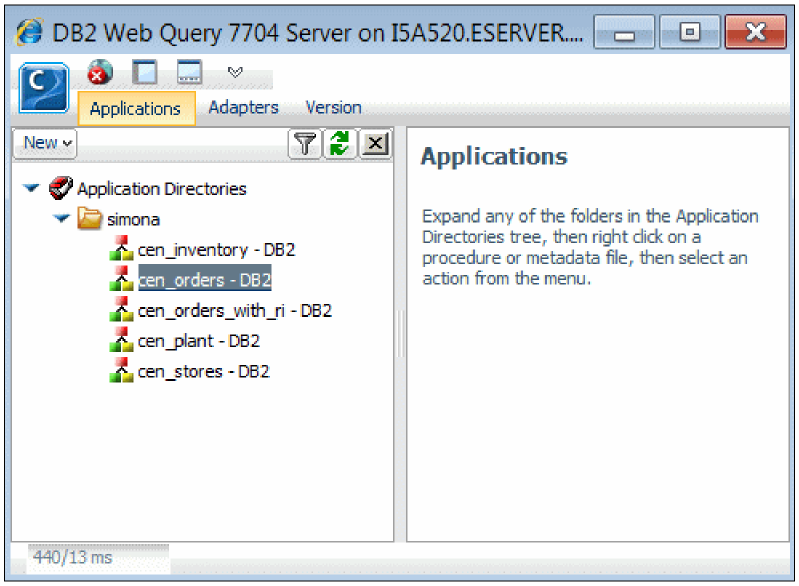
- *Copy of existing synonym* means that the content of the second ("joined") metadata is copied in the main one ("joiner"). The "joined" synonym can be deleted if it is not used stand alone. If the "joined" synonym is updated/refreshed these changes are not reflected in the joiner.
- *Add segment via metadata import* enables you to add reference to a new object using the Create Synonym tool. This tool includes in the existing synonym the reference to another object and describes its content. The result is just one synonym definition, the pre-existing one, that includes all definition needed to access multiple objects.
- *Add segment manually* implies coding the reference manually, that is, describing not only the location of the joined object and the join relationship, but also all columns that are in the objects. This is a very cumbersome task and highly error prone, hence it is not recommended.

All of the various approaches described are available when using the Developer Workbench tool, while when using the web interface, only *Reference to existing synonym* and *Add segment manually* are available.

Defining joins in metadata using the web interface

Here we document how to define a join in the web interface, using the Reference to existing synonym capability. Notice that with this technique all of the synonyms must exist and have to be maintained. The result can be a proliferation of synonyms that make maintenance more difficult. In our example, we will be joining together some synonyms we have already created, CEN_ORDERS, CEN_INVENTORY, CEN_PLANT and CEN_STORES:

1. In your folder select **Metadata** → **Edit**.
2. Double-click the metadata you want to enrich with a join, **cen_orders** in our example (Figure 3-38).



Applications

Expand any of the folders in the Application Directories tree, then right click on a procedure or metadata file, then select an action from the menu.

Figure 3-38 Edit in Synonym Editor

The cen_orders master file is opened in the synonym editor.

3. Right-click the **CEN_ORDERS** segment and select **Insert** → **Reference to Existing Synonym**, as shown in Figure 3-39.

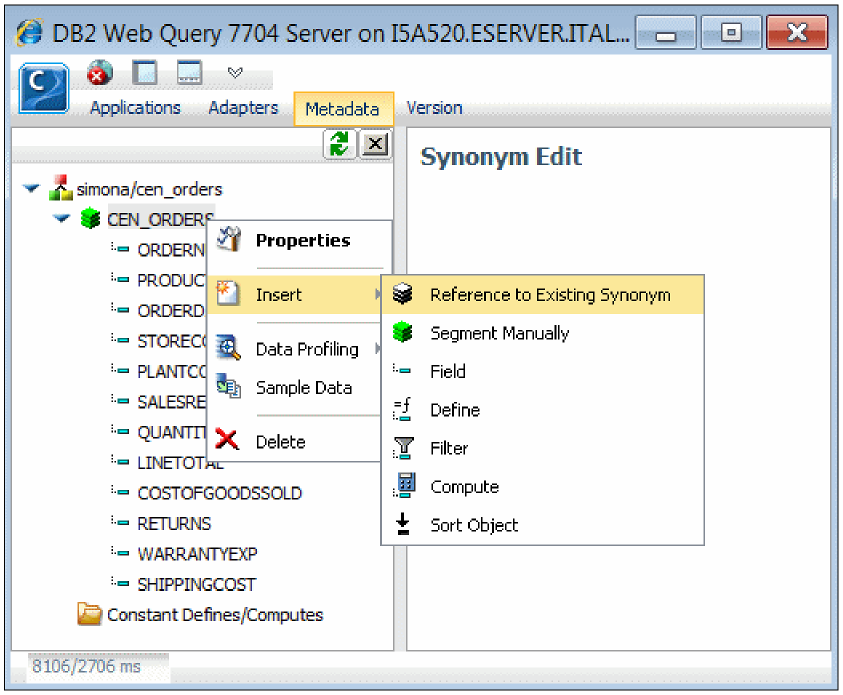
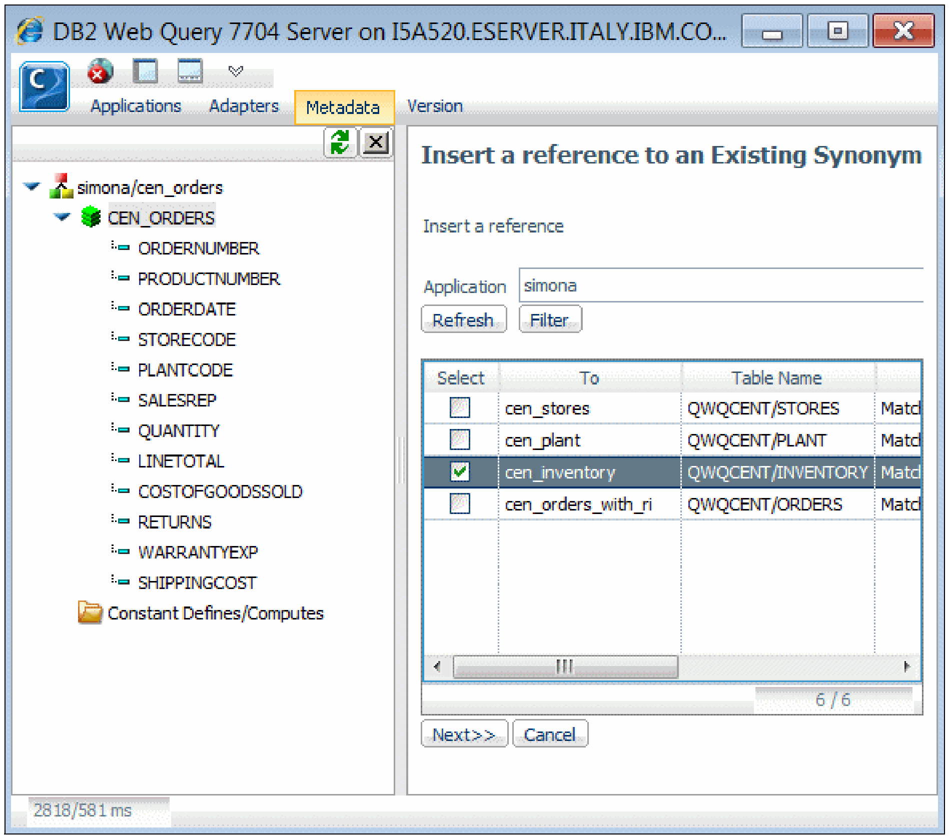


Figure 3-39 Add join segment to synonym

The list of existing synonyms is presented.

4. From the presented list, select the **cen_inventory** synonym, as shown in Figure 3-40, and click the **Next** button.


DB2 Web Query 7704 Server on I5A520.ESERVER.ITALY.IBM.CO...
Applications
Adapters
Metadata
Version
Insert a reference to an Existing Synonym
Insert a reference
Refresh
Filter
Next>>
Cancel
6 / 6
2818/581 ms

simona/cen_orders
CEN_ORDERS
ORDERNUMBER
PRODUCTNUMBER
ORDERDATE
STORECODE
PLANTCODE
SALESREP
QUANTITY
LINETOTAL
COSTOFGOODSSOLD
RETURNS
WARRANTYEXP
SHIPPINGCOST
Constant Defines/Computes

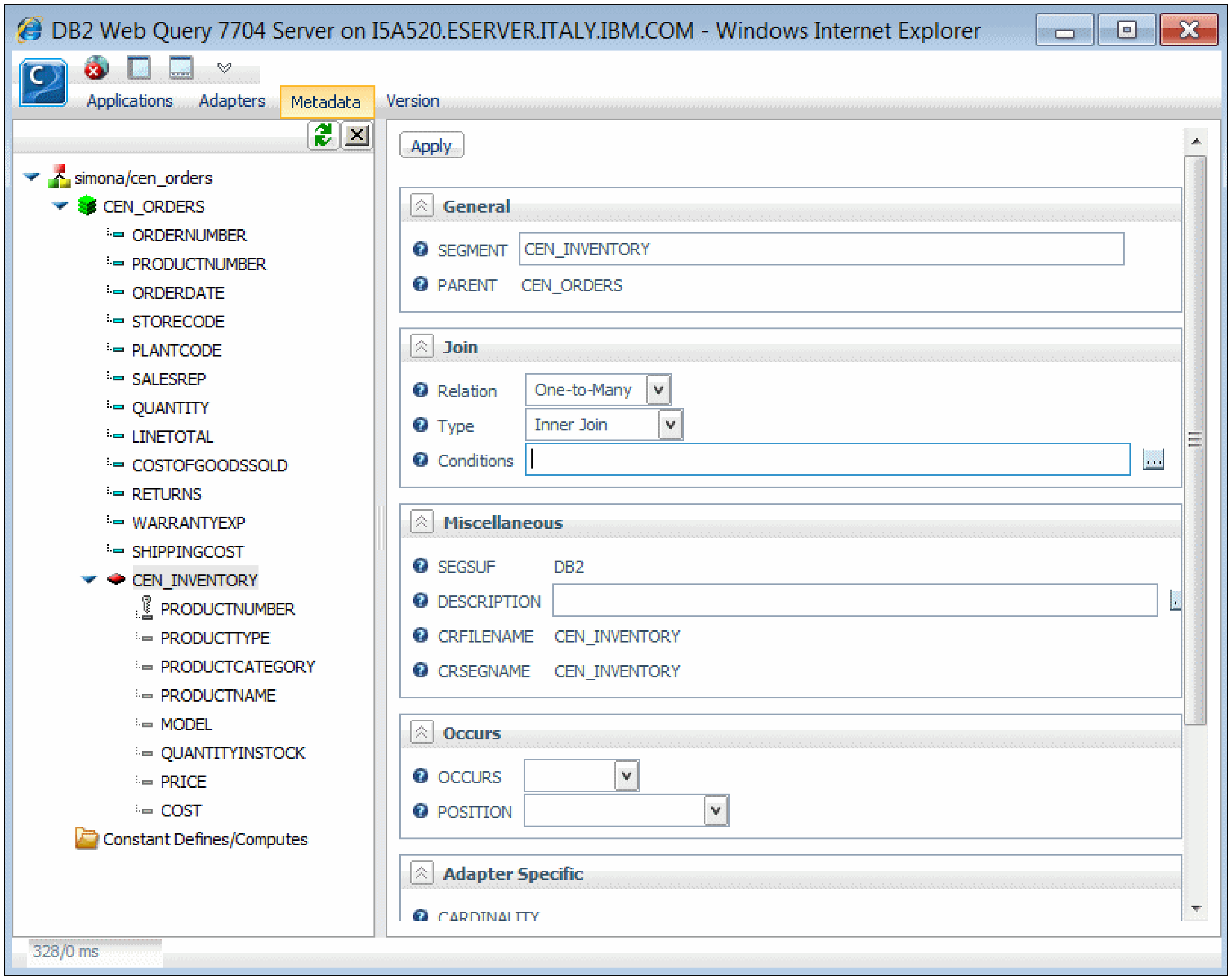
Application: simona

Select	To	Table Name	
☐	cen_stores	QWQCENT/STORES	Matcl
☐	cen_plant	QWQCENT/PLANT	Matcl
✓	cen_inventory	QWQCENT/INVENTORY	Matcl
☐	cen_orders_with_ri	QWQCENT/ORDERS	Matcl

Figure 3-40 Select synonym to add

The segment CEN_INVENTORY is added under CEN_ORDERS.

5. Select the **CEN_INVENTORY** segment and, in the right-hand part of the pane, set **relation**, **type** and **conditions** of the join (Figure 3-41).


DB2 Web Query 7704 Server on I5A520.ESERVER.ITALY.IBM.COM - Windows Internet Explorer
Applications
Adapters
Metadata
Version
Apply
328/0 ms

simona/cen_orders
- CEN_ORDERS
  - ORDERNUMBER
  - PRODUCTNUMBER
  - ORDERDATE
  - STORECODE
  - PLANTCODE
  - SALESREP
  - QUANTITY
  - LINETOTAL
  - COSTOFGOODSSOLD
  - RETURNS
  - WARRANTYEXP
  - SHIPPINGCOST
  - CEN_INVENTORY
    - PRODUCTNUMBER
    - PRODUCTTYPE
    - PRODUCTCATEGORY
    - PRODUCTNAME
    - MODEL
    - QUANTITYINSTOCK
    - PRICE
    - COST
- Constant Defines/Computes

General

SEGMENT	CEN_INVENTORY
PARENT	CEN_ORDERS

Join

Relation	One-to-Many
Type	Inner Join
Conditions	

Miscellaneous

SEGSUF	DB2
DESCRIPTION	
CRFILENAME	CEN_INVENTORY
CRSEGNAME	CEN_INVENTORY

Occurs

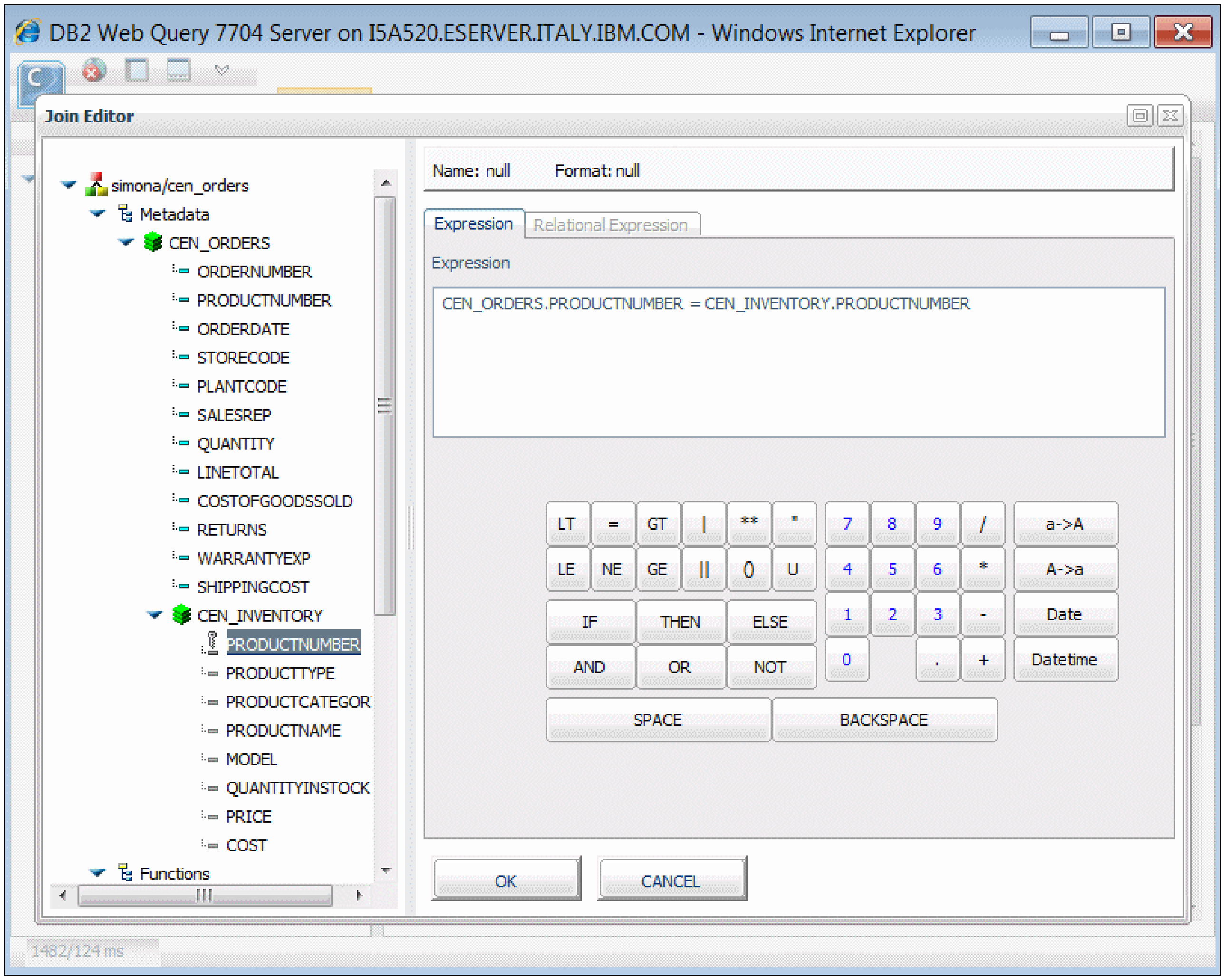
OCCURS	
POSITION	

Adapter Specific

CARDINALITY

Figure 3-41 Specify join properties

6. To help with defining the “conditions,” start the “calculator” using the little dotted button on the right of the field to set the join. PRODUCTNUMBER is the field that is used to join ORDERS to INVENTORY. Select this field on the ORDERS table; it gets inserted in the calculator panel. Select the = symbol and then select the PRODUCTNUMBER filed in the INVENTORY table. When you are done, select **OK** (Figure 3-42).

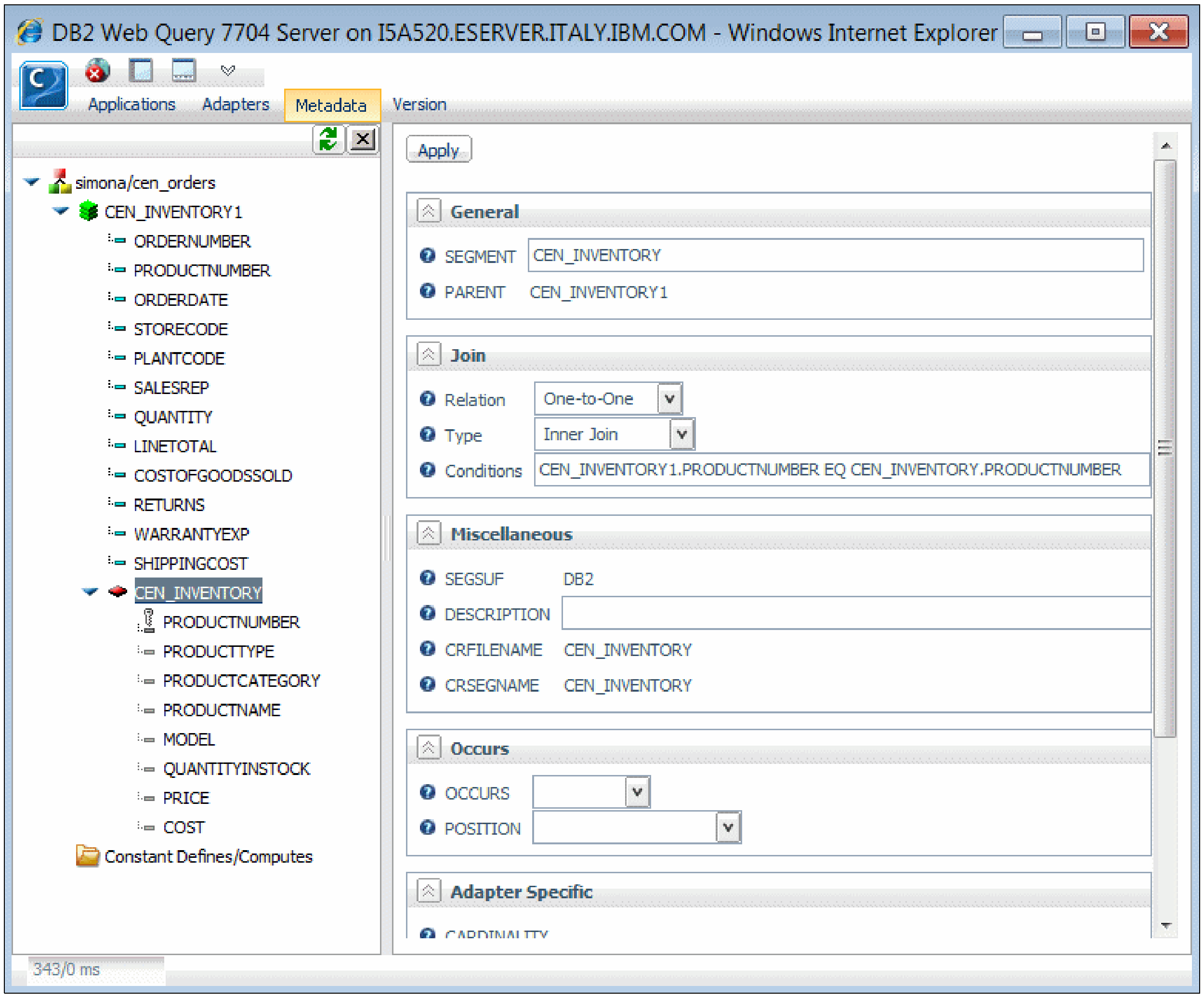


CEN_ORDERS.PRODUCTNUMBER = CEN_INVENTORY.PRODUCTNUMBER

Figure 3-42 Specify join condition

Note: Once you have set the relation to one-to-one, the icon next to the CEN_INVENTORY segment becomes a red color.

7. Once the relationship is defined, click **Apply**, at the top on the pane (Figure 3-43).


DB2 Web Query 7704 Server on I5A520.ESERVER.ITALY.IBM.COM - Windows Internet Explorer
Applications
Adapters
Metadata
Version
Apply
343/0 ms

simona/cen_orders
- CEN_INVENTORY1
  - ORDERNUMBER
  - PRODUCTNUMBER
  - ORDERDATE
  - STORECODE
  - PLANTCODE
  - SALESREP
  - QUANTITY
  - LINETOTAL
  - COSTOFGOODSSOLD
  - RETURNS
  - WARRANTYEXP
  - SHIPPINGCOST
  - CEN_INVENTORY
    - PRODUCTNUMBER
    - PRODUCTTYPE
    - PRODUCTCATEGORY
    - PRODUCTNAME
    - MODEL
    - QUANTITYINSTOCK
    - PRICE
    - COST
- Constant Defines/Computes

General

SEGMENT CEN_INVENTORY
PARENT CEN_INVENTORY1

Join

Relation One-to-One
Type Inner Join
Conditions CEN_INVENTORY1.PRODUCTNUMBER EQ CEN_INVENTORY.PRODUCTNUMBER

Miscellaneous

SEGSUF DB2
DESCRIPTION
CRFILENAME CEN_INVENTORY
CRSEGNAME CEN_INVENTORY

Occurs

OCCURS
POSITION

Adapter Specific

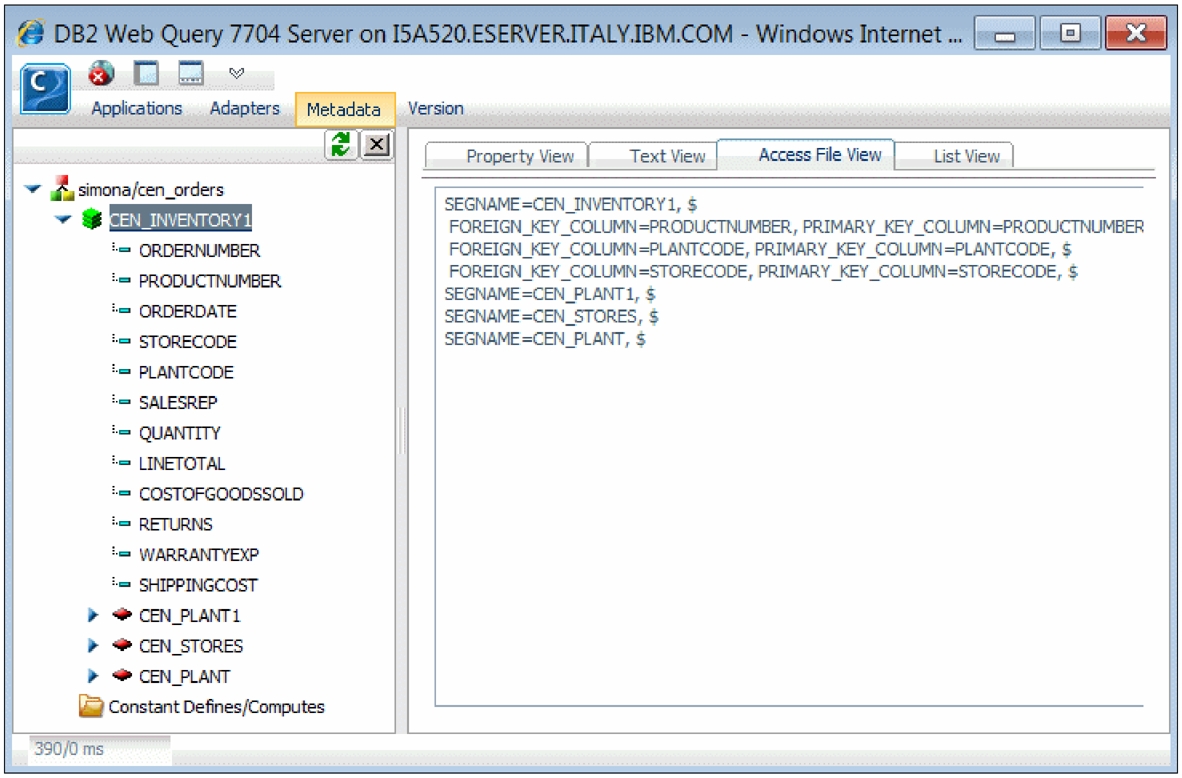
CARDINALITY

Figure 3-43 Join properties

8. Repeat step 3 on page 61 to step 7 for the CEN_STORES segment. The join field for this join is STORECODE.

9. Repeat step 3 on page 61 to step 7 for the CEN_PLANT segment. The join field for this join is PLANTCODE.

10.At this point you have created a *cluster* for the ORDERS table. Notice that the access file has been updated to mention all the tables that have been described to be connected. The segments of this cluster should look like the example shown in Figure 3-44.



```
SEGNAME=CEN_INVENTORY1, $
 FOREIGN_KEY_COLUMN=PRODUCTNUMBER, PRIMARY_KEY_COLUMN=PRODUCTNUMBER
 FOREIGN_KEY_COLUMN=PLANTCODE, PRIMARY_KEY_COLUMN=PLANTCODE, $
 FOREIGN_KEY_COLUMN=STORECODE, PRIMARY_KEY_COLUMN=STORECODE, $
SEGNAME=CEN_PLANT1, $
SEGNAME=CEN_STORES, $
SEGNAME=CEN_PLANT, $
```

Figure 3-44 CEN_ORDERS with all join segments

11.Save your work by selecting the control icon on the top left-hand corner of the pane and selecting **Save**, as shown in Figure 3-45.

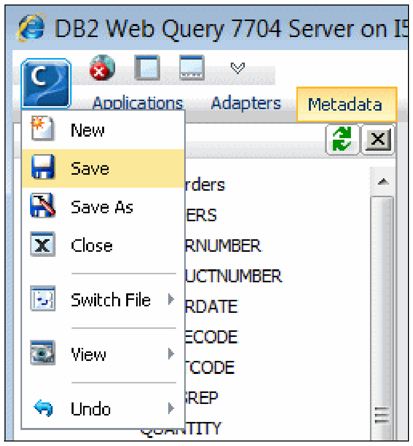


Figure 3-45 Save synonym

Note: Creating a cluster synonym has no negative effect when querying data. The SQL statement is composed to reference ONLY the tables whose columns are queried, not all the tables that are defined in the synonym.

Defining joins in metadata with Developer Workbench

Developer Workbench is described in 16.5, “DB2 Web Query Developer Workbench” on page 500.

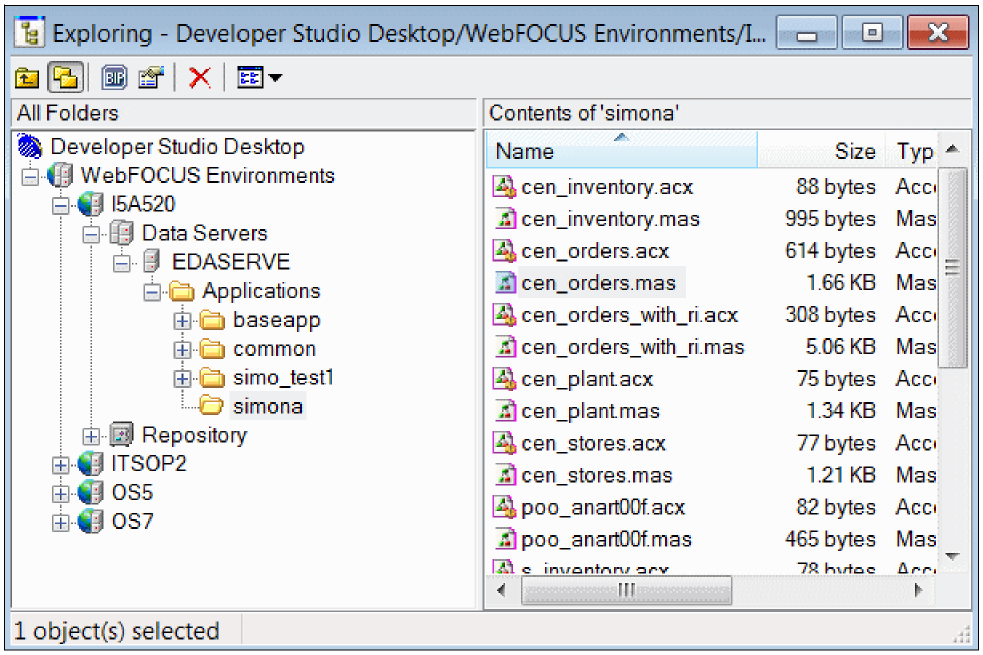
It gives a very powerful interface to interact with metadata, in a more graphical way than with the web interface included in the base product and with more options and functions.

The differences among the different join options have already been discussed in 3.6.5, “Defining joins in DB2 Web Query synonyms” on page 59. Here we document how to add a join using two different techniques, *copy of existing synonym* and *segment via metadata import.* These techniques were only available with Developer Workbench at the time this book was written. Availability may change over time.

Join - copy of existing synonym

Follow these steps:

1. Open Developer Workbench, select the system on which to work, and validate with your userid and password. Remember that the user must be part of the DevWorkbench group (use **Administration** → **Security Center** in the web interface to set up). Select **Data Servers** → **EDASERVE** → **Applications** → ***<yourfolder>*** as shown in Figure 3-46.



Name	Size	Typ
cen_inventory.acx	88 bytes	Acc
cen_inventory.mas	995 bytes	Mas
cen_orders.acx	614 bytes	Acc
cen_orders.mas	1.66 KB	Mas
cen_orders_with_ri.acx	308 bytes	Acc
cen_orders_with_ri.mas	5.06 KB	Mas
cen_plant.acx	75 bytes	Acc
cen_plant.mas	1.34 KB	Mas
cen_stores.acx	77 bytes	Acc
cen_stores.mas	1.21 KB	Mas
poo_anart00f.acx	82 bytes	Acc
poo_anart00f.mas	465 bytes	Mas
s_inventory.acx	78 bytes	Acc

Figure 3-46 Join in Developer Workbench - Step 1

2. Double-click the synonym you want to enrich with join relationships to open it. In our example, we will change CEN_ORDERS to add joins with the INVENTORY, PLANT, and STORES tables. Since we will be using “copy of existing synonym” the synonyms for these tables must already exist.

3. Right-click the main segment name, CEN_ORDERS in our example, and select **Insert** → **Copy of Existing Synonym** (Figure 3-47).

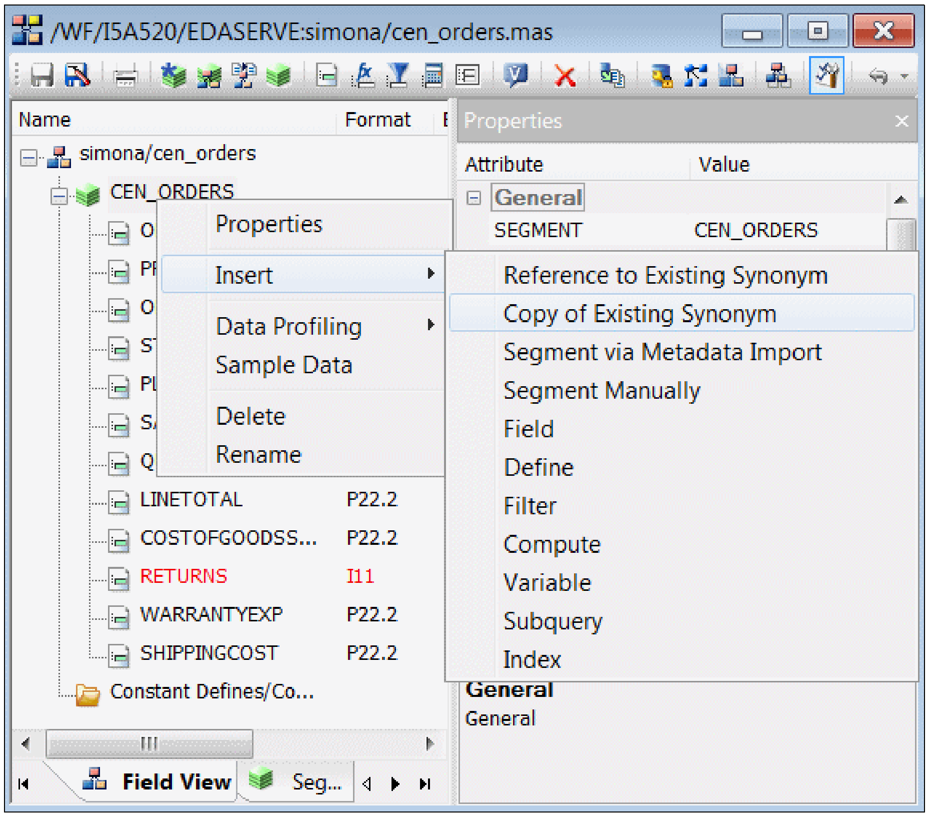
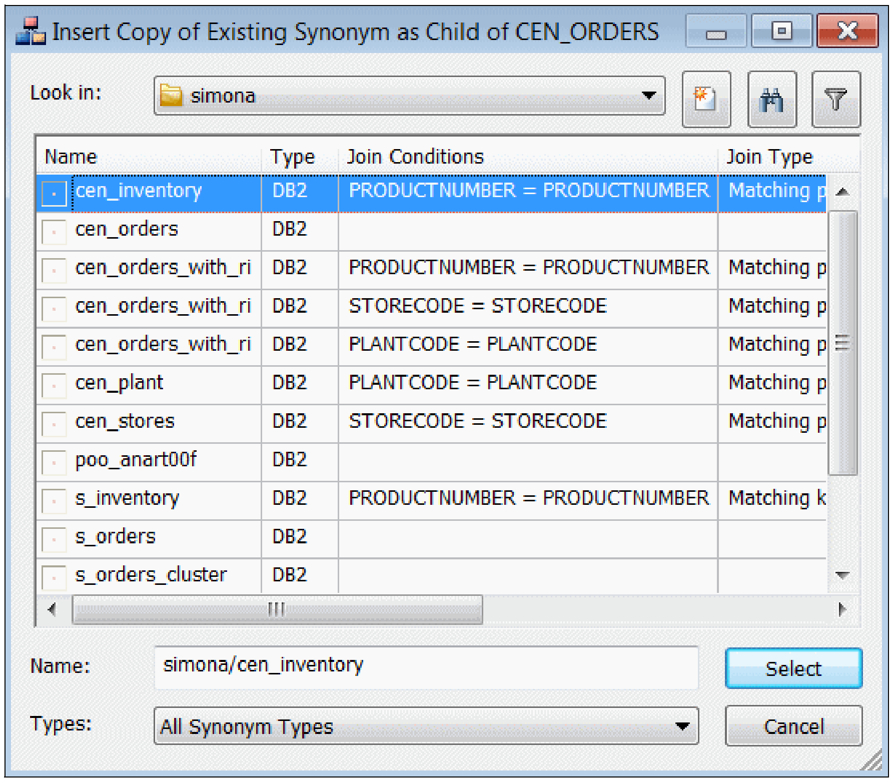


Figure 3-47 Join in Developer Workbench - copy of existing synonym - Step 1

4. You are presented with a list of synonyms available in the folder where you started to work. Highlight **CEN_INVENTORY** and click **Select** (Figure 3-48). If the synonym you want to use resides in a different folder, locate it using the **Look in** box at the top of the pane.



Name	Type	Join Conditions	Join Type
cen_inventory	DB2	PRODUCTNUMBER = PRODUCTNUMBER	Matching p
cen_orders	DB2		
cen_orders_with_ri	DB2	PRODUCTNUMBER = PRODUCTNUMBER	Matching p
cen_orders_with_ri	DB2	STORECODE = STORECODE	Matching p
cen_orders_with_ri	DB2	PLANTCODE = PLANTCODE	Matching p
cen_plant	DB2	PLANTCODE = PLANTCODE	Matching p
cen_stores	DB2	STORECODE = STORECODE	Matching p
poo_anart00f	DB2		
s_inventory	DB2	PRODUCTNUMBER = PRODUCTNUMBER	Matching k
s_orders	DB2		
s_orders_cluster	DB2		

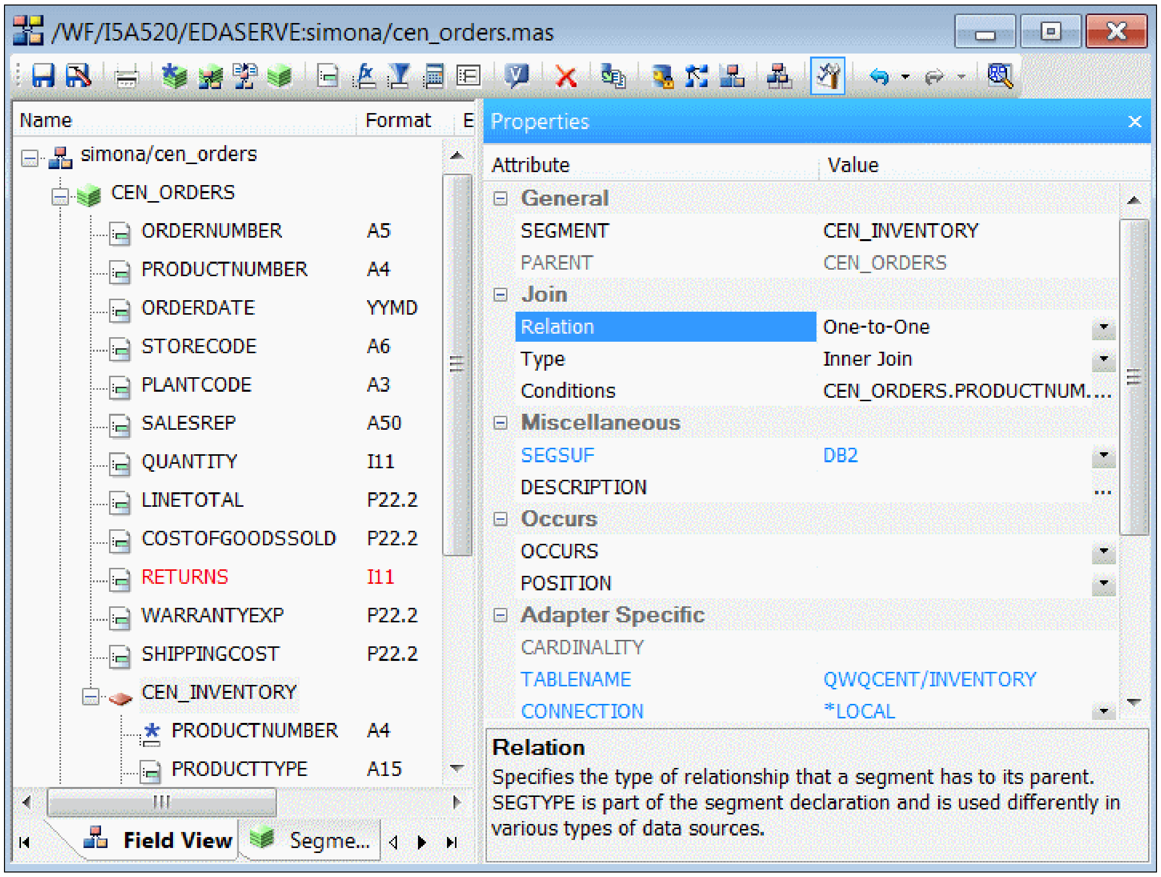
Name: simona/cen_inventory

Types: All Synonym Types

Figure 3-48 Join in Developer Workbench - copy of existing synonym - Step 2

5. On the left-hand part of the pane, notice the join option. The options can be edited and changed where it is necessary using the pull downs and the list items on the side of each definition. In Figure 3-49 you can see what we have selected:

   – Relation = One-to-One
   – Type = Inner Join
   – Condition CEN_ORDERS.PRODUCTNUMBER=CEN_INVENTORY.PRODUCTNUMBER



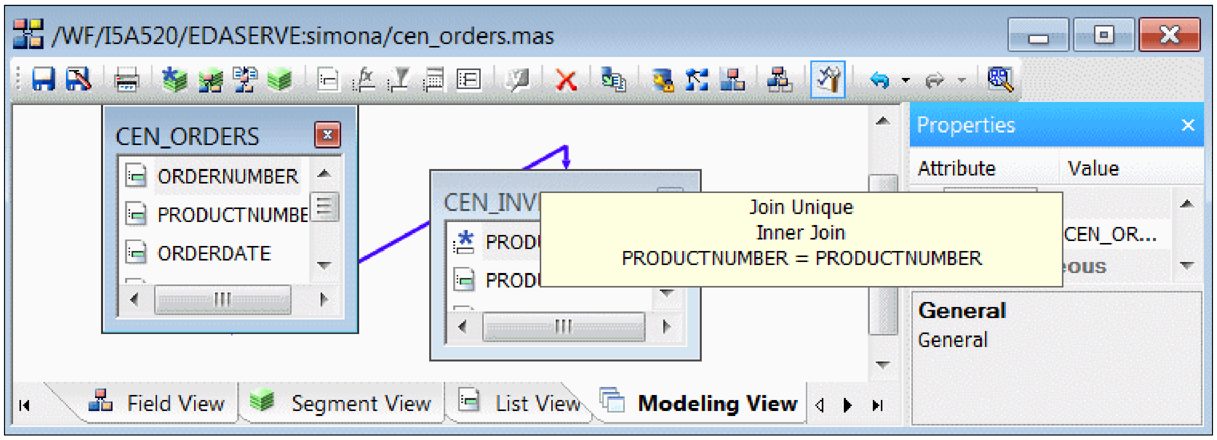
Name	Format
simona/cen_orders	
CEN_ORDERS	
ORDERNUMBER	A5
PRODUCTNUMBER	A4
ORDERDATE	YYMD
STORECODE	A6
PLANTCODE	A3
SALESREP	A50
QUANTITY	I11
LINETOTAL	P22.2
COSTOFGOODSSOLD	P22.2
RETURNS	I11
WARRANTYEXP	P22.2
SHIPPINGCOST	P22.2
CEN_INVENTORY	
PRODUCTNUMBER	A4
PRODUCTTYPE	A15

Attribute	Value
General	
SEGMENT	CEN_INVENTORY
PARENT	CEN_ORDERS
Join	
Relation	One-to-One
Type	Inner Join
Conditions	CEN_ORDERS.PRODUCTNUM....
Miscellaneous	
SEGSUF	DB2
DESCRIPTION	
Occurs	
OCCURS	
POSITION	
Adapter Specific	
CARDINALITY	
TABLENAME	QWQCENT/INVENTORY
CONNECTION	*LOCAL

Relation
Specifies the type of relationship that a segment has to its parent. SEGTYPE is part of the segment declaration and is used differently in various types of data sources.

Figure 3-49 Join in Developer Workbench - copy of existing synonym - Step 3

6. If you want to get a graphical view of the relationship, select the **Modeling View** tab at the bottom of the panel and hover over the arrow between the tables as shown in Figure 3-50.



Join Unique
Inner Join
PRODUCTNUMBER = PRODUCTNUMBER

General
General

Figure 3-50 Join in Developer Workbench - copy of existing synonym - Step 4

7. When the line connecting the two files is dotted, it means that some elements are missing in the join definition. You can always see and edit the join conditions by right-clicking the line and selecting **join properties**, as shown in Figure 3-51.

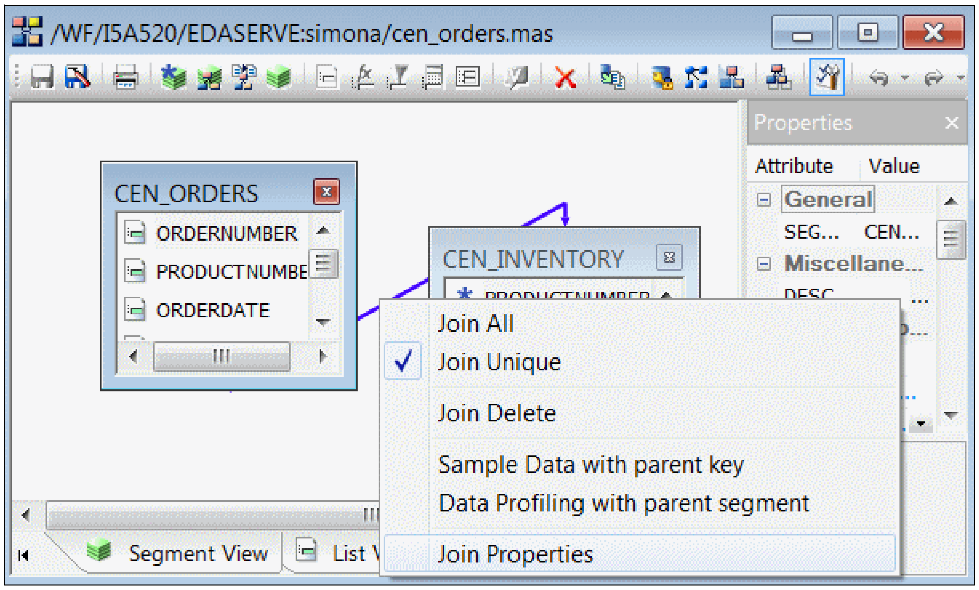
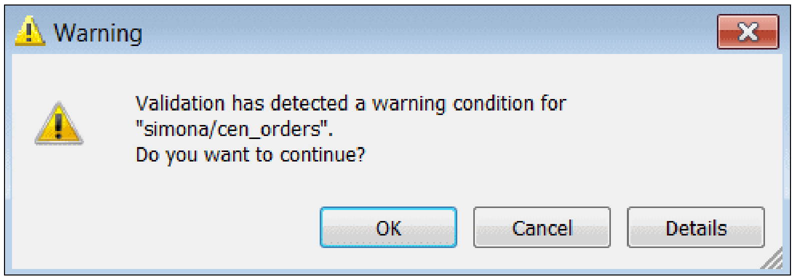


Figure 3-51 Join in Developer Workbench - copy of existing synonym - edit condition

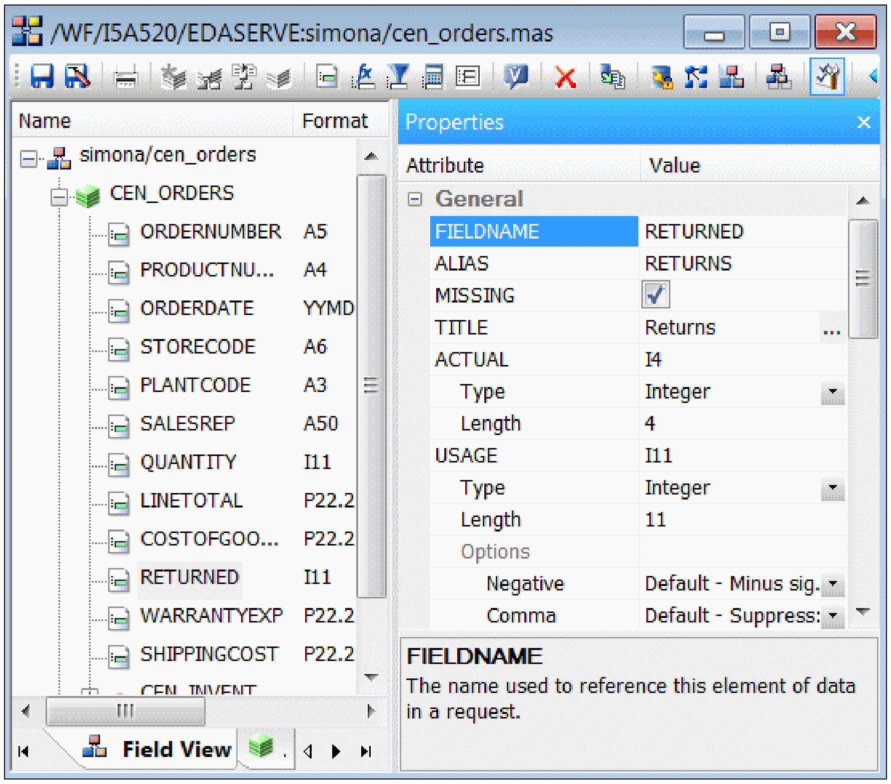
8. Save the change using the Save icon on the tool bar. You may receive a message regarding a warning condition (Figure 3-52); select **OK** to continue saving.


Warning

Validation has detected a warning condition for "simona/cen_orders".
Do you want to continue?

Figure 3-52 Join in Developer Workbench - copy of existing synonym - error message on save

This error message is caused by one of the columns in the Orders tables, which is named RETURNS. This is a reserved word in SQL. If you want to get rid of this message, rename the field as shown in Figure 3-53 and save again.


/WF/I5A520/EDASERVE:simona/cen_orders.mas

Name	Format
simona/cen_orders	
CEN_ORDERS	
ORDERNUMBER	A5
PRODUCTNU...	A4
ORDERDATE	YYMD
STORECODE	A6
PLANTCODE	A3
SALESREP	A50
QUANTITY	I11
LINETOTAL	P22.2
COSTOFGOO...	P22.2
RETURNED	I11
WARRANTYEXP	P22.2
SHIPPINGCOST	P22.2
CEN_INVENT	

Properties

Attribute	Value
General	
FIELDNAME	RETURNED
ALIAS	RETURNS
MISSING	✓
TITLE	Returns
ACTUAL	I4
Type	Integer
Length	4
USAGE	I11
Type	Integer
Length	11
Options	
Negative	Default - Minus sig.
Comma	Default - Suppress:

FIELDNAME
The name used to reference this element of data in a request.

Figure 3-53 Rename field

Note: Rename in the FIELDNAME (the name used by DB2 Web Query). If you change the ALIAS, the metadata will not work any more. Against logic, the term ALIAS is used to refer to the actual column name in the database.

Repeat the process for CEN_PLANT and CEN_STORES to get the full environment.

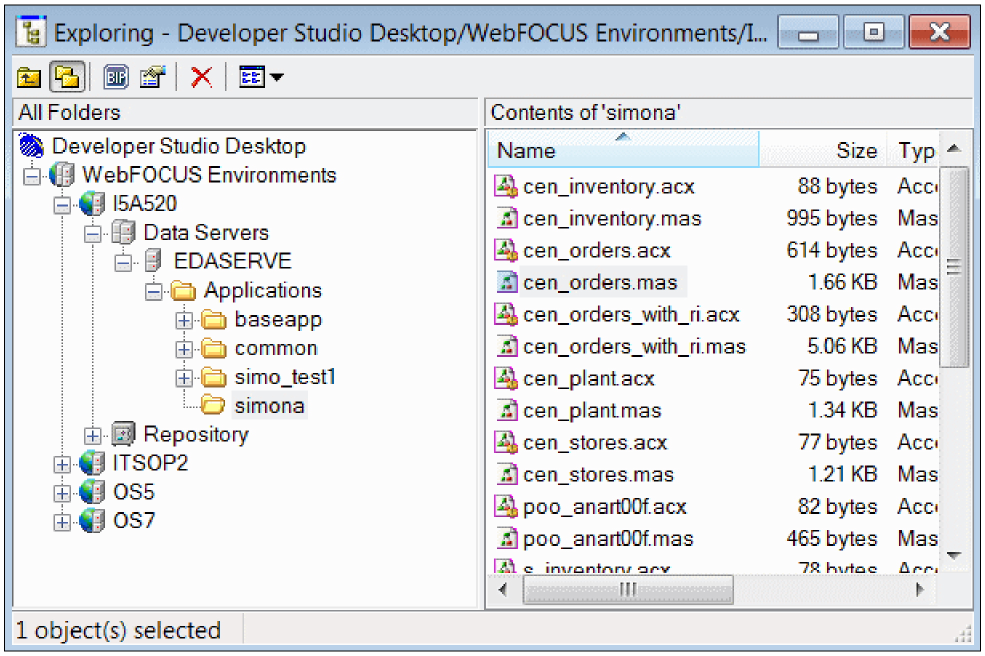
After COPYING the synonym, the original CEN_INVENTORY, CEN_PLANT and CEN_STORES can be deleted unless you want to use them for other purposes.

Remember that if your report only reference columns in one segment (table) the product will generate an SQL statement to reference that one object even if the metadata used has references to multiple segments.

Join - segment via metadata import

Follow these steps:

1. Open Developer Workbench, select the system on which to work, and validate with your userid and password. Remember that the user must be part of the DevWorkbench group (use **Administration** → **Security Center** in the web interface to set up). Select **Data Servers** → **EDASERVE** → **Applications** → ***<yourfolder>*** as shown in Figure 3-54 and double-click the .mas file of the synonym you want to edit.



Name	Size	Typ
cen_inventory.acx	88 bytes	Acc
cen_inventory.mas	995 bytes	Mas
cen_orders.acx	614 bytes	Acc
cen_orders.mas	1.66 KB	Mas
cen_orders_with_ri.acx	308 bytes	Acc
cen_orders_with_ri.mas	5.06 KB	Mas
cen_plant.acx	75 bytes	Acc
cen_plant.mas	1.34 KB	Mas
cen_stores.acx	77 bytes	Acc
cen_stores.mas	1.21 KB	Mas
poo_anart00f.acx	82 bytes	Acc
poo_anart00f.mas	465 bytes	Mas

Figure 3-54 Join in Developer Workbench - Step 1 - Import

2. Right-click the segment to which you want to add a join and select **Insert** → **Segment via metadata** import (Figure 3-55).

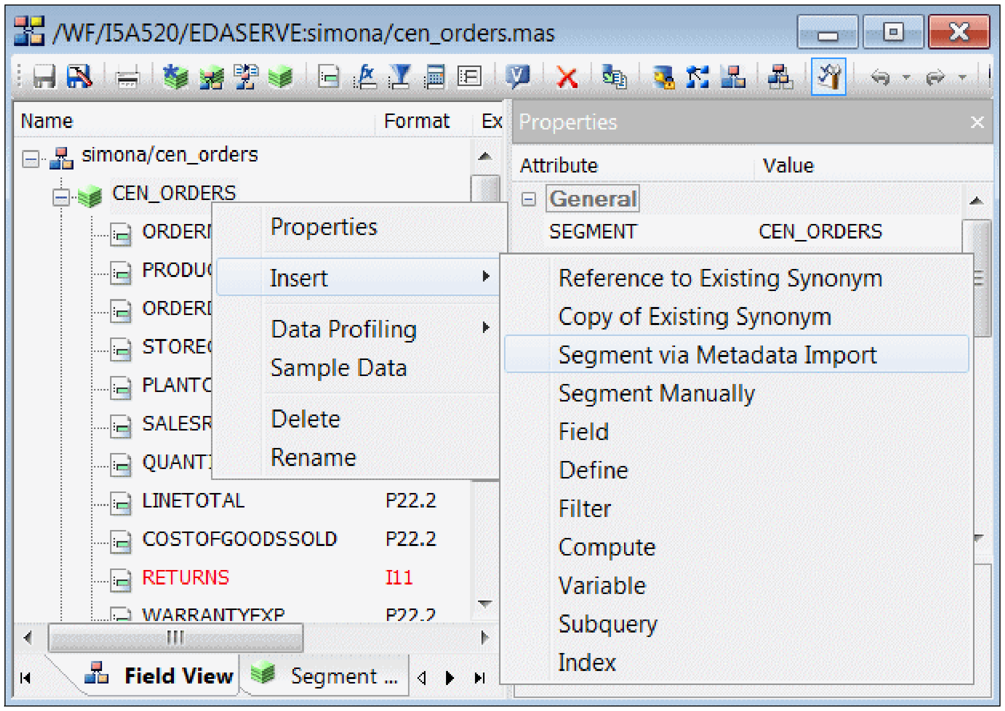


Figure 3-55 Join in Developer Workbench - Import - Step 2

3. You are presented with a panel where you can choose the data source. Select ***LOCAL** as shown in Figure 3-56 and click **OK**.

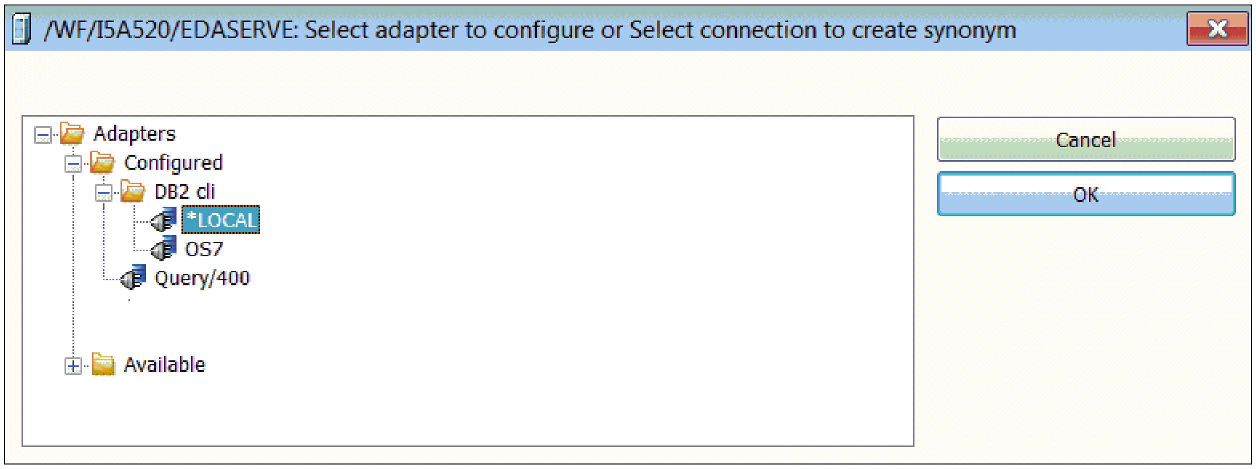
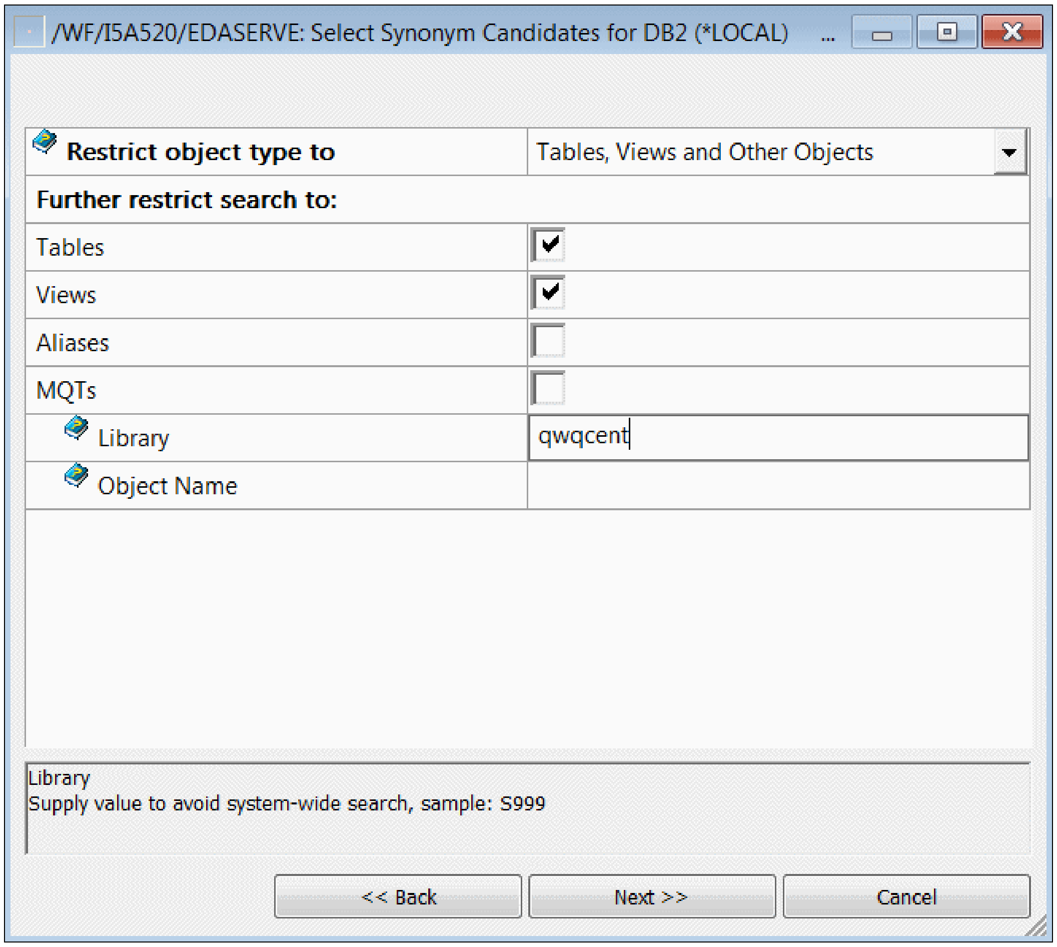

/WF/I5A520/EDASERVE: Select adapter to configure or Select connection to create synonym
Adapters
Configured
DB2 cli
*LOCAL
OS7
Query/400
Available
Cancel
OK

Figure 3-56 Join in Developer Workbench - Import - Step 3

4. Set the name of the library - **QWQCENT** - and select **Next** (Join in Developer Workbench - Import - Step 4).

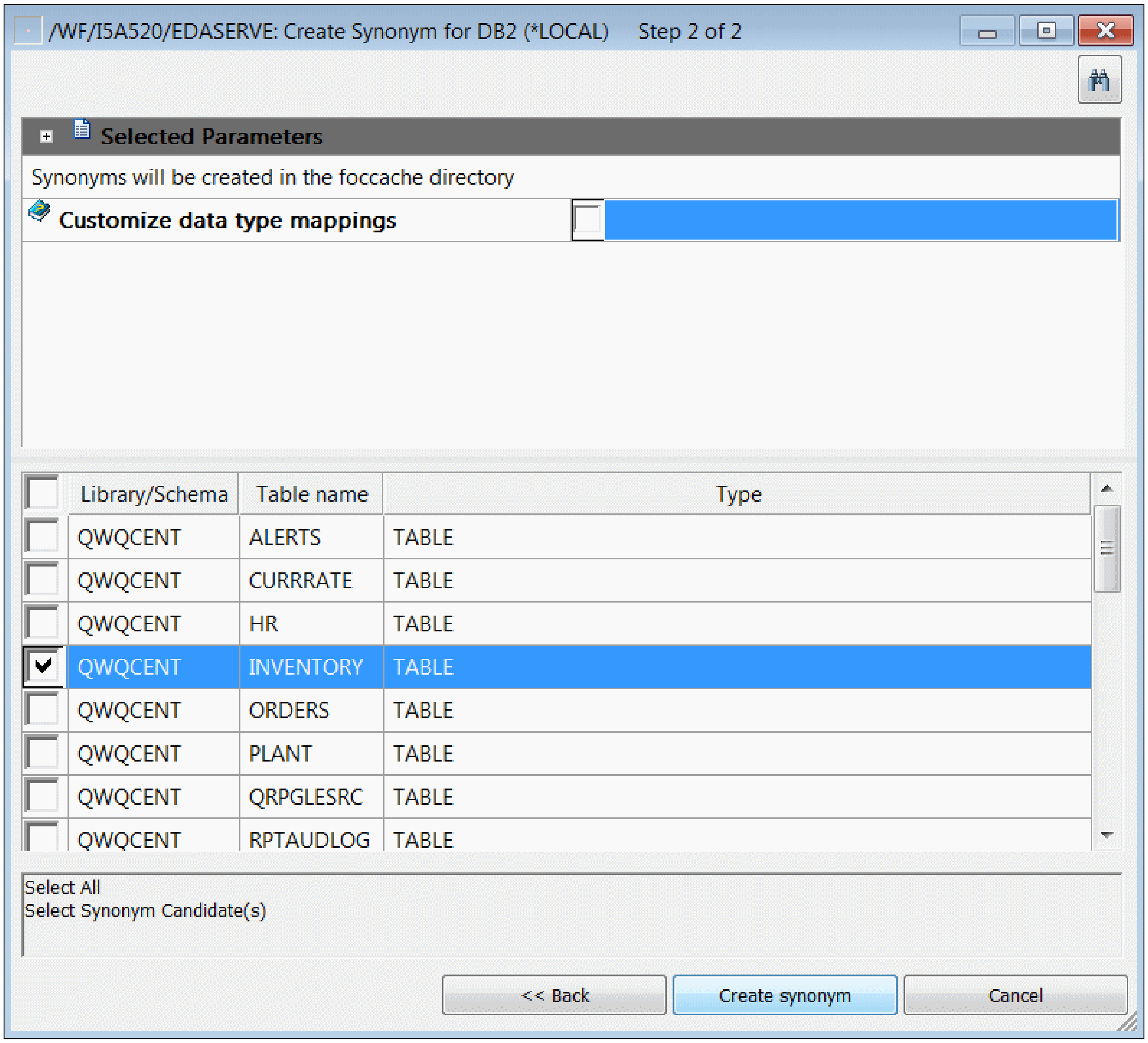

/WF/I5A520/EDASERVE: Select Synonym Candidates for DB2 (*LOCAL) ...
<< Back
Next >>
Cancel

Restrict object type to	Tables, Views and Other Objects
Further restrict search to:	
Tables	✓
Views	✓
Aliases	
MQTs	
Library	qwqcent
Object Name	

Library
Supply value to avoid system-wide search, sample: S999

Figure 3-57 Join in Developer Workbench - Import - Step 4

5. Select the table you want to join in, **INVENTORY** in this example, and click **Create Synonym** (Figure 3-58).


/WF/I5A520/EDASERVE: Create Synonym for DB2 (*LOCAL)   Step 2 of 2

Selected Parameters

Synonyms will be created in the foccache directory

Customize data type mappings

☐	Library/Schema	Table name	Type
☐	QWQCENT	ALERTS	TABLE
☐	QWQCENT	CURRRATE	TABLE
☐	QWQCENT	HR	TABLE
✓	QWQCENT	INVENTORY	TABLE
☐	QWQCENT	ORDERS	TABLE
☐	QWQCENT	PLANT	TABLE
☐	QWQCENT	QRPGLESRC	TABLE
☐	QWQCENT	RPTAUDLOG	TABLE

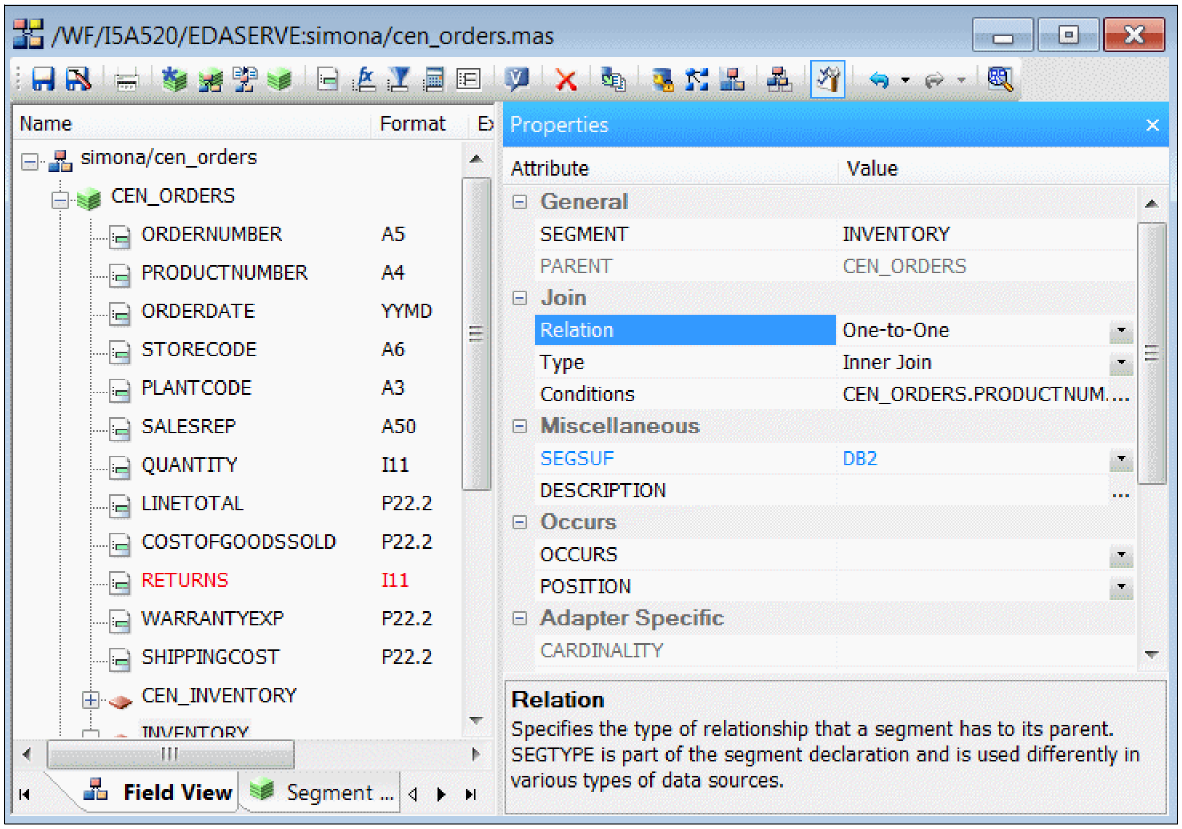
Select All
Select Synonym Candidate(s)

Figure 3-58 Join in Developer Workbench - Import - Step 5

6. You will get a message saying that the segment has been created successfully. Click **Close** to get back to the synonym editor.
7. In the synonym editor you are presented with the option to edit the join relationship. The options can be edited and changed where it is necessary using the pull-downs and the list items on the side of each definition.

In Figure 3-59, you can see what we have selected:

– Relation = One-to-One
– Type = Inner Join
– Condition CEN_ORDERS.PRODUCTNUMBER=CEN_INVENTORY.PRODUCTNUMBER (calculator can be used to define join condition, as documented in “Join - copy of existing synonym” on page 67).


/WF/I5A520/EDASERVE:simona/cen_orders.mas

Name	Format
simona/cen_orders	
CEN_ORDERS	
ORDERNUMBER	A5
PRODUCTNUMBER	A4
ORDERDATE	YYMD
STORECODE	A6
PLANTCODE	A3
SALESREP	A50
QUANTITY	I11
LINETOTAL	P22.2
COSTOFGOODSSOLD	P22.2
RETURNS	I11
WARRANTYEXP	P22.2
SHIPPINGCOST	P22.2
CEN_INVENTORY	
INVENTORY	

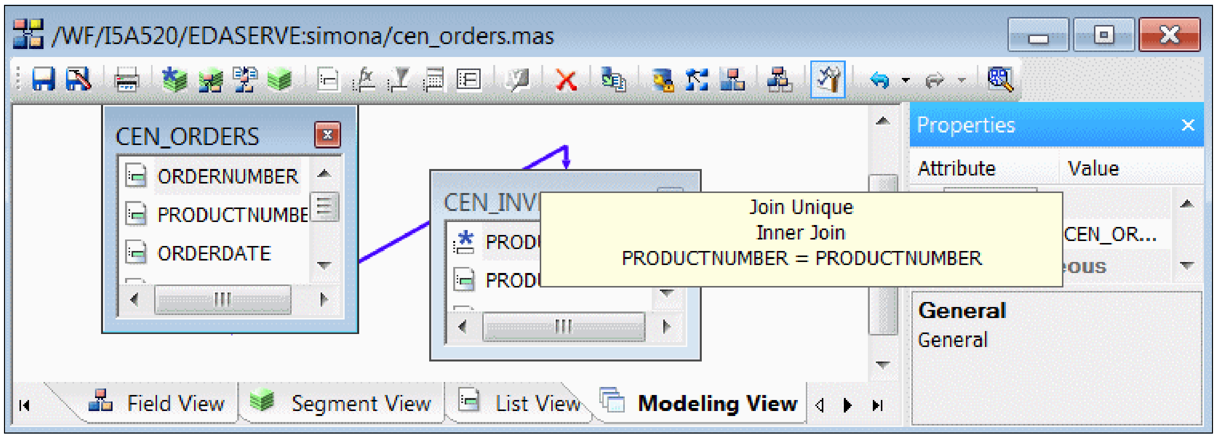
Properties

Attribute	Value
General	
SEGMENT	INVENTORY
PARENT	CEN_ORDERS
Join	
Relation	One-to-One
Type	Inner Join
Conditions	CEN_ORDERS.PRODUCTNUM....
Miscellaneous	
SEGSUF	DB2
DESCRIPTION	
Occurs	
OCCURS	
POSITION	
Adapter Specific	
CARDINALITY	

Relation
Specifies the type of relationship that a segment has to its parent. SEGTYPE is part of the segment declaration and is used differently in various types of data sources.

Figure 3-59 Join in Developer Workbench - Import - Step 6

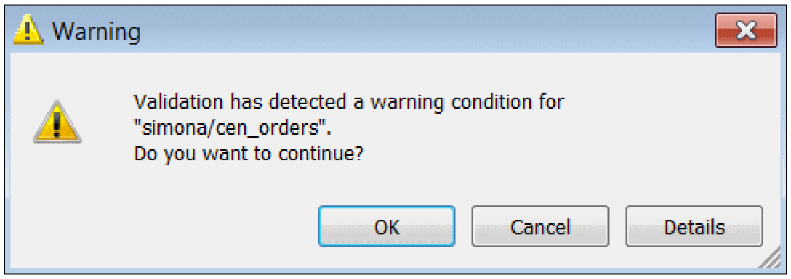
8. If you want to get a graphical view of the relationship, select the **Modeling View** tab at the bottom of the panel and hover over the arrow between the tables as shown in Figure 3-60.


/WF/I5A520/EDASERVE:simona/cen_orders.mas

Join Unique
Inner Join
PRODUCTNUMBER = PRODUCTNUMBER

Figure 3-60 Join in Developer Workbench - import - Step 7

9. Save the change using the disc icon on the tool bar. You may receive a message regarding a warning condition (Figure 3-61); select **Ok** to continue saving.

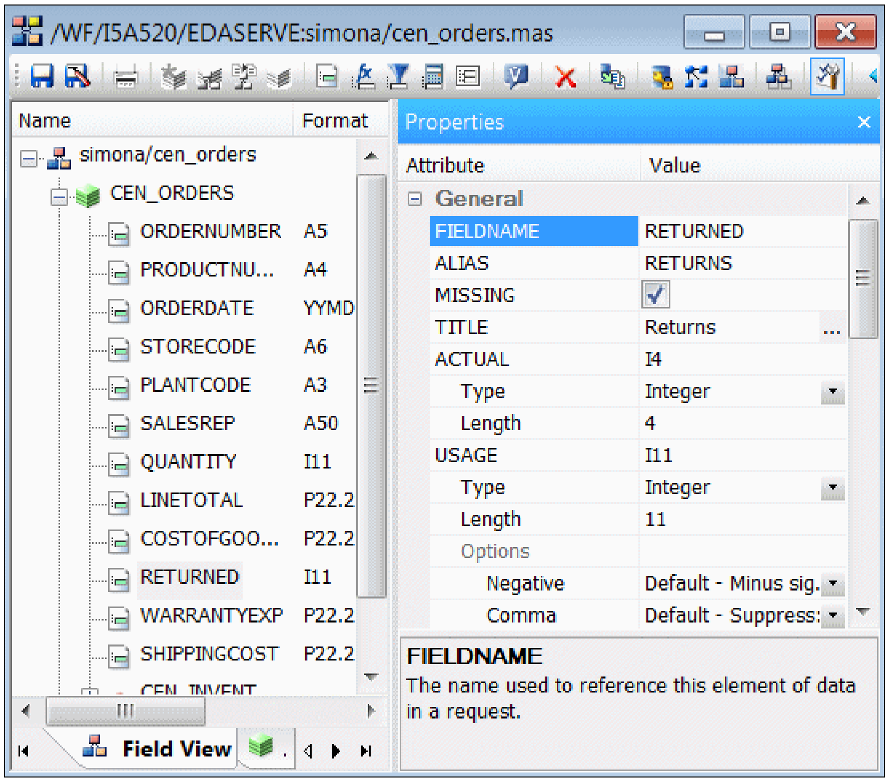


Warning

Validation has detected a warning condition for "simona/cen_orders". Do you want to continue?

Figure 3-61 Join in Developer Workbench - import - error message on save

This error message is caused by one of the columns in the Orders tables, which is named RETURNS. This is a reserved word in SQL. If you want to get rid of this message, rename the field as shown in Figure 3-62 and save again.



/WF/I5A520/EDASERVE:simona/cen_orders.mas

Name	Format
simona/cen_orders	
CEN_ORDERS	
ORDERNUMBER	A5
PRODUCTNU...	A4
ORDERDATE	YYMD
STORECODE	A6
PLANTCODE	A3
SALESREP	A50
QUANTITY	I11
LINETOTAL	P22.2
COSTOFGOO...	P22.2
RETURNED	I11
WARRANTYEXP	P22.2
SHIPPINGCOST	P22.2

Properties

Attribute	Value
General	
FIELDNAME	RETURNED
ALIAS	RETURNS
MISSING	✓
TITLE	Returns
ACTUAL	I4
Type	Integer
Length	4
USAGE	I11
Type	Integer
Length	11
Options	
Negative	Default - Minus sig.
Comma	Default - Suppress:

FIELDNAME
The name used to reference this element of data in a request.

Figure 3-62 Rename field

Note: Rename in the FIELDNAME (the name used by DB2 Web Query). If you change the ALIAS, the metadata will not work any more. Against logic, the term ALIAS is used to refer to the actual column name in database.

Repeat the process for tables PLANT and STORES to get the full environment.

Remember that if your report only references columns in one segment (table), the product will generate an SQL statement to reference that one object even if the metadata used has references to multiple segments.

3.7 Date decomposition

In our ORDERS table, there is a date column, ORDERDATE. We want to be able to use its primary logical elements (YEAR, MONTH, QUARTER, DAY) when reporting. This is very useful when we have to aggregate and compare elements based on the date basic components (that is, compare sales for year xx, on a quarter basis) without having to code complex formulas and calculation in our reports and we want these functionalities to be available to all report developers without any hassles.

To overcome this situation, DB2 Web Query provides a DATE DECOMPOSITION functionality, really easy to implement. This can be obtained both with the web interface or by using the Developer Workbench tool.

In the latest releases of DB2 Web Query, applying the date decomposition function the date transformation gets pushed to the database level, applying to appropriate SQL function on the date field. For instance, if you want to use the year portion of the order date field in a report, you will see YEAR(ORDERDATE) coded in the underneath SQL statement. This may lead to significant performance improvements when compared to the previous releases, when this request would have lead to a transformation at a higher level.

Either the web interface or the Developer Workbench tool can be used to decompose date fields. Here we present both, for your reference.

Note: It is now possible to get a field decomposed “automatically” while building a DATE DIMENSION; see 3.12, “Defining dimensions: InfoMini and OLAP” on page 124.

3.7.1 Date decomposition with web interface

In our sample database in the ORDERS table, we have a date field, ORDERDATE, from which we want to gather basic elements, YEAR, QUARTER, MONTH, and DAY. To decompose the ORDERDATE date field with the web interface, proceed as follows:

1. Right-click folder **Century Electronics** and select **Metadata** → **Edit** as shown in Figure 3-63.

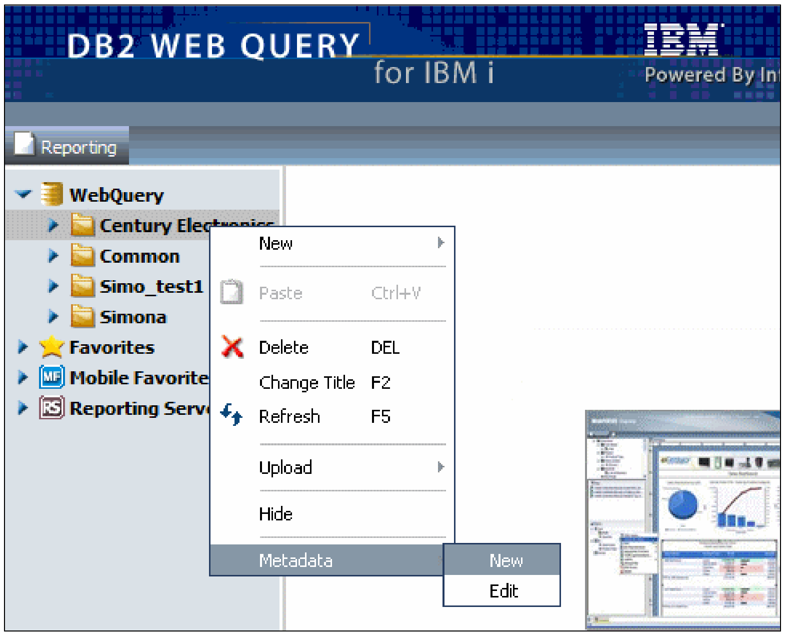


Figure 3-63 Date decomposition - Step 1

2. Right-click **cen_orders** and select **Open** (Figure 3-64).

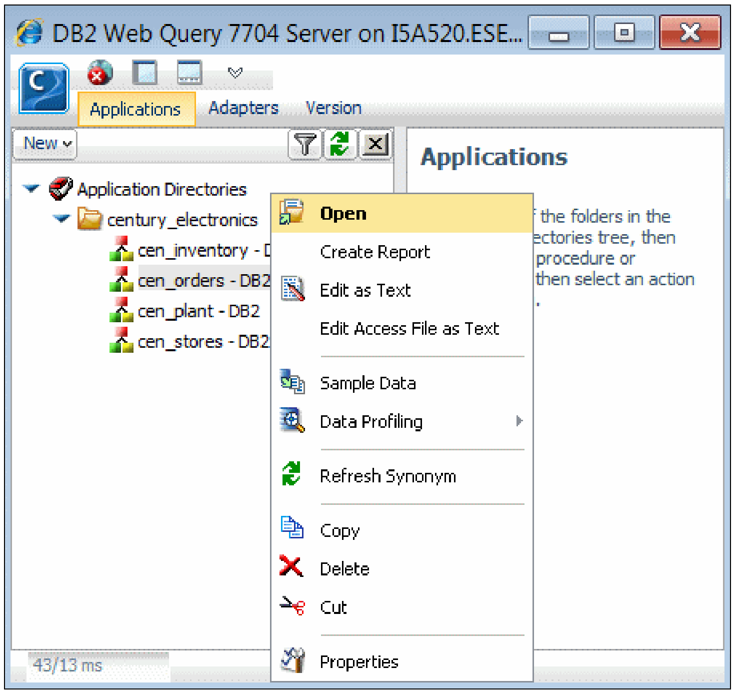


Figure 3-64 Date decomposition - Step 2

3. The synonym editor opens up. Right-click field **ORDERDATE** and select **Decompose Date** (Figure 3-65).

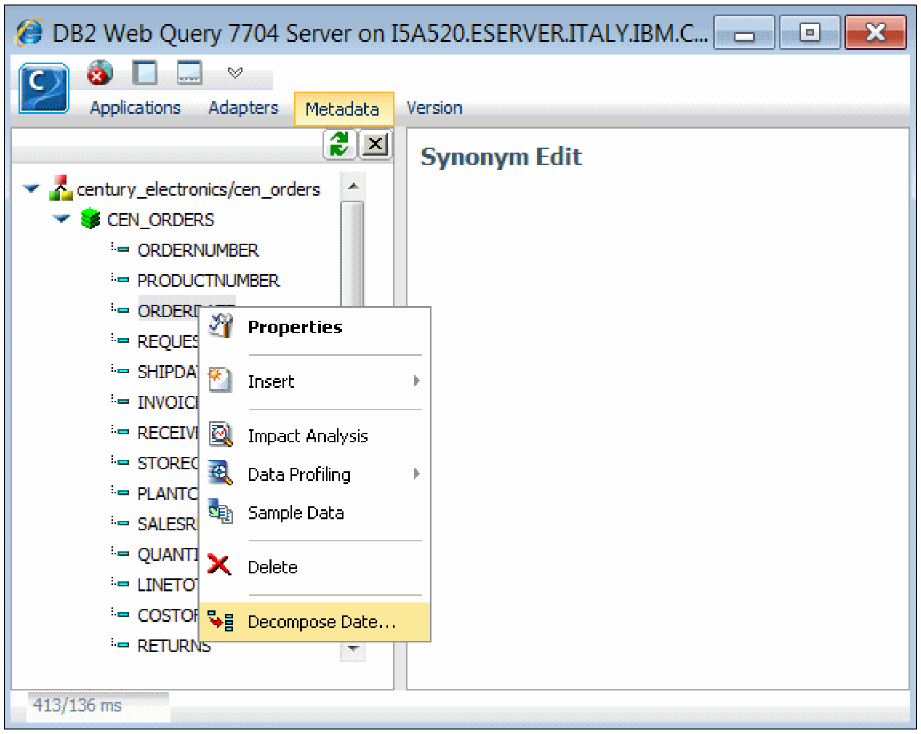


Figure 3-65 Date decomposition - Step 3

4. The synonym gets updated with four new fields representing the date base component, as shown in Figure 3-66.

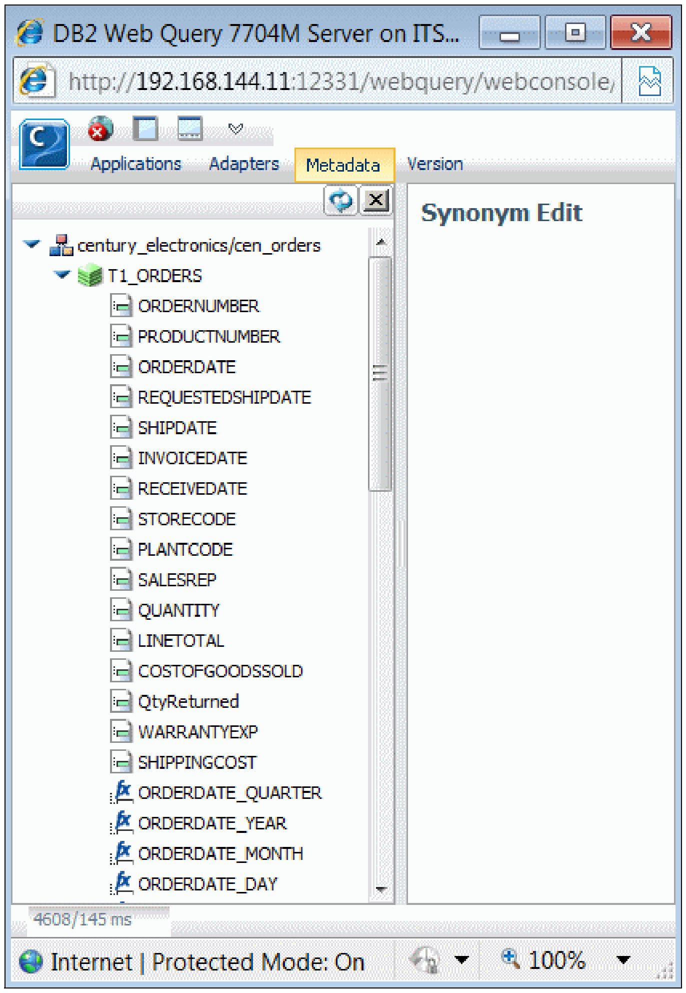


Figure 3-66 Date decomposed

3.7.2 Date decomposition using Developer Workbench

To decompose a date field using the Developer Workbench synonym editor, proceed as follows:

1. Open Developer Workbench, select the system on which to work, and validate with your userid and password. Remember that the user must be part of the DevWorkbench group (use **Administration** → **Security Center** in the web interface to set up - see Chapter 4, "Security Center" on page 149). Select **Data Servers** → **EDASERVE** → **Applications** → <***yourfolder***> and double-click the .mas file of the synonym that you want to edit.

2. In your synonym, right-click the field date to be decomposed (**ORDERDATE** in this example) and select **Decompose Date** as shown in Figure 3-67.

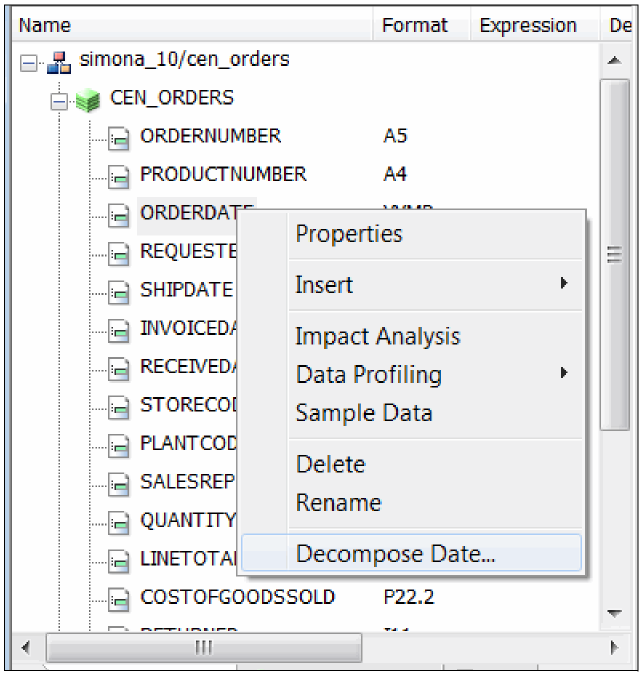
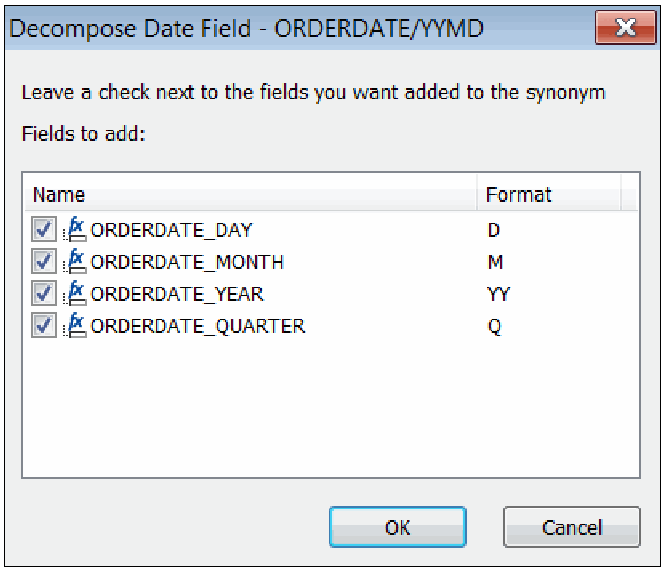


Figure 3-67   Decompose date with Developer Workbench - Step 1

3. You are presented with a panel (Figure 3-68) where you can select the decomposition items: Day, Month, Quarter and Year. Select **OK**.



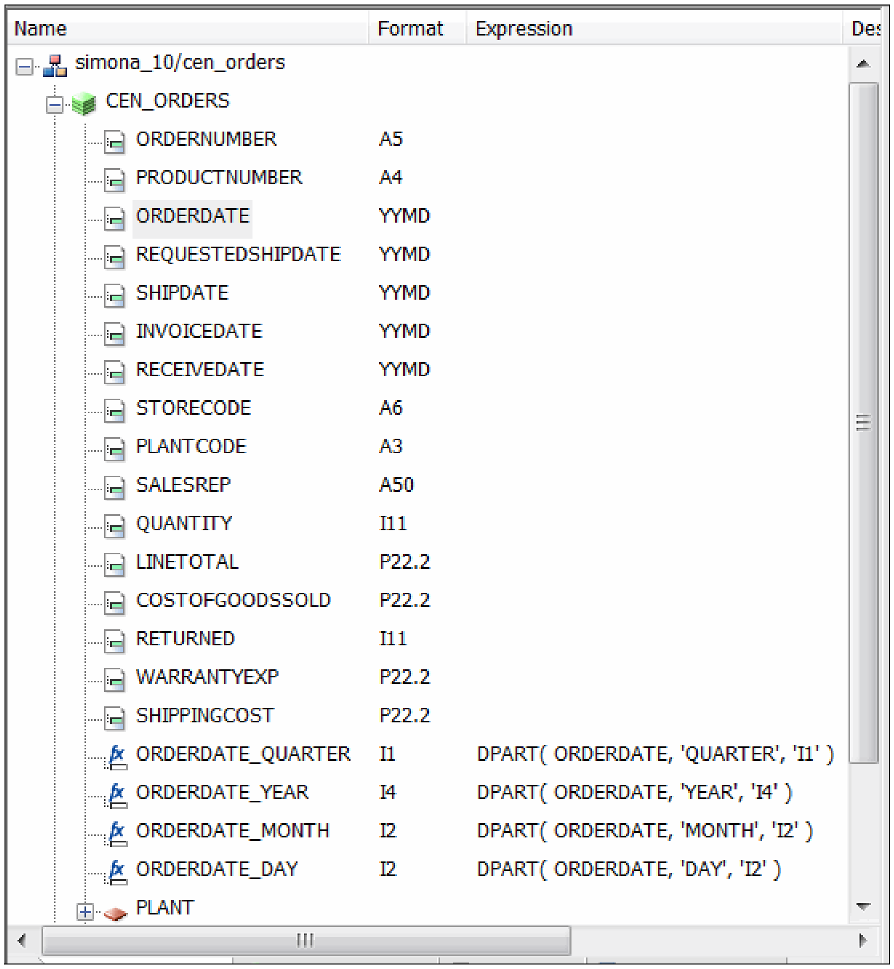
Leave a check next to the fields you want added to the synonym

Fields to add:

Name	Format
✓ ORDERDATE_DAY	D
✓ ORDERDATE_MONTH	M
✓ ORDERDATE_YEAR	YY
✓ ORDERDATE_QUARTER	Q

Figure 3-68   Decompose date with Developer Workbench - Step 2

4. In Figure 3-69, you can see the decomposition results. Save your synonym and exit.



Name	Format	Expression	Des
simona_10/cen_orders			
CEN_ORDERS			
ORDERNUMBER	A5		
PRODUCTNUMBER	A4		
ORDERDATE	YYMD		
REQUESTEDSHIPDATE	YYMD		
SHIPDATE	YYMD		
INVOICEDATE	YYMD		
RECEIVEDATE	YYMD		
STORECODE	A6		
PLANTCODE	A3		
SALESREP	A50		
QUANTITY	I11		
LINETOTAL	P22.2		
COSTOFGOODSSOLD	P22.2		
RETURNED	I11		
WARRANTYEXP	P22.2		
SHIPPINGCOST	P22.2		
ORDERDATE_QUARTER	I1	DPART(ORDERDATE, 'QUARTER', 'I1')	
ORDERDATE_YEAR	I4	DPART(ORDERDATE, 'YEAR', 'I4')	
ORDERDATE_MONTH	I2	DPART(ORDERDATE, 'MONTH', 'I2')	
ORDERDATE_DAY	I2	DPART(ORDERDATE, 'DAY', 'I2')	
PLANT			

Figure 3-69 Decompose date with Developer Workbench

3.8 Understanding your data

To make your reporting successful, you have to know and understand your data:

- How many rows are there in tables?
- How many rows are there for any “differentiator” in your tables (% and total numbers)? Even or uneven distribution?
- What are the relationships between tables?

There are instances where the report developers may not have a good understanding of the data they are dealing with, and this may lead to unexpected results.

To help developers with understanding the data structure, DB2 Web Query provides some nice features, both in the web interface and in Developer Workbench.

Here we show you these features using the web interface.

Right-click the synonym you want to analyze, as shown in Figure 3-70 on page 84.

Notice the following options (Figure 3-70):

- Sample data
- Data profiling:
  - Statistics
  - Count
  - Key analysis

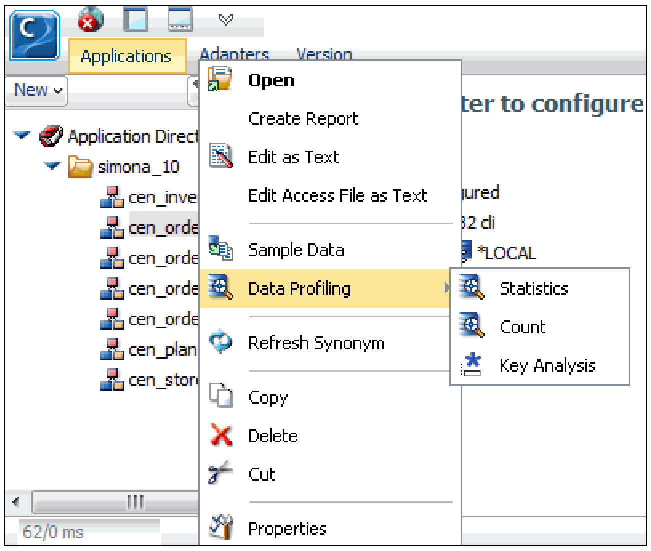
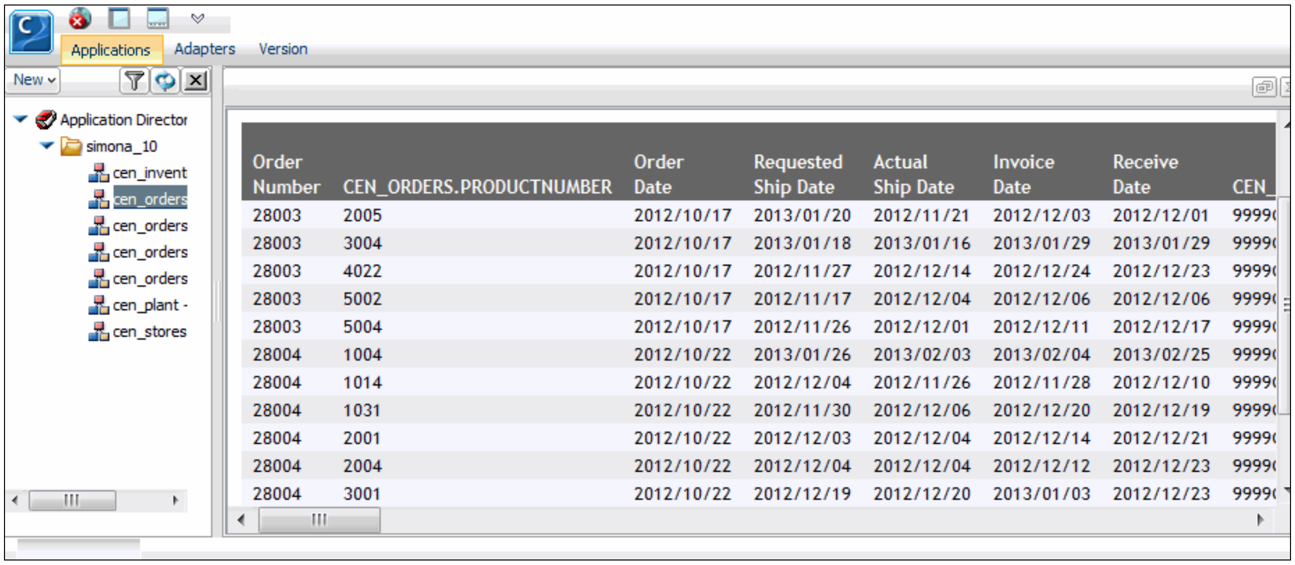


Figure 3-70 Web Interface - understanding your data

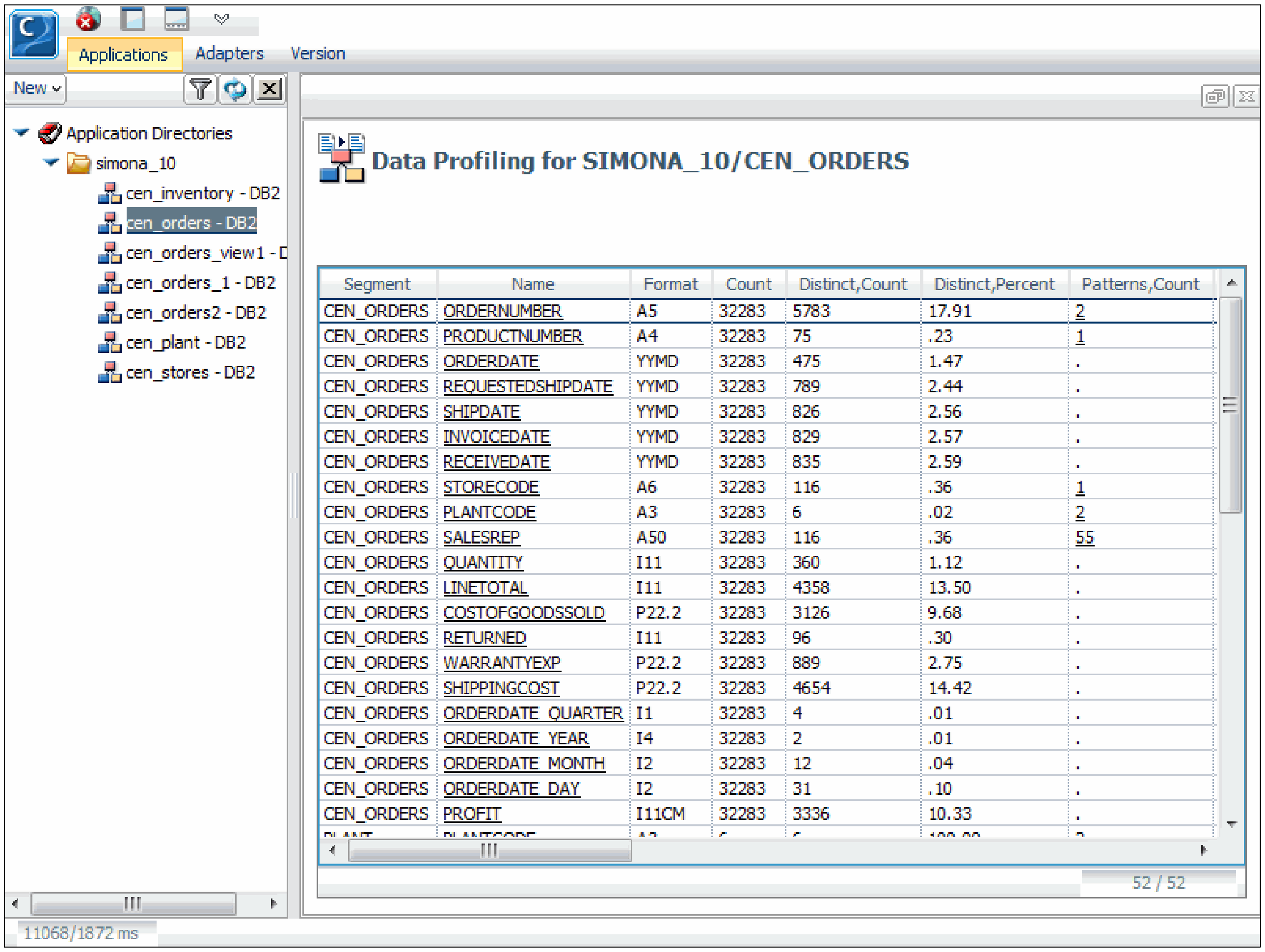
In Figure 3-71 you can see the SAMPLE DATA output.



Order Number	CEN_ORDERS.PRODUCTNUMBER	Order Date	Requested Ship Date	Actual Ship Date	Invoice Date	Receive Date	CEN_
28003	2005	2012/10/17	2013/01/20	2012/11/21	2012/12/03	2012/12/01	9999
28003	3004	2012/10/17	2013/01/18	2013/01/16	2013/01/29	2013/01/29	9999
28003	4022	2012/10/17	2012/11/27	2012/12/14	2012/12/24	2012/12/23	9999
28003	5002	2012/10/17	2012/11/17	2012/12/04	2012/12/06	2012/12/06	9999
28003	5004	2012/10/17	2012/11/26	2012/12/01	2012/12/11	2012/12/17	9999
28004	1004	2012/10/22	2013/01/26	2013/02/03	2013/02/04	2013/02/25	9999
28004	1014	2012/10/22	2012/12/04	2012/11/26	2012/11/28	2012/12/10	9999
28004	1031	2012/10/22	2012/11/30	2012/12/06	2012/12/20	2012/12/19	9999
28004	2001	2012/10/22	2012/12/03	2012/12/04	2012/12/14	2012/12/21	9999
28004	2004	2012/10/22	2012/12/04	2012/12/04	2012/12/12	2012/12/23	9999
28004	3001	2012/10/22	2012/12/19	2012/12/20	2013/01/03	2012/12/23	9999

Figure 3-71 Sample data output

In Figure 3-72, you can see the STATISTICS output.

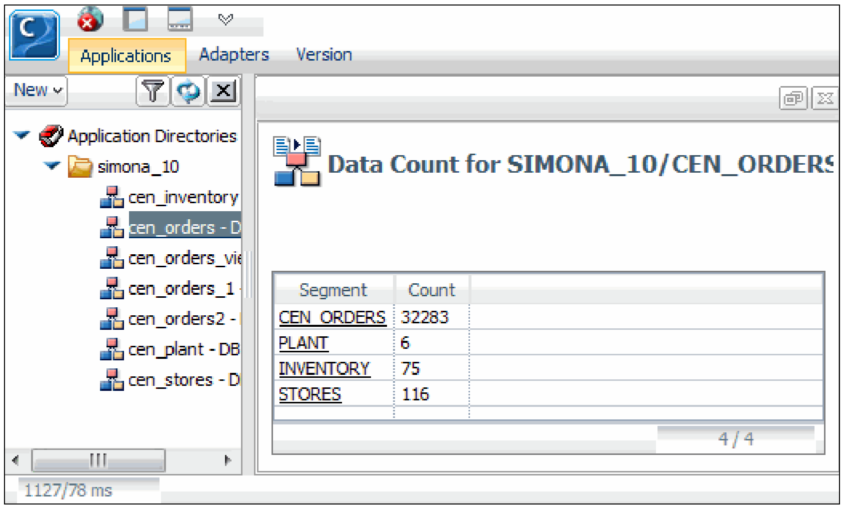


Data Profiling for SIMONA_10/CEN_ORDERS

Segment	Name	Format	Count	Distinct,Count	Distinct,Percent	Patterns,Count
CEN_ORDERS	ORDERNUMBER	A5	32283	5783	17.91	2
CEN_ORDERS	PRODUCTNUMBER	A4	32283	75	.23	1
CEN_ORDERS	ORDERDATE	YYMD	32283	475	1.47	.
CEN_ORDERS	REQUESTEDSHIPDATE	YYMD	32283	789	2.44	.
CEN_ORDERS	SHIPDATE	YYMD	32283	826	2.56	.
CEN_ORDERS	INVOICEDATE	YYMD	32283	829	2.57	.
CEN_ORDERS	RECEIVEDATE	YYMD	32283	835	2.59	.
CEN_ORDERS	STORECODE	A6	32283	116	.36	1
CEN_ORDERS	PLANTCODE	A3	32283	6	.02	2
CEN_ORDERS	SALESREP	A50	32283	116	.36	55
CEN_ORDERS	QUANTITY	I11	32283	360	1.12	.
CEN_ORDERS	LINETOTAL	I11	32283	4358	13.50	.
CEN_ORDERS	COSTOFGOODSSOLD	P22.2	32283	3126	9.68	.
CEN_ORDERS	RETURNED	I11	32283	96	.30	.
CEN_ORDERS	WARRANTYEXP	P22.2	32283	889	2.75	.
CEN_ORDERS	SHIPPINGCOST	P22.2	32283	4654	14.42	.
CEN_ORDERS	ORDERDATE_QUARTER	I1	32283	4	.01	.
CEN_ORDERS	ORDERDATE_YEAR	I4	32283	2	.01	.
CEN_ORDERS	ORDERDATE_MONTH	I2	32283	12	.04	.
CEN_ORDERS	ORDERDATE_DAY	I2	32283	31	.10	.
CEN_ORDERS	PROFIT	I11CM	32283	3336	10.33	.

Figure 3-72 Web Interface - Statistics

In Figure 3-73, you can see the COUNT output.



Data Count for SIMONA_10/CEN_ORDERS

Segment	Count
CEN_ORDERS	32283
PLANT	6
INVENTORY	75
STORES	116

Figure 3-73 Web Interface - Count

Similar functions are available in Developer Workbench.

3.9 Field formatting

It is very common that to present information nicely, you may need to change the default formatting of fields (decimal, thousands delimiters, currency symbol, titles, and so on). This can be done report by report (which can be a very cumbersome task), or you can exploit the power of metadata and just do it once in the synonym and get the new formatting in all of the reports where the synonym is used.

You can use either the web interface or the Developer Workbench tool. Here, for your reference, we present both ways.

3.9.1 Field formatting with the web interface

In our example data library QWQCENT/ORDERS, we have a LINETOTAL field that is defined as a Decimal field. However, in our reports, we prefer it to be presented as an INTEGER, with thousands delimiters and the $ currency symbol. Also, we are in a multilingual environment and we want to provide titles in different languages, according to the various users' preferences.

To change the way the LINETOTAL field is presented, proceed as follows:

1. In the web interface, select the folder where the synonym is located, right-click, and select **Metadata** → **Edit**.
2. Locate the synonym you want to edit in the list, double-click it or right-click it, and select **Open** as shown in Figure 3-81.

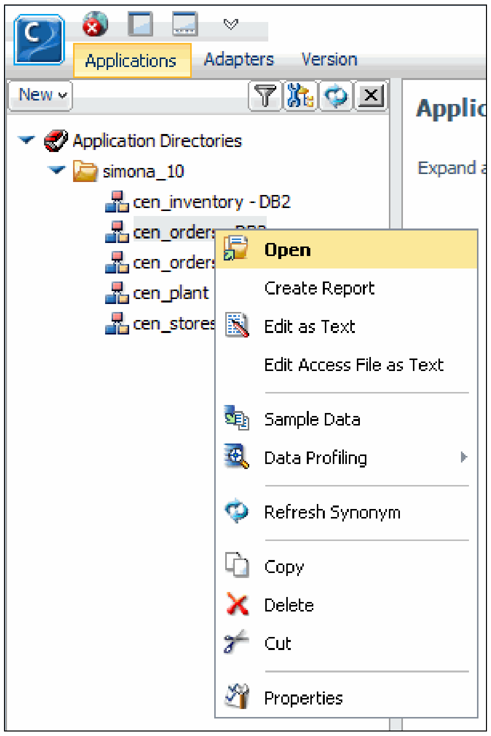
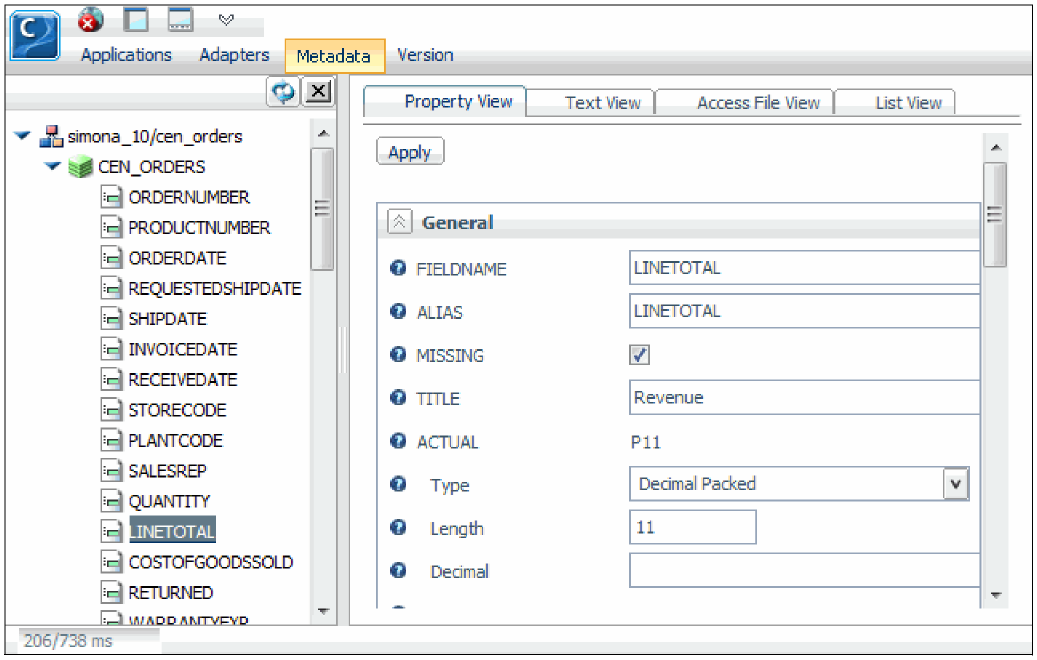


Figure 3-74 Open synonym with web interface

3. You are presented with a panel with all the existing fields. Double-click the **LINETOTAL** field. On the right-hand side of the panel, you will be presented with all details on the selected field, starting from the **Property View** tab, as shown in Figure 3-75.



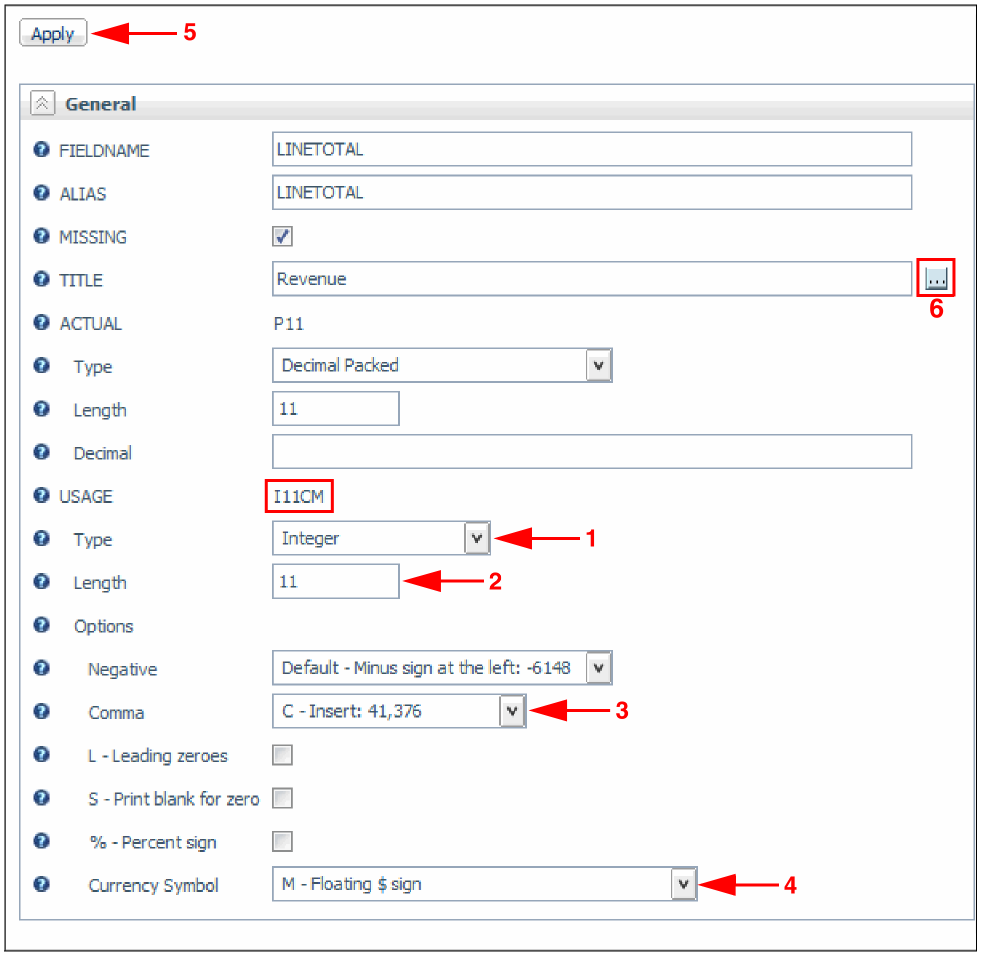
FIELDNAME: LINETOTAL
ALIAS: LINETOTAL
MISSING: ✓
TITLE: Revenue
ACTUAL: P11
Type: Decimal Packed
Length: 11
Decimal:

Figure 3-75 Web interface - format field 1

4. Here you can make all the changes required. In the USAGE portion, set the field to show as a INTEGER (#1), 11 digits long (#2), then select **Comma** (#3) and **Floating $** (#4). Select the **Apply** button to confirm your changes, as shown in Figure 3-76.

Important: Only change the USAGE portion; if you change the ACTUAL section, you end up with a database mismatch.

Note: The appearance of the thousands delimiter depends on your regional/national settings. Even if the caption says "comma," you will get a . as your delimiter if you have a European environmental setting.



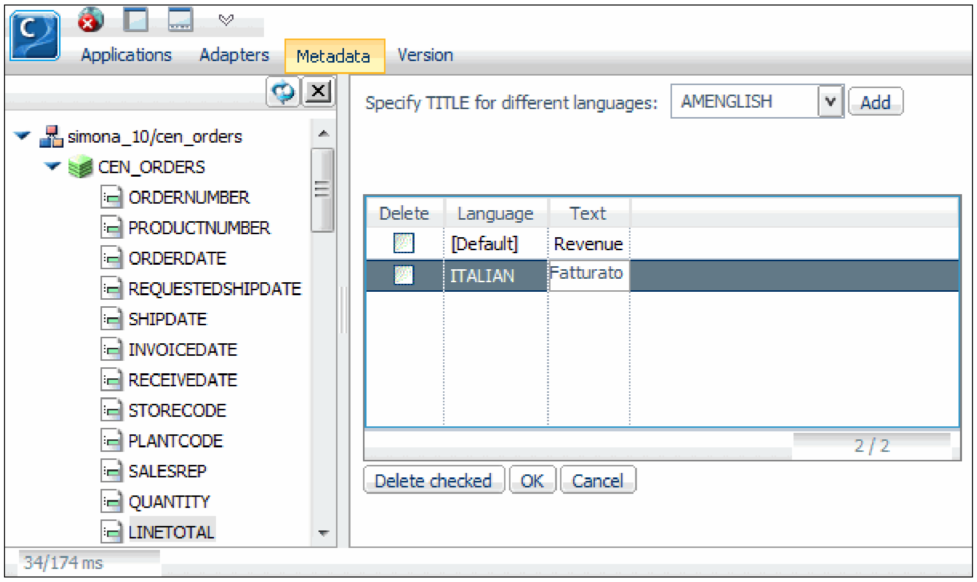
FIELDNAME	LINETOTAL
ALIAS	LINETOTAL
MISSING	✓
TITLE	Revenue
ACTUAL	P11
Type	Decimal Packed
Length	11
Decimal	
USAGE	I11CM
Type	Integer
Length	11
Options	
Negative	Default - Minus sign at the left: -6148
Comma	C - Insert: 41,376
L - Leading zeroes	☐
S - Print blank for zero	☐
% - Percent sign	☐
Currency Symbol	M - Floating $ sign

Figure 3-76 Web Interface, format field 2

5. We also want to set up the title to show up in the language preferred by each user. To do this, select the little dotted button on the side of the TITLE field (# 6 in Figure 3-76).

You are presented with a new panel (Figure 3-77) where you have to take these actions:

a. Select the language you are adding from the pull-down.
b. Select the **ADD** button to add the selected language.
c. Key in the appropriate text.
d. Repeat the process for all the languages you need.
e. Select **OK** to get back to the **PROPERTY VIEW**.



Specify TITLE for different languages: AMENGLISH Add

Delete	Language	Text
☐	[Default]	Revenue
☐	ITALIAN	Fatturato

Figure 3-77 Web interface - add multilingual titles

6. In the PROPERTY VIEW tab select APPLY to confirm your change.
7. Click the “More Options” and select SAVE to save your changes to the synonym, as shown in Figure 3-78.

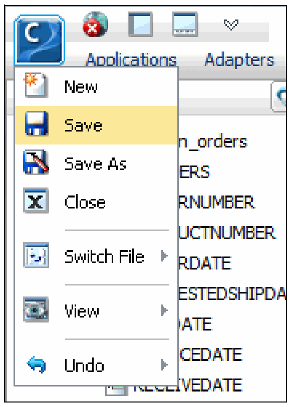


Figure 3-78 Web interface - save synonym

3.9.2 Field formatting with Developer Workbench

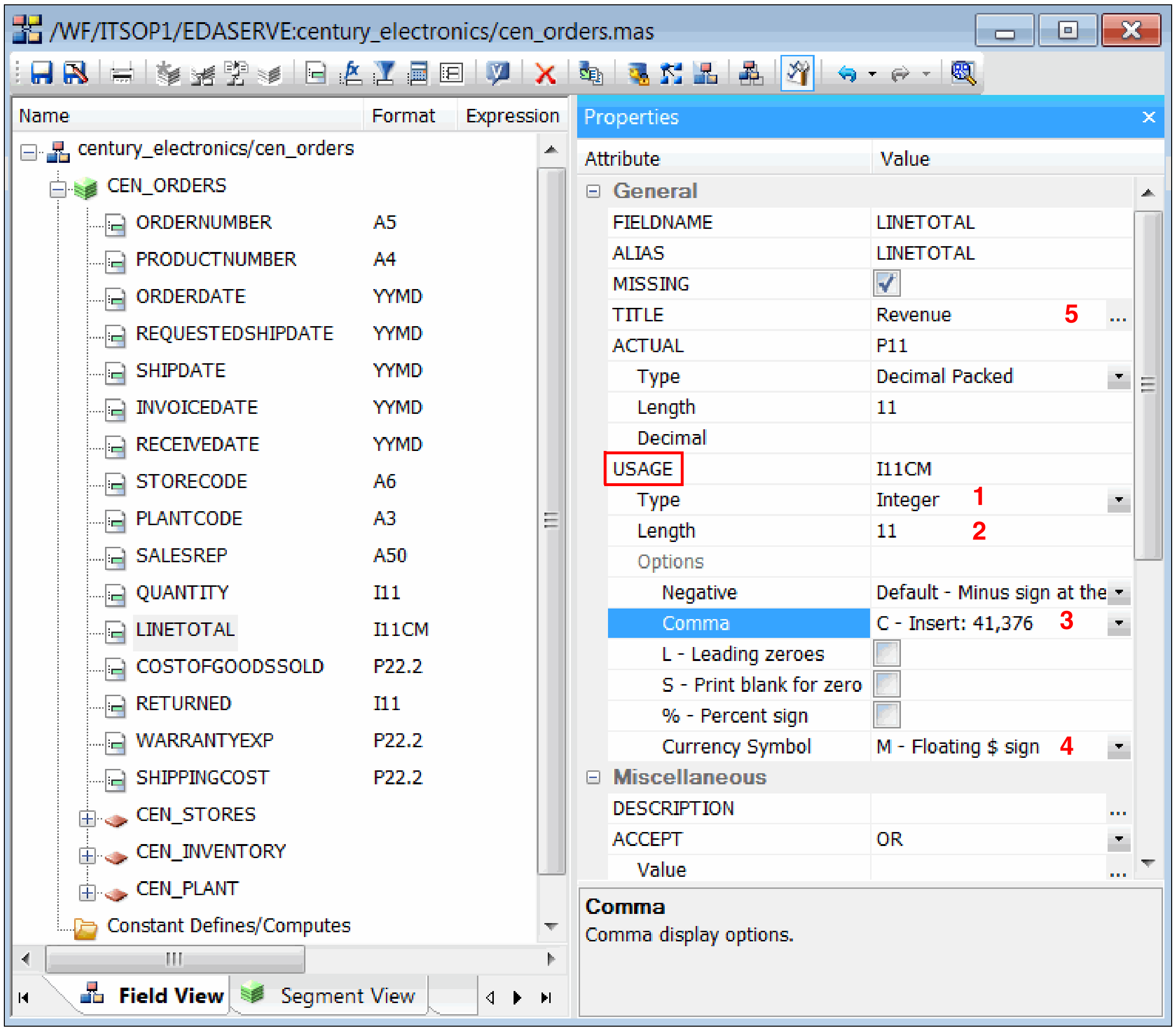
To format a field with Developer Workbench, proceed as follows:

1. Open Developer Workbench, select the system on which to work, and validate with your userid and password. Remember that the user must be part of the DevWorkbench group (use **Administration** → **Security Center** in the web interface to set up. See 4.1.2, “Security concepts” on page 150. Select **Data Servers** → **EDASERVE** → **Applications** → <***yourfolder***> and double-click the .mas file of the synonym you want to edit.

2. In your synonym, right-click the name of the field you want to format (**LINETOTAL** in this example) and select properties or just double-click it. In the PROPERTIES pane, locate the USAGE portion; see Figure 3-79. Set the field to show as an INTEGER (#1), 11 digits long (#2), then select **Comma** (#3) and **Floating $** (#4).

Important: Only change the USAGE portion; if you change the ACTUAL section, you end up with a database mismatch.

Note: The appearance of the thousands delimiter depends on your regional/national settings. Even if the caption says "comma," you will get a . as your delimiter if you have a European environmental setting.



/WF/ITSOP1/EDASERVE:century_electronics/cen_orders.mas

Name	Format	Expression
century_electronics/cen_orders		
CEN_ORDERS		
ORDERNUMBER	A5	
PRODUCTNUMBER	A4	
ORDERDATE	YYMD	
REQUESTEDSHIPDATE	YYMD	
SHIPDATE	YYMD	
INVOICEDATE	YYMD	
RECEIVEDATE	YYMD	
STORECODE	A6	
PLANTCODE	A3	
SALESREP	A50	
QUANTITY	I11	
LINETOTAL	I11CM	
COSTOFGOODSSOLD	P22.2	
RETURNED	I11	
WARRANTYEXP	P22.2	
SHIPPINGCOST	P22.2	
CEN_STORES		
CEN_INVENTORY		
CEN_PLANT		
Constant Defines/Computes		

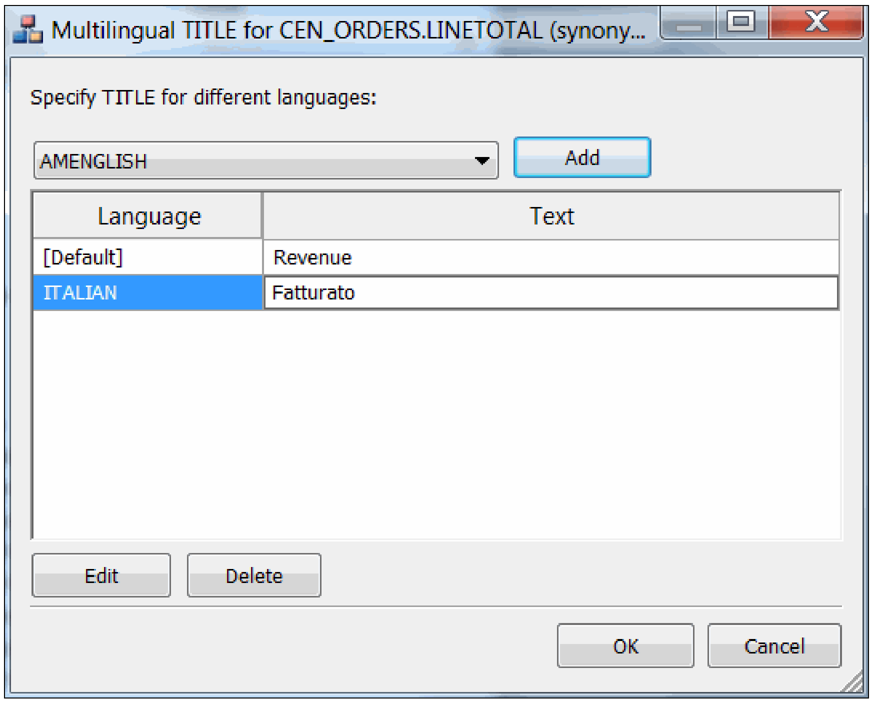
Properties

Attribute	Value
General	
FIELDNAME	LINETOTAL
ALIAS	LINETOTAL
MISSING	✓
TITLE	Revenue 5
ACTUAL	P11
Type	Decimal Packed
Length	11
Decimal	
USAGE	I11CM
Type	Integer 1
Length	11 2
Options	
Negative	Default - Minus sign at the
Comma	C - Insert: 41,376 3
L - Leading zeroes	
S - Print blank for zero	
% - Percent sign	
Currency Symbol	M - Floating $ sign 4
Miscellaneous	
DESCRIPTION	
ACCEPT	OR
Value	

Comma
Comma display options.

Figure 3-79 Developer Workbench, formatting LINETOTAL field

3. We also want to set up the title to show up in the language preferred by each user. To do this, select the little dotted button on the side of the TITLE field (# 5 in Figure 3-79). You are presented with a new panel (Figure 3-80) where you have to take these actions:
   a. Select the language you are adding from the pull-down.
   b. Select the **ADD** button to add the selected language.
   c. Key in the appropriate text.
   d. Repeat the process for all the languages you need.



Specify TITLE for different languages:

AMENGLISH

Language	Text
[Default]	Revenue
ITALIAN	Fatturato

Figure 3-80 Developer Workbench - formatting title for field LINETOTAL

4. Select **OK** to get back to the **PROPERTY VIEW**.
5. When you are done save your synonym using the Save icon in the tool bar. Close your synonym.

3.10 New fields: Define versus Compute

There are two types of calculated fields in DB2 Web Query, DEFINE and COMPUTE.

When you use the *Define field*, you add the definition to the list of fields within the table. This field is calculated every time that a record is read and selected.

When you create a *Compute* field, the field is not calculated until after the data is sorted and all aggregation is complete. Compute fields are often required for percentages and variances.

Note: A computed field is a field whose calculations are performed, not on each row that enters in the data set, but in the final step, after needed basic data has already been aggregated together. In some cases this can make a difference. Let us say we want to calculate a percentage on revenue by country. To calculate the percentage on each LINETOTAL row and then sum each percentage is very different than summing all rows by country and then calculating each country percentage.

Both type of fields can be created in the synonym to gain these benefits:

- Ensure consistency across reports
- Shield report developers from data definition complexity
- Make report development quicker and easier

New fields such as these can accomplish very sophisticated and complex tasks:

- Arithmetic operators (+, -, *, /)
- Concatenation (| or ||)
- Comparison operators (EQ, NE, LE, GE, LT, GT)
- Logic operators (IF, THEN, ELSE, AND, OR, NOT)
- Upper/Lower case (a $\rightarrow$ A, A $\rightarrow$ a)
- Date / Datetime
- Transformation and decoding functions provided by DB2 Web Query (full documentation available in the products manual, *Using Functions,* downloadable from the DB2 Web Query wiki page

  References:

  - DB2 Web Query wiki website:

    `https://www.ibm.com/developerworks/mydeveloperworks/wikis/home/wiki/W516d8b60d32c_4fc5_a811_5f3d840bf524?lang=en`

  - *Using Functions* manual:

    `https://www.ibm.com/developerworks/mydeveloperworks/wikis/home?lang=en#/wiki/W516d8b60d32c_4fc5_a811_5f3d840bf524/page/Functions`

- System functions (current date and time, system variables, current user...)

Though new calculated fields (defined and computed) may be created at the report level, we strongly encourage you to exploit the power of metadata by creating them in synonyms. Moreover, some of these listed functionalities may only be available when using the Developer Workbench to create a new field.

Generally speaking either the web interface or the Developer Workbench tool can be used to create Define and Compute fields, though the two interface can have different capabilities in some cases. Here we present both ways of doing things, for your reference.

3.10.1 Define field: Web interface

We want a new PROFIT field, that calculates (LINETOTAL - COSTOFGOODSOLD). This will be used in many reports, hence it is worth while creating it once in metadata and making it available for all development needs.

To create a new Define field in the web interface, proceed as follows:

1. In the web interface select the folder where the synonym is located, right-click and select **Metadata** → **Edit**.
2. Locate the synonym you want to edit in the list, right-click it, and select **Open** as shown in Figure 3-81.

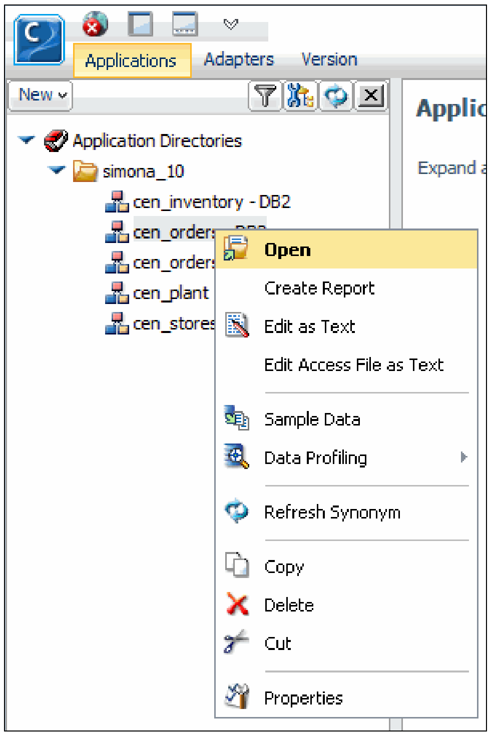


Figure 3-81 Open synonym with web interface

3. You are presented with a panel with all the existing fields. Right-click the segment name and select **Insert** → **Define** (Figure 3-82). Or, alternatively, go to the end of the list of fields and click the little folder icon you find in it that says `Constant Define/Compute to create a new field`.

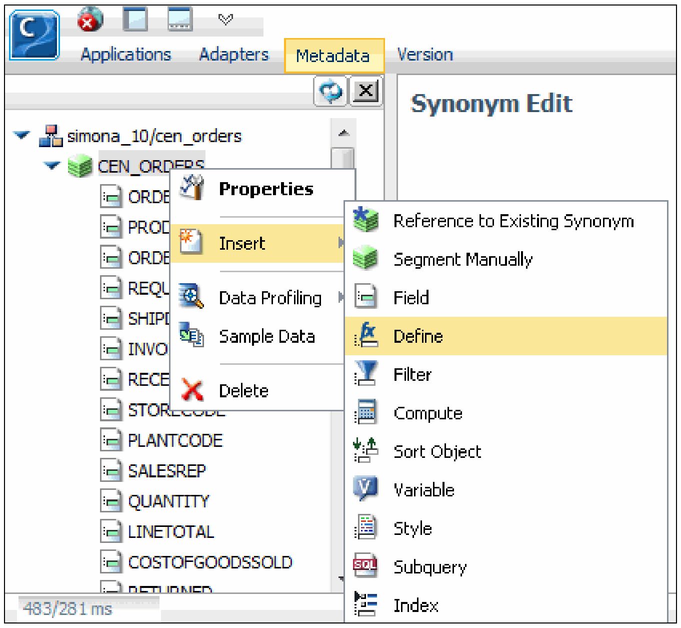
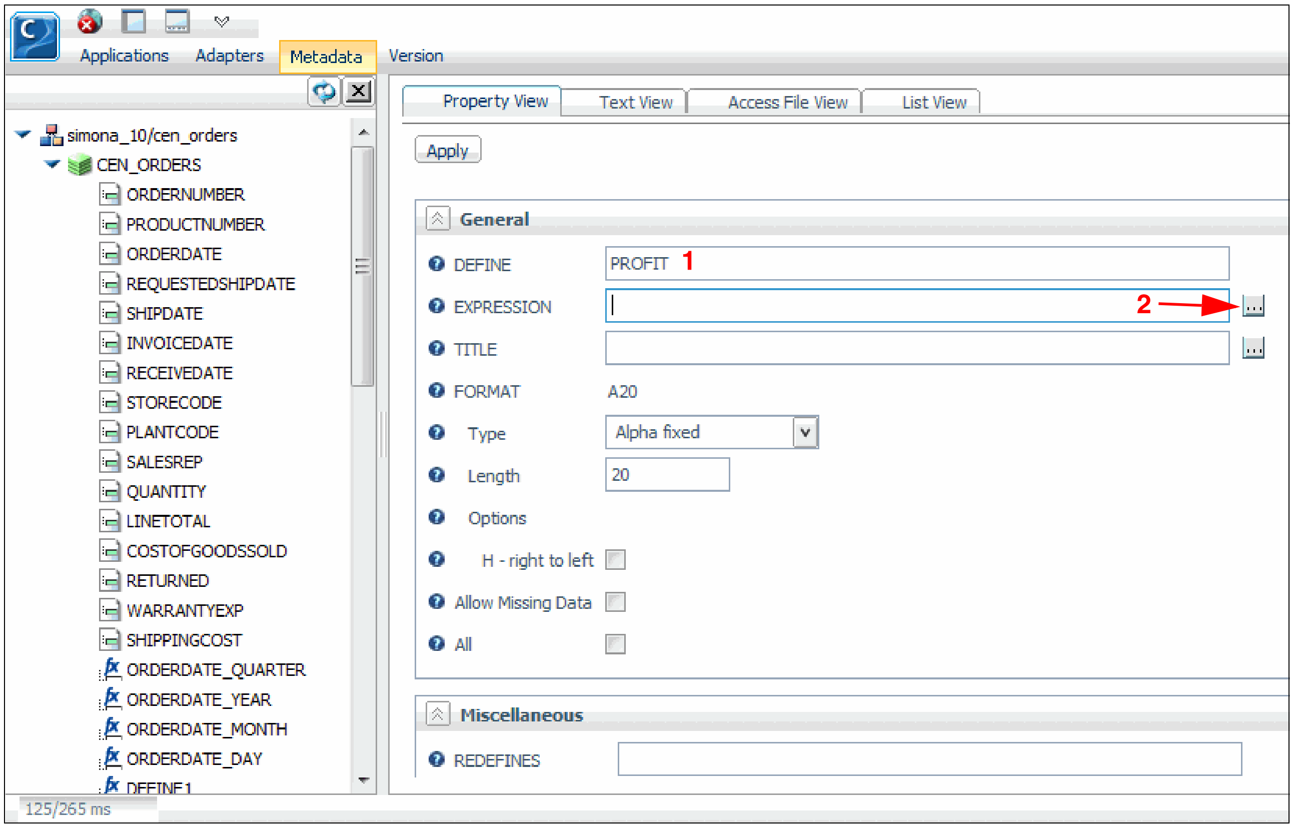


Figure 3-82 Create a new Define field - web interface - Step 1

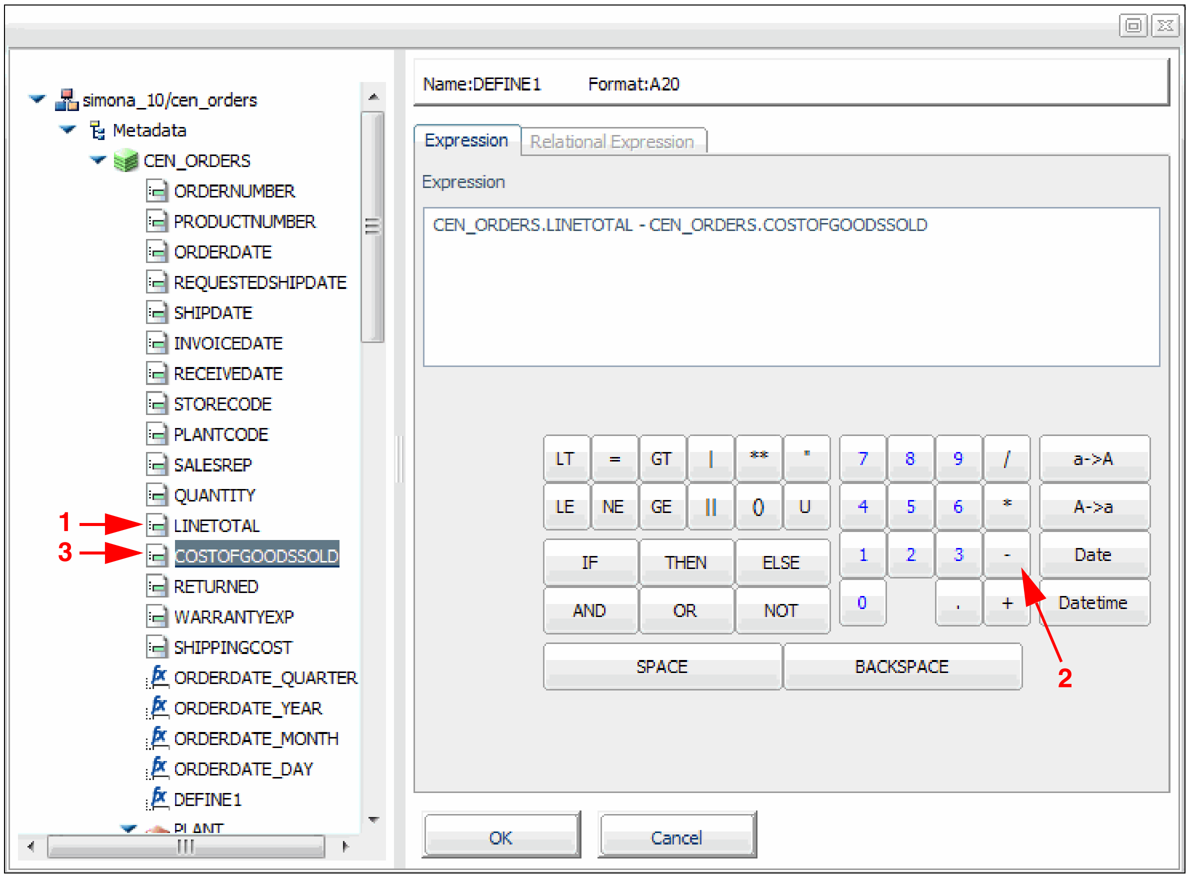
4. Type the desired field name in “Define”, here we use **PROFIT** (see #1), and then click the little dotted button on the right of “Expression” (see **#2**) as shown in Figure 3-83.



DEFINE	PROFIT
EXPRESSION	
TITLE	
FORMAT	A20
Type	Alpha fixed
Length	20
Options	
H - right to left	☐
Allow Missing Data	☐
All	☐
REDEFINES	

Figure 3-83 Define field

5. You are presented with a “calculator” where you can set up your formula (see Figure 3-84). Drag the LINETOTAL (#1) field in the expression box, click the - symbol (#2), then drag the COSTOFGOODSSOLD (#3) field in the expression box. Click **OK**.

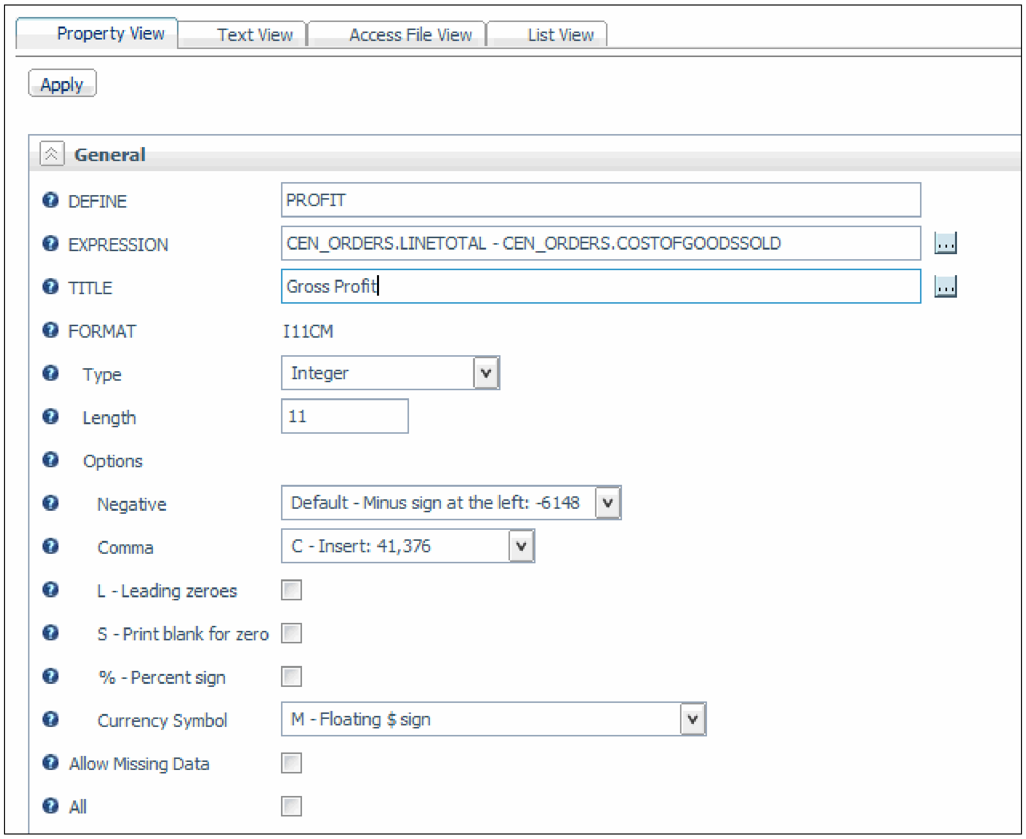

Name:DEFINE1 Format:A20

Expression

CEN_ORDERS.LINETOTAL - CEN_ORDERS.COSTOFGOODSSOLD

Figure 3-84 Define Field calculator

6. Finish defining and formatting your field. Select **Apply** when you are done. See the online help for the various formatting capabilities. In this example (Figure 3-85 on page 96) to get a title, numeric field, 11 digits long with thousands separator and the currency symbol we have used:
   a. Title - Gross Profit
   b. Type - Integer
   c. Length - 11
   d. Comma - C
   e. Currency symbol - M



DEFINE	PROFIT
EXPRESSION	CEN_ORDERS.LINETOTAL - CEN_ORDERS.COSTOFGOODSSOLD
TITLE	Gross Profit
FORMAT	I11CM
Type	Integer
Length	11
Options	
Negative	Default - Minus sign at the left: -6148
Comma	C - Insert: 41,376
L - Leading zeroes	☐
S - Print blank for zero	☐
% - Percent sign	☐
Currency Symbol	M - Floating $ sign
Allow Missing Data	☐
All	☐

Figure 3-85 Formatting options

7. You will notice that your new field is added to the list of the available fields for the segment (Figure 3-86).

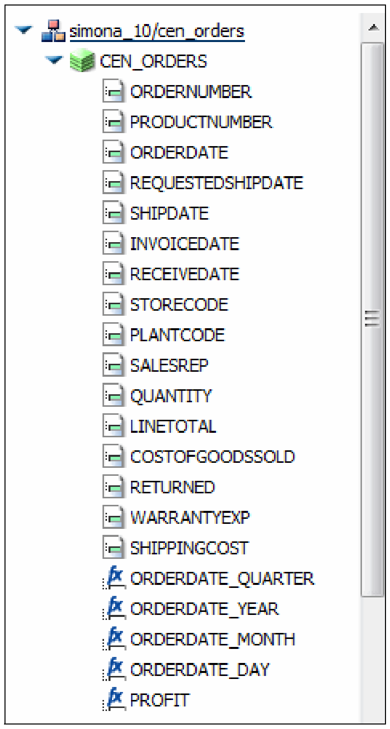


Figure 3-86 Fields list

8. Right-click the "Other options" icon and select **Save** as shown in Figure 3-87.

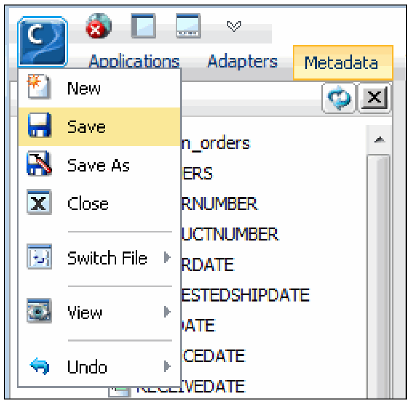


Figure 3-87 Save synonym

3.10.2 Define field: Developer Workbench

We want a new PROFIT field, that calculates (LINETOTAL - COSTOFGOODSOLD). This will be used in many reports, hence it is worth while creating it once in metadata and making it available for all development needs.

To create a new Define field, with Developer Workbench, proceed as follows:

1. Open Developer Workbench, select the system on which to work, and validate with your userid and password. Remember that the user must be part of the DevWorkbench group (use **Administration** → **Security Center** in the web interface to set up; see Chapter 4, “Security Center” on page 149). Select **Data Servers** → **EDASERVE** → **Applications** → <***yourfolder***> and double-click the .mas file of the synonym you want to edit.
2. In your synonym, right-click the segment name (**ORDERS** in this example) and select **Insert** → **Define** as shown in Figure 3-88. Another way of doing this would be selecting the *fx - Define* icon from the tool bar.

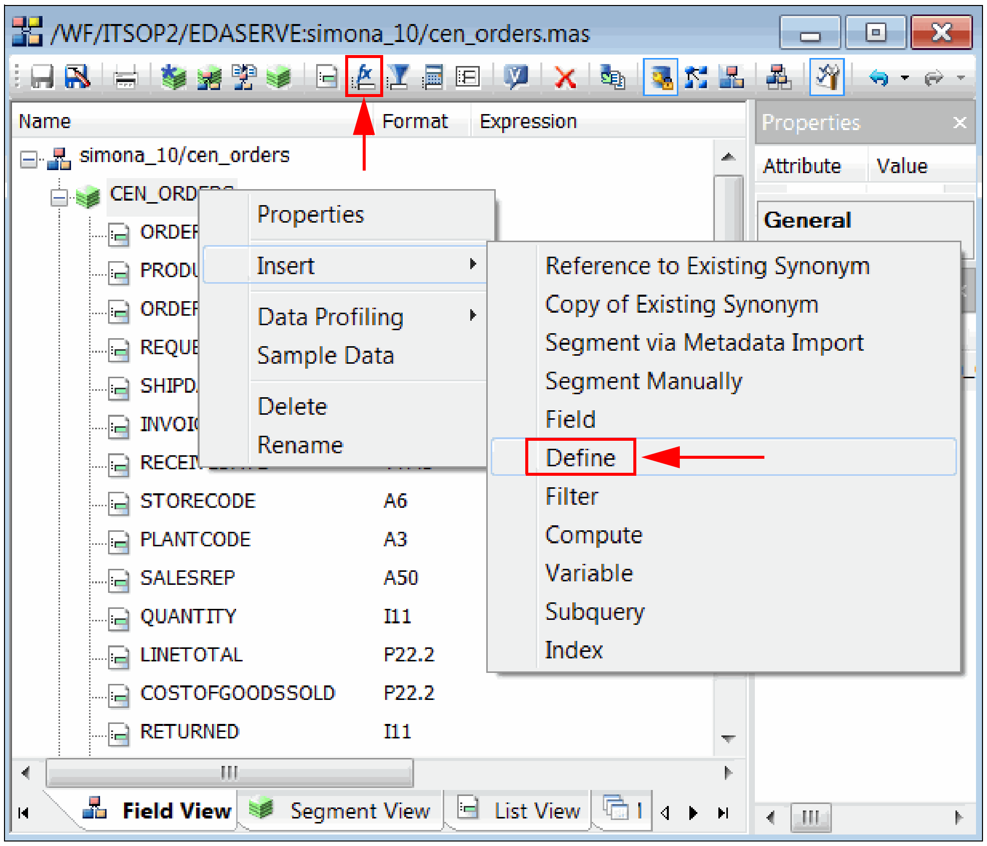
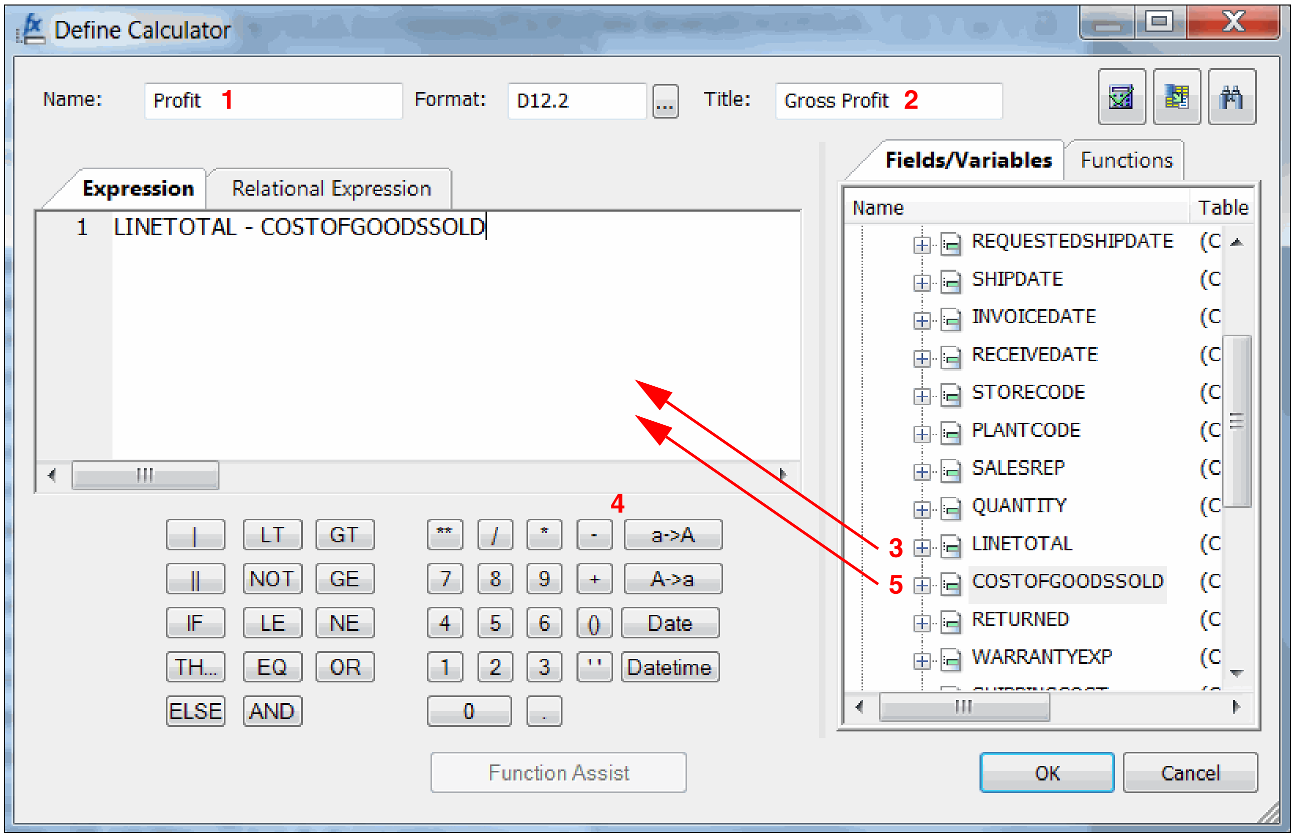


Figure 3-88 Define Field, Developer Workbench

3. You are presented with the Define Calculator (Figure 3-89). Type in the field name (#1) and a title (#2). Drag (or double-click) the LINETOTAL field to the calculator area (#3), click the - symbol (#4), and drag the COSTOFGOODSOLD filed to the calculator (#5).


Define Calculator

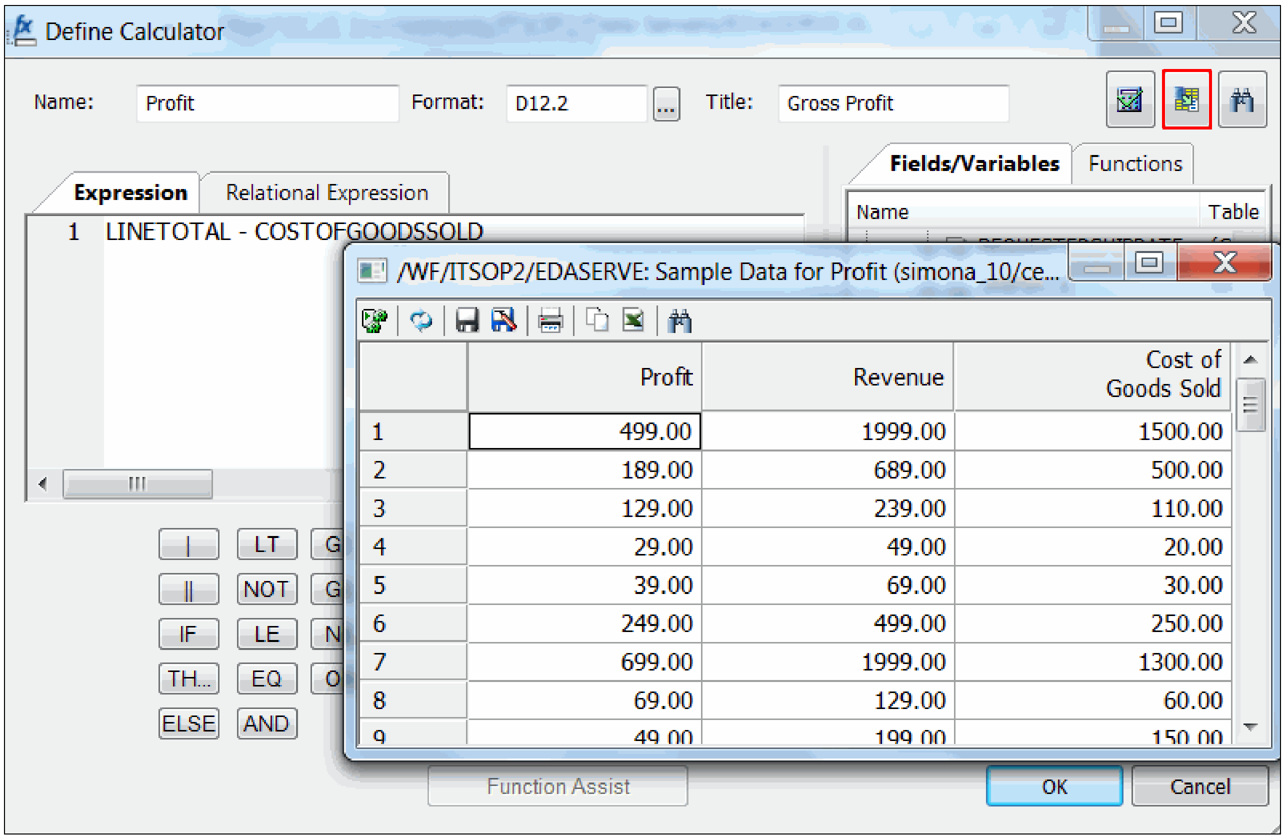
Name: Profit 1 Format: D12.2 Title: Gross Profit 2

Expression | Relational Expression

```
LINETOTAL - COSTOFGOODSSOLD
```

Figure 3-89 Developer Workbench - Define Calculator

4. Now you can check the data resulting from your formula using the **Sample Data** button on the top right side of the pane as shown in Figure 3-90.


Define Calculator

Name: Profit Format: D12.2 Title: Gross Profit

1 LINETOTAL - COSTOFGOODSSOLD

/WF/ITSOP2/EDASERVE: Sample Data for Profit (simona_10/ce...

	Profit	Revenue	Cost of Goods Sold
1	499.00	1999.00	1500.00
2	189.00	689.00	500.00
3	129.00	239.00	110.00
4	29.00	49.00	20.00
5	39.00	69.00	30.00
6	249.00	499.00	250.00
7	699.00	1999.00	1300.00
8	69.00	129.00	60.00
9	49.00	199.00	150.00

Figure 3-90 Developer Workbench - Define Calculator - Sample data

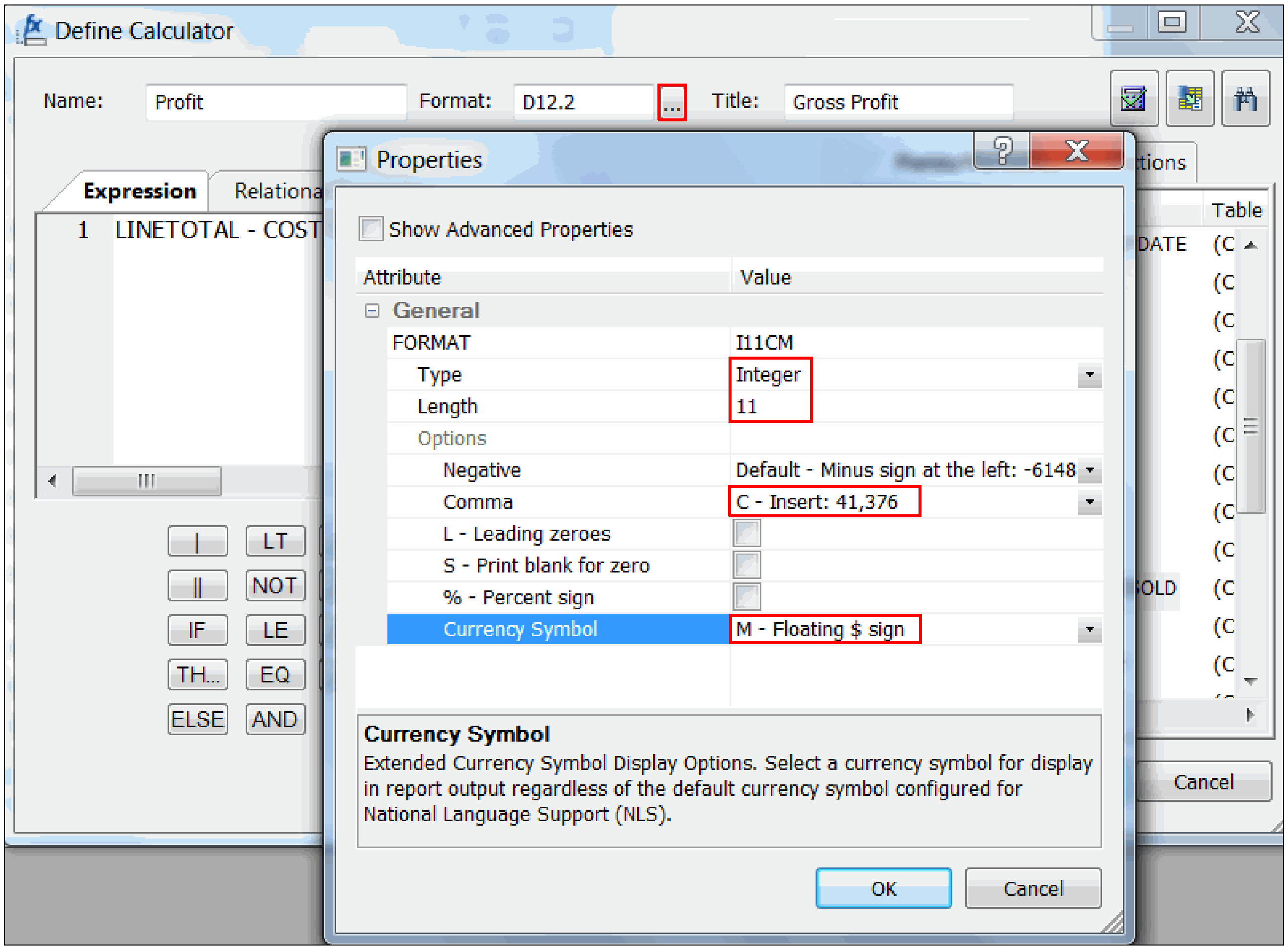
5. You may now want to format your field. Click the dotted button next to the format field and you will be presented with the formatting pane (Figure 3-91); select **OK** when you are done. In this example, to get a numeric field, 11 digits long with thousands separator and the currency symbol, we have used the following values:

   a. Type - Integer

   b. Length - 11

   c. Comma - C

   d. Currency symbol - M


Define Calculator

Name: Profit Format: D12.2 Title: Gross Profit

Expression

1 LINETOTAL - COST

Properties

Show Advanced Properties

Attribute	Value
General	
FORMAT	I11CM
Type	Integer
Length	11
Options	
Negative	Default - Minus sign at the left: -6148
Comma	C - Insert: 41,376
L - Leading zeroes	
S - Print blank for zero	
% - Percent sign	
Currency Symbol	M - Floating $ sign

Currency Symbol
Extended Currency Symbol Display Options. Select a currency symbol for display in report output regardless of the default currency symbol configured for National Language Support (NLS).

Figure 3-91 Developer Workbench - Define Calculator - Format options

6. Select **OK** to close the Define Calculator and save your synonym.

Developer Workbench provides you with more options than the web interface. Notice the Function tab in the Define Calculator, as shown in Figure 3-92. From there, you can get to all of the DB2 Web Query functions. For documentation, see the *Using Functions* manual, downloadable for the DB2 Web Query wiki.

References:

- DB2 Web Query wiki website:

  https://www.ibm.com/developerworks/mydeveloperworks/wikis/home/wiki/W516d8b60d32c_4fc5_a811_5f3d840bf524?lang=en

- The *Using Functions* manual:

  https://www.ibm.com/developerworks/mydeveloperworks/wikis/home?lang=en#/wiki/W516d8b60d32c_4fc5_a811_5f3d840bf524/page/Functions

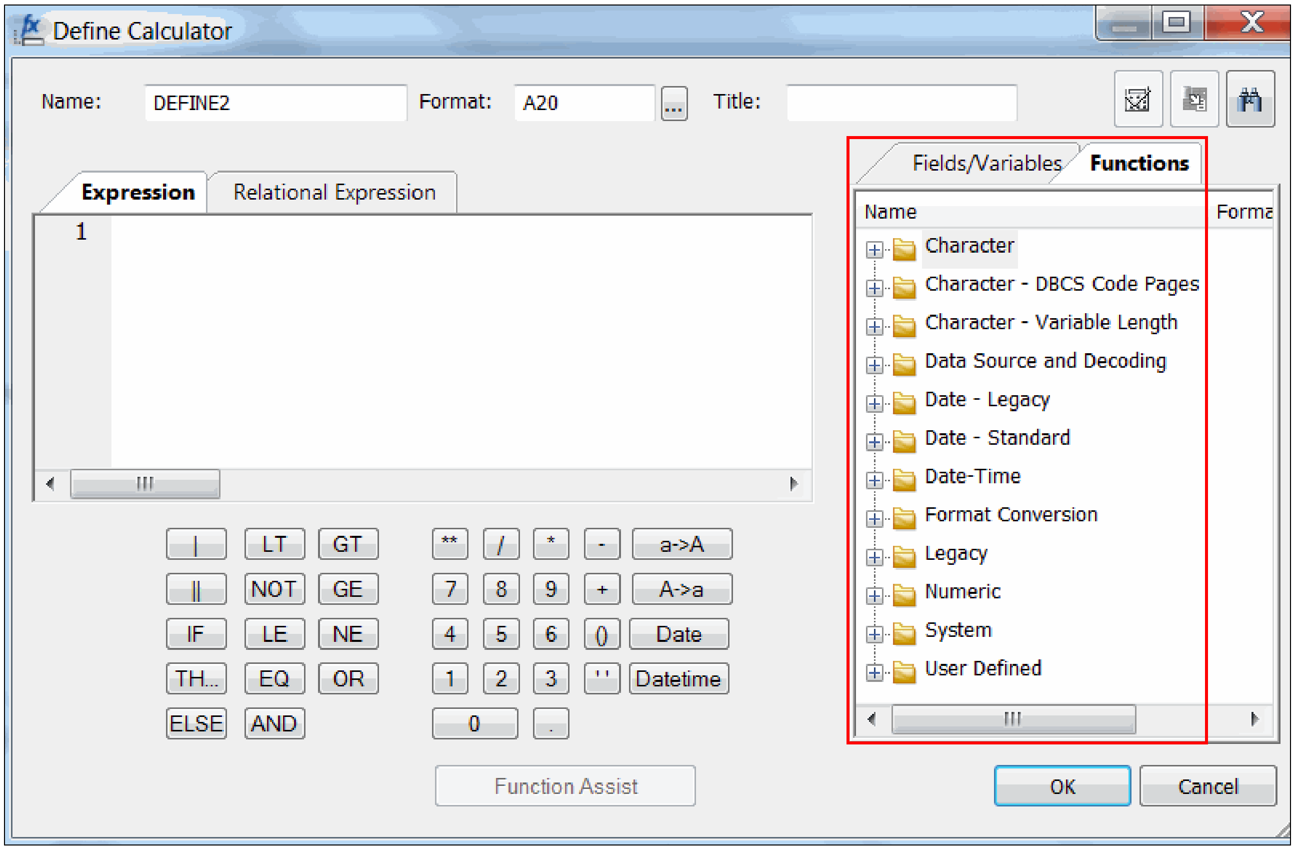


Figure 3-92 Developer Workbench - functions

3.10.3 Compute field: Web interface

As stated, a computed field is a field whose calculations are performed not on each row that enters in the data set, but in the final step, after needed basic data have already been aggregated together.

We now want to create a new MARGIN field, to calculate the gross profit margin, expressed as (((LINETOTAL - COSTOFGOODSSOLD) / COSTOFGOODSSOLD) *100). In the previous section, we have defined a new PROFIT field (LINETOTAL - COSTOFGOODSSOLD) and we can now use it in our PROFIT definition: ((PROFIT / COSTOFGOODSSOLD) *100).

Notes:

- The order in which things are created is important to DB2 Web Query.
- If you want to reference field XX in field YY, XX **MUST** have already been described when you describe YY or YY will not work.

To create a new Compute field in the web interface, proceed as follows:

1. Right-click folder Century Electronics and select **Metadata** → **Edit** as shown in Figure 3-93.

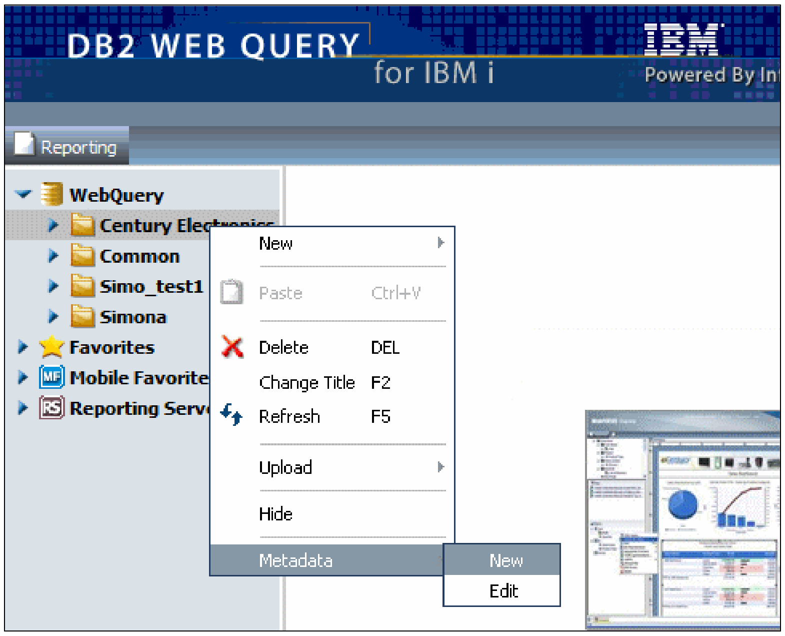


Figure 3-93 Web Interface - Edit Metadata

2. Right-click **cen_orders** and select **Open** (Figure 3-94).

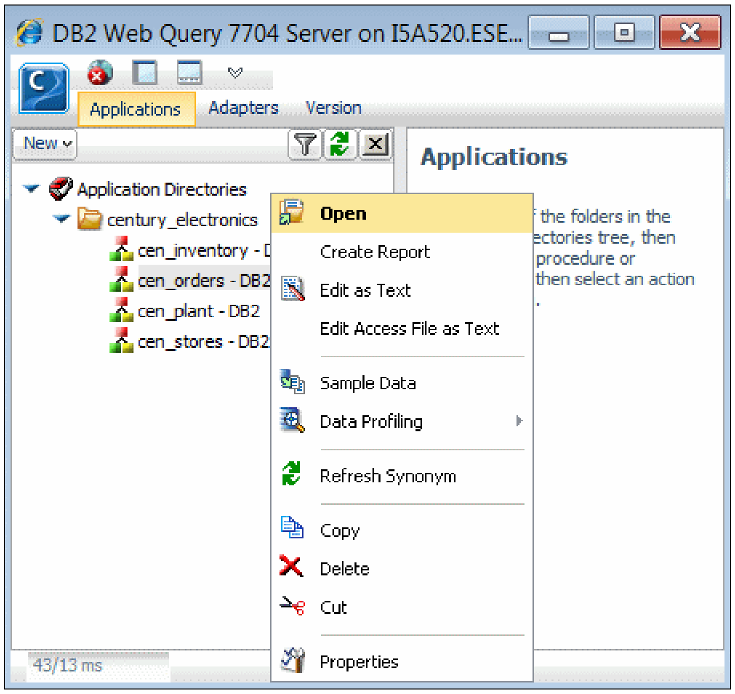


Figure 3-94 Web Interface - open metadata

3. The synonym editor opens up. Right-click segment **ORDERS** and select **Insert** → **Compute** (Figure 3-95).

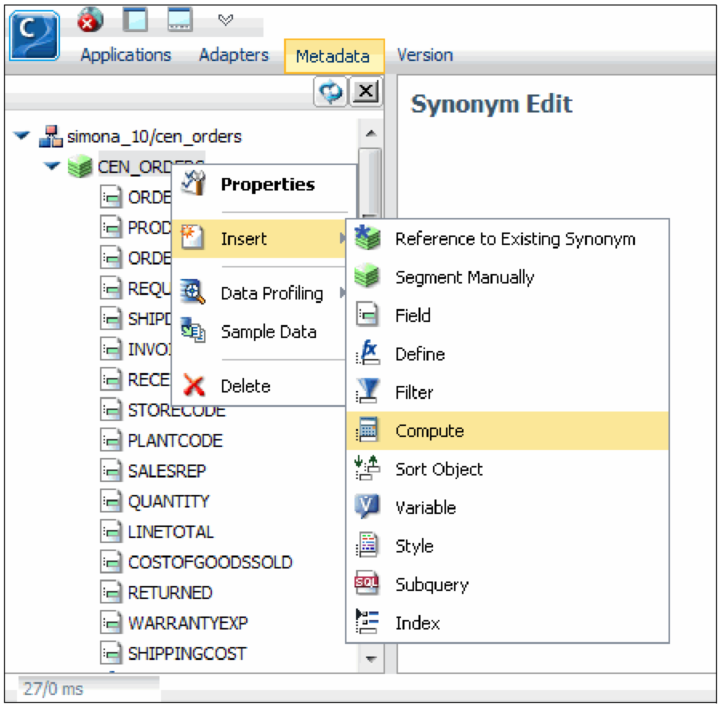
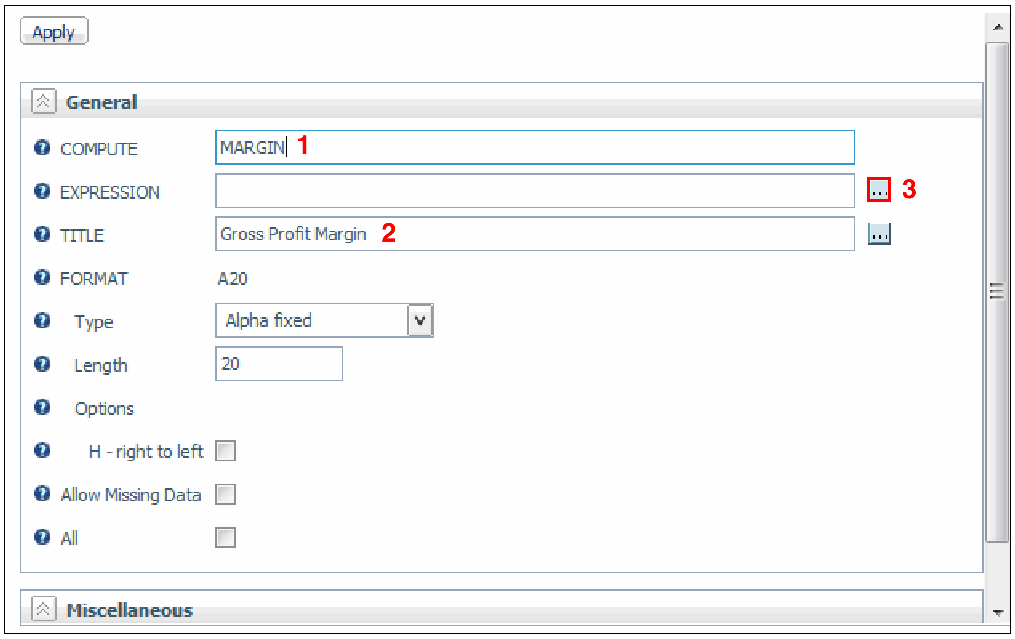


Figure 3-95 Web Interface - Insert Compute field

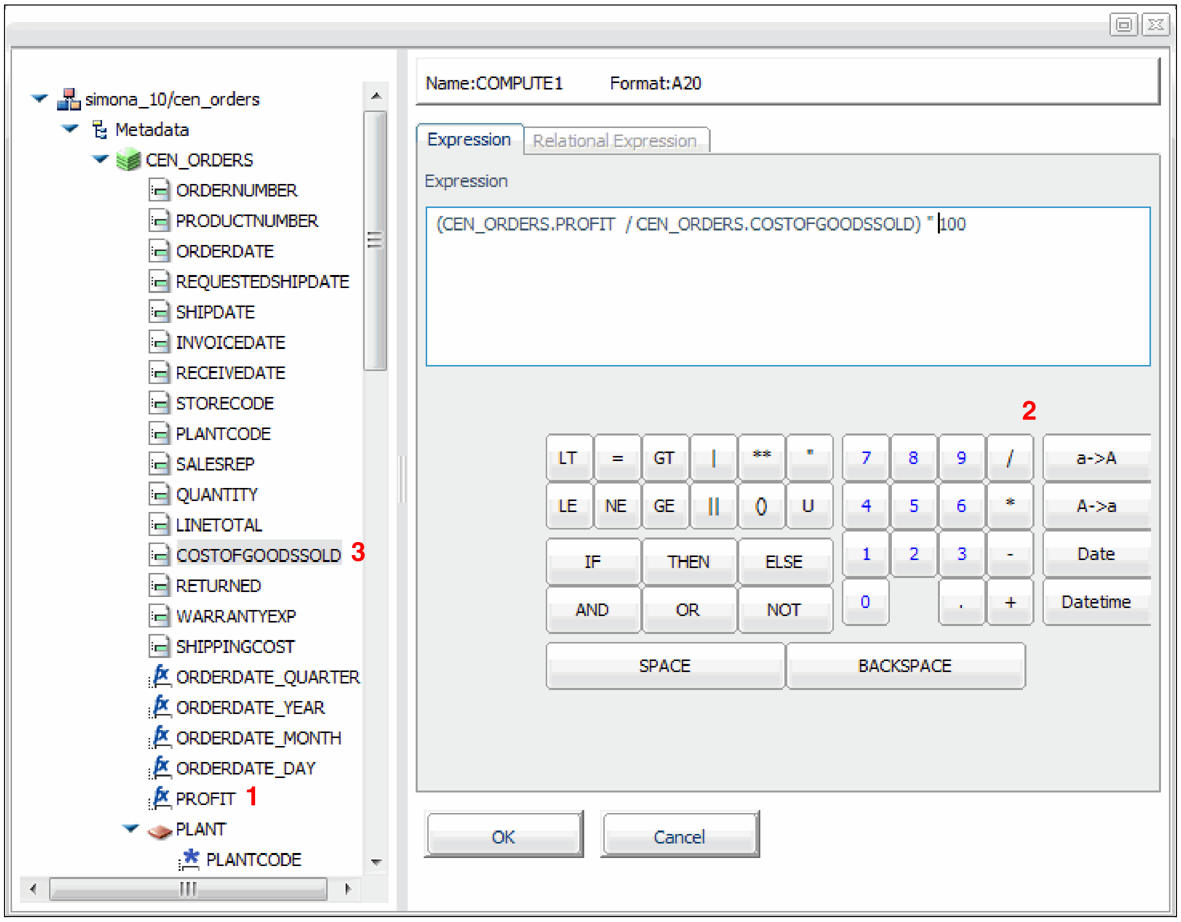
4. You are presented with the Synonym editor, Figure 3-96. Type in a name and a title for the new field, **MARGIN** (1#) and **Gross Profit Margin** (2#) in this example, then click the dotted button on the "Expression" field.



COMPUTE	MARGIN
EXPRESSION	
TITLE	Gross Profit Margin
FORMAT	A20
Type	Alpha fixed
Length	20
Options	
H - right to left	☐
Allow Missing Data	☐
All	☐

Figure 3-96 Web interface - Compute field

5. You are presented with a calculator where you can input the formula for your new field. Double-click the field you need to get in the calculator or just type in it. Usually it is better to select fields to prevent typos. Scroll the list until you get at the **PROFIT** field (#1), double-click it to select, double-click the / symbol (#2), double-click **COSTFGOODSSOLD** (#3) then type in ***100** as shown in Figure 3-97. When you are done, select **OK**.

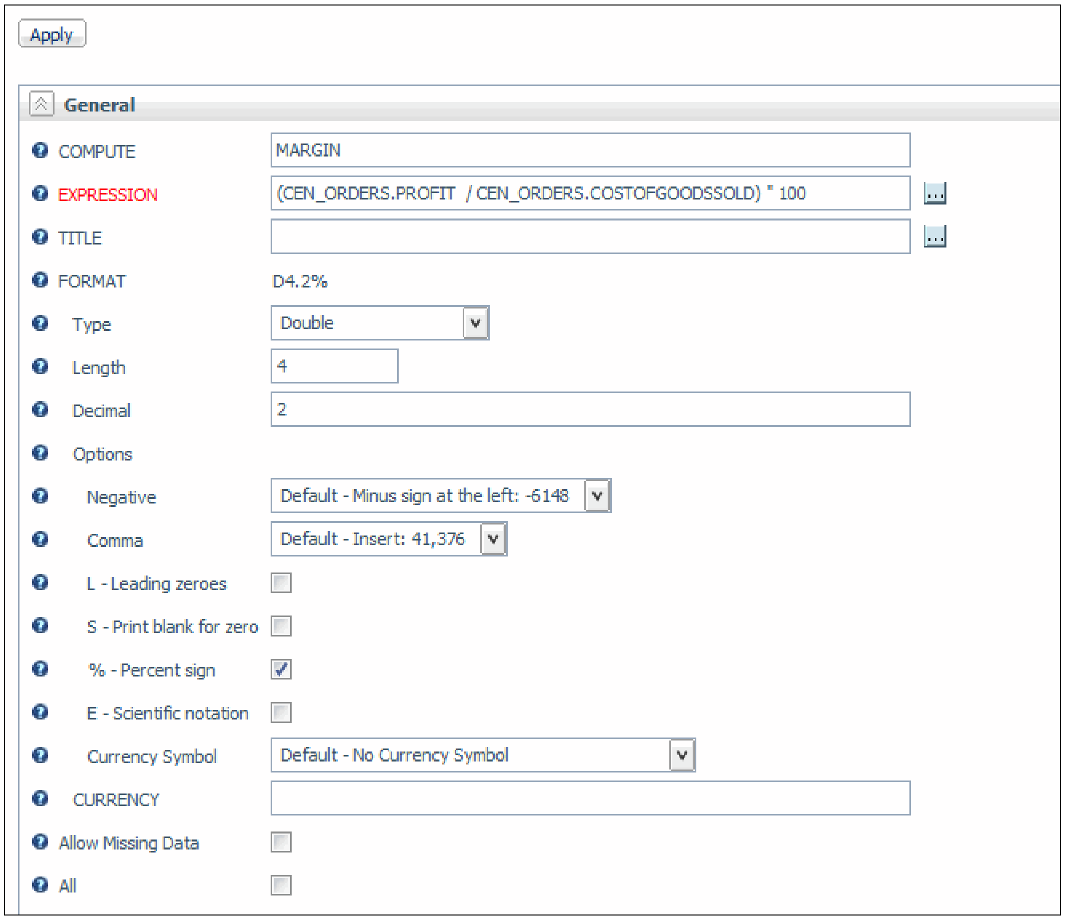

Name:COMPUTE1 Format:A20

Expression

(CEN_ORDERS.PROFIT / CEN_ORDERS.COSTOFGOODSSOLD) * 100

Figure 3-97 Web Interface - Compute field calculator

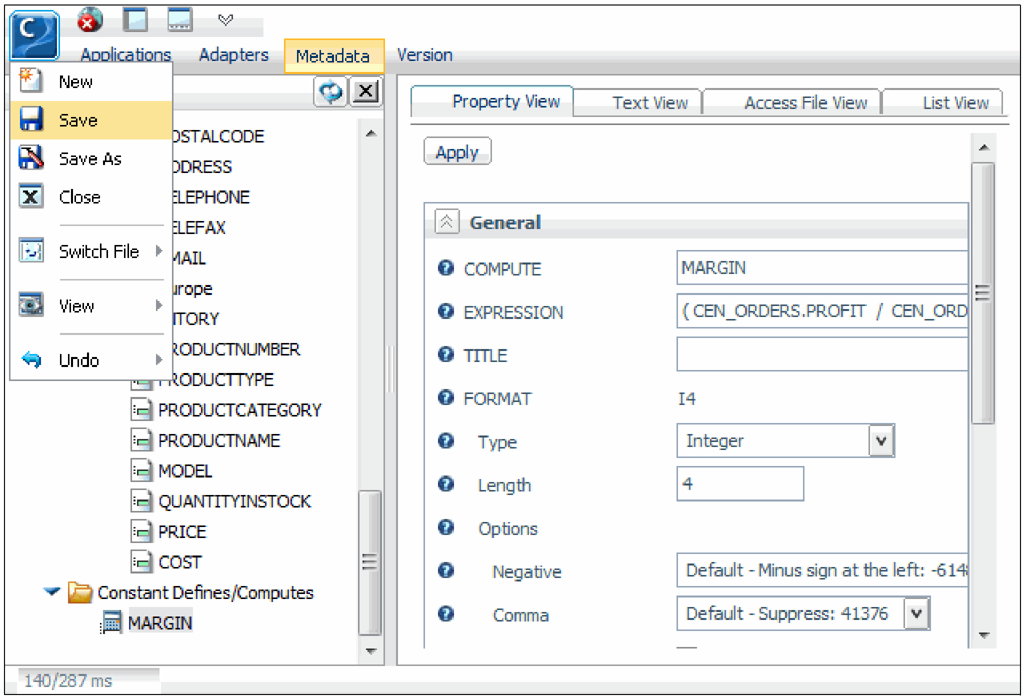
6. Make sure the format options are consistent with the data you are handling, change type, length and any other options according to needs as shown in Figure 3-98. Here we have also selected the % sign to be shown in our field. When you are done, select **Apply**.



COMPUTE	MARGIN
EXPRESSION	(CEN_ORDERS.PROFIT / CEN_ORDERS.COSTOFGOODSSOLD) * 100
TITLE	
FORMAT	D4.2%
Type	Double
Length	4
Decimal	2
Options	
Negative	Default - Minus sign at the left: -6148
Comma	Default - Insert: 41,376
L - Leading zeroes	☐
S - Print blank for zero	☐
% - Percent sign	✓
E - Scientific notation	☐
Currency Symbol	Default - No Currency Symbol
CURRENCY	
Allow Missing Data	☐
All	☐

Figure 3-98 Web Interface - Format Compute field

7. Your new Compute field will appear in the Constant Defines/Computes folder at the bottom of the list. Selecting it, you will be presented with its properties in the right-hand side of the pane. Select the “**Other options**” icon and **Select**, as shown in Figure 3-99.



COMPUTE	MARGIN
EXPRESSION	(CEN_ORDERS.PROFIT / CEN_ORD
TITLE	
FORMAT	I4
Type	Integer
Length	4
Options	
Negative	Default - Minus sign at the left: -614
Comma	Default - Suppress: 41376

Figure 3-99 Web Interface - Save synonym

3.10.4 Compute field: Developer Workbench

To create a new Compute field with Developer Workbench, proceed as follows:

1. Open Developer Workbench, select the system on which to work, and validate with your userid and password. Remember that the user must be part of the DevWorkbench group (use **Administration** → **Security Center** in the web interface to set up; see Chapter 4, "Security Center" on page 149). Select **Data Servers** → **EDASERVE** → **Applications** → ***<yourfolder>*** and double-click the .mas file of the synonym you want to edit.
2. In your synonym, right-click the segment name (ORDERS in this example) and select **Insert** → **Compute** as shown in Figure 3-100. Alternatively, you can select the little calculator icon (1) in the tool bar.

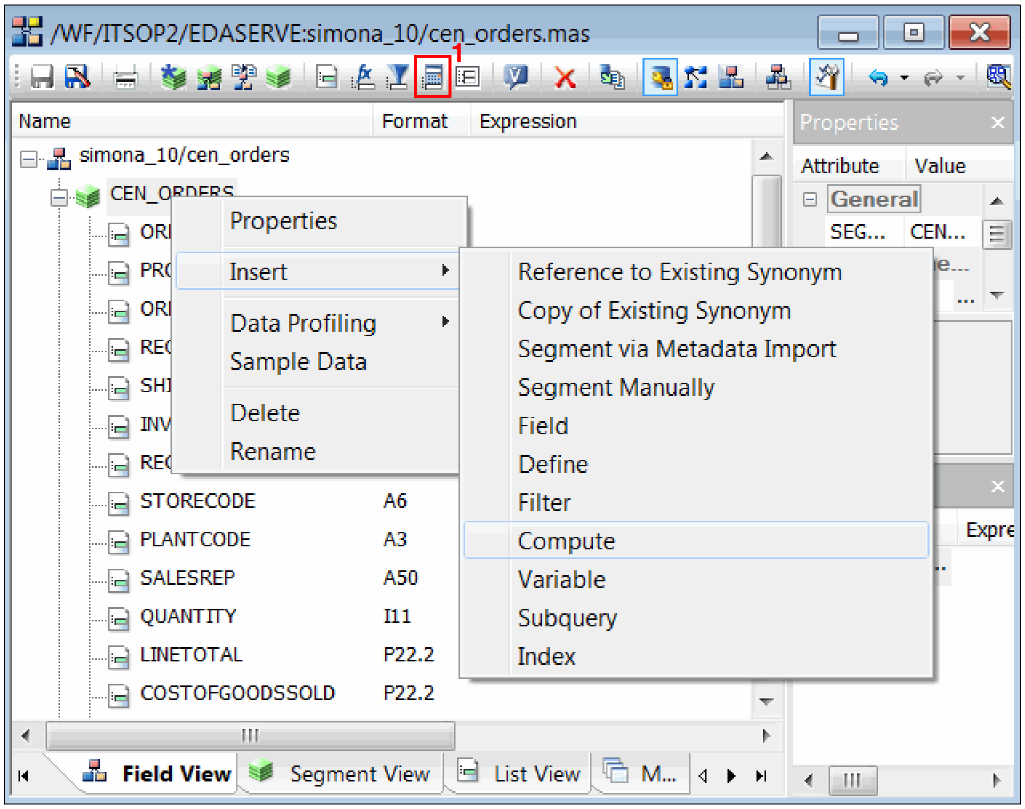
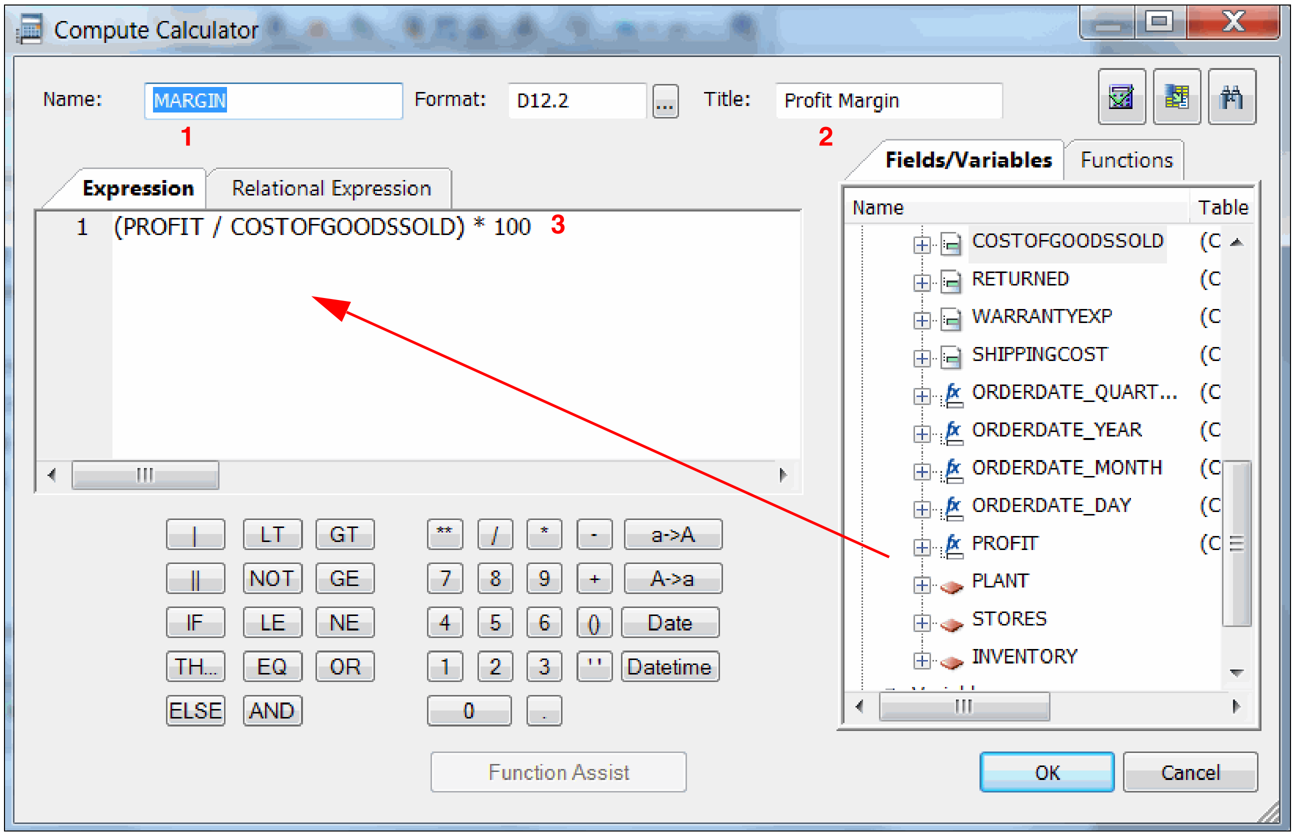


Figure 3-100 Developer Workbench - Insert Compute field

3. You are presented with the Compute Calculator. Type in a name (#1) and title (#2) for the field (here we use **MARGIN** and **Gross Profit Margin**) and fill in the desired formula ((PROFIT / COSTOFGOODSSOLD) *100) (#3); this can be done typing or selecting items (we encourage you to select items, to avoid typos). Notice the “Function” tab, see 3.10.2, “Define field: Developer Workbench” on page 98 for more information.

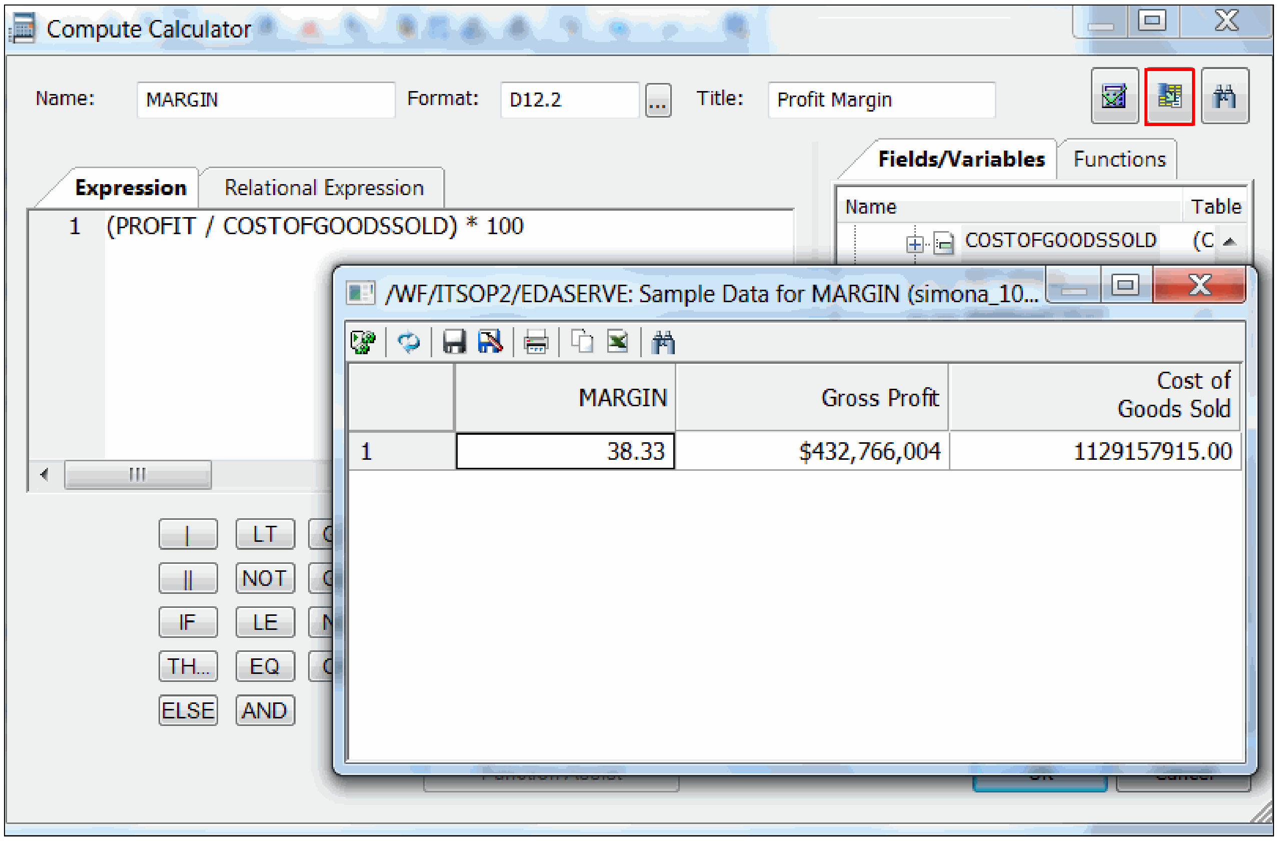


Name: MARGIN
Format: D12.2
Title: Profit Margin

```
(PROFIT / COSTOFGOODSSOLD) * 100
```

Figure 3-101 Developer Workbench - Compute Calculator

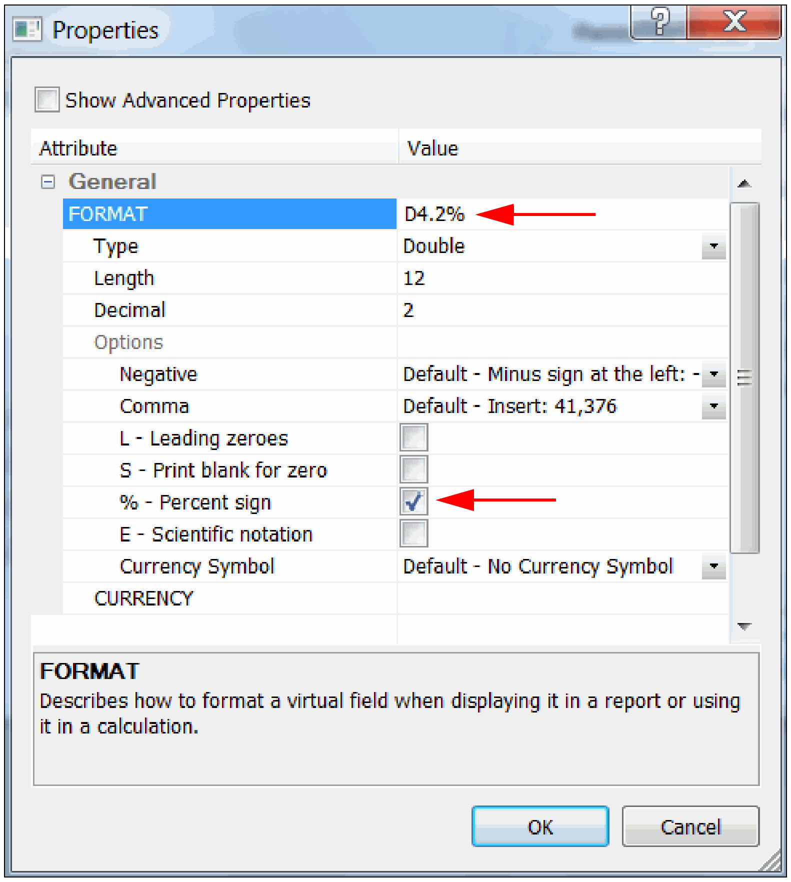
4. Using the Sample Data button, you can check what is the result of your new Compute field, as shown in Figure 3-102.



	MARGIN	Gross Profit	Cost of Goods Sold
1	38.33	$432,766,004	1129157915.00

Figure 3-102 Developer Workbench - Compute Calculator sample data

5. Selecting the little dotted button on side of Format, you get the formatting pane. You can notice in Figure 3-103 that we have asked for the % symbol to be posted in our new field. Select **OK** to confirm your choices.

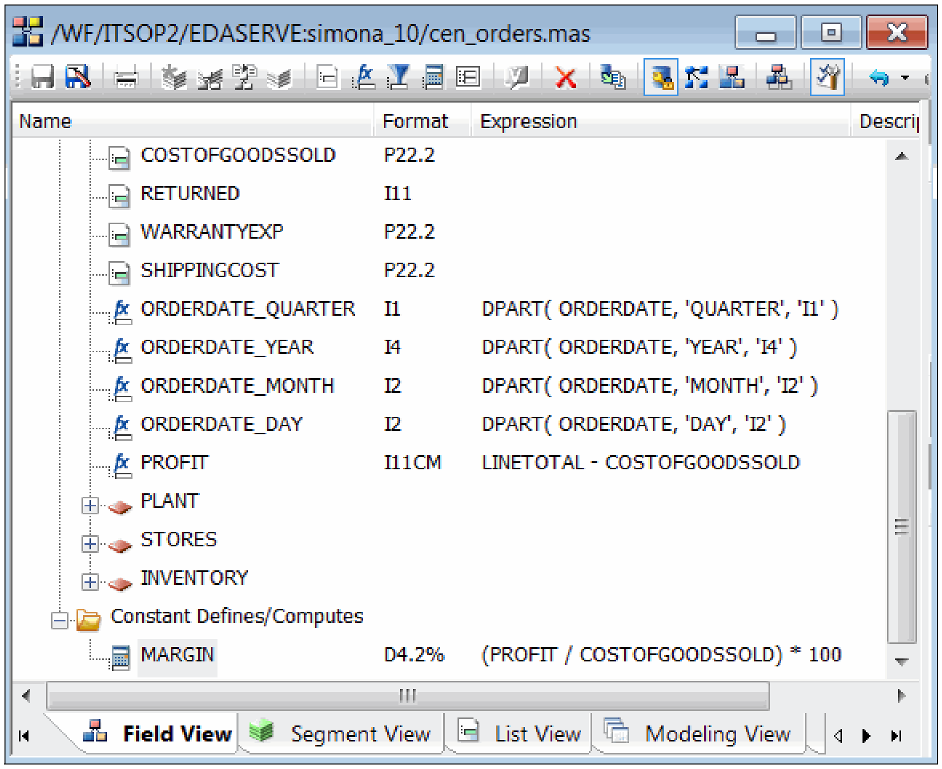


Attribute	Value
General	
FORMAT	D4.2%
Type	Double
Length	12
Decimal	2
Options	
Negative	Default - Minus sign at the left: -
Comma	Default - Insert: 41,376
L - Leading zeroes	☐
S - Print blank for zero	☐
% - Percent sign	✓
E - Scientific notation	☐
Currency Symbol	Default - No Currency Symbol
CURRENCY	

FORMAT
Describes how to format a virtual field when displaying it in a report or using it in a calculation.

Figure 3-103 Developer Workbench - format Compute field

6. You will be brought back to the Compute Calculator, select **OK** to confirm. On the synonym editor (Figure 3-104), use the Save icon to save your synonym and close it.



Name	Format	Expression
COSTOFGOODSSOLD	P22.2	
RETURNED	I11	
WARRANTYEXP	P22.2	
SHIPPINGCOST	P22.2	
ORDERDATE_QUARTER	I1	DPART(ORDERDATE, 'QUARTER', 'I1')
ORDERDATE_YEAR	I4	DPART(ORDERDATE, 'YEAR', 'I4')
ORDERDATE_MONTH	I2	DPART(ORDERDATE, 'MONTH', 'I2')
ORDERDATE_DAY	I2	DPART(ORDERDATE, 'DAY', 'I2')
PROFIT	I11CM	LINETOTAL - COSTOFGOODSSOLD
PLANT		
STORES		
INVENTORY		
Constant Defines/Computes		
MARGIN	D4.2%	(PROFIT / COSTOFGOODSSOLD) * 100

Figure 3-104 Developer Workbench - save synonym

3.11 Creating filters

You can predefine standard selection criteria in the master file, both using the web interface and the Developer Workbench tool. This allows the criteria to be specified once and used in multiple reports. In this example, we create a filter called Europe. This allows us to easily run a report including/excluding the European countries.

3.11.1 Creating filters with the web interface

To create a filter in your synonym with the web interface, proceed as follows:

1. Open the DB2 web Query web interface, right-click your folder, and select **Metadata** → **Edit** as shown in Figure 3-105.

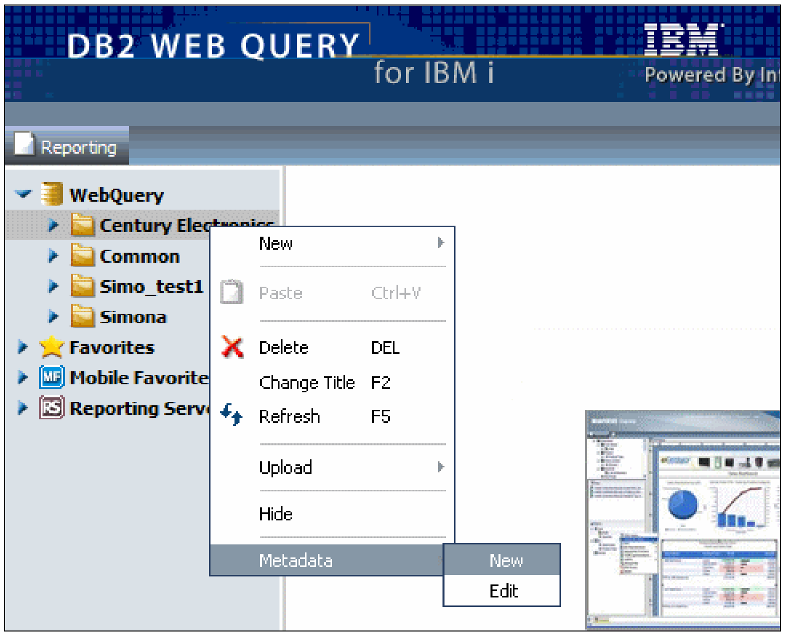


Figure 3-105 Web Interface - Edit Metadata

2. Right-click your synonym; here we use **cen_orders** - and select **Open** (Figure 3-106).

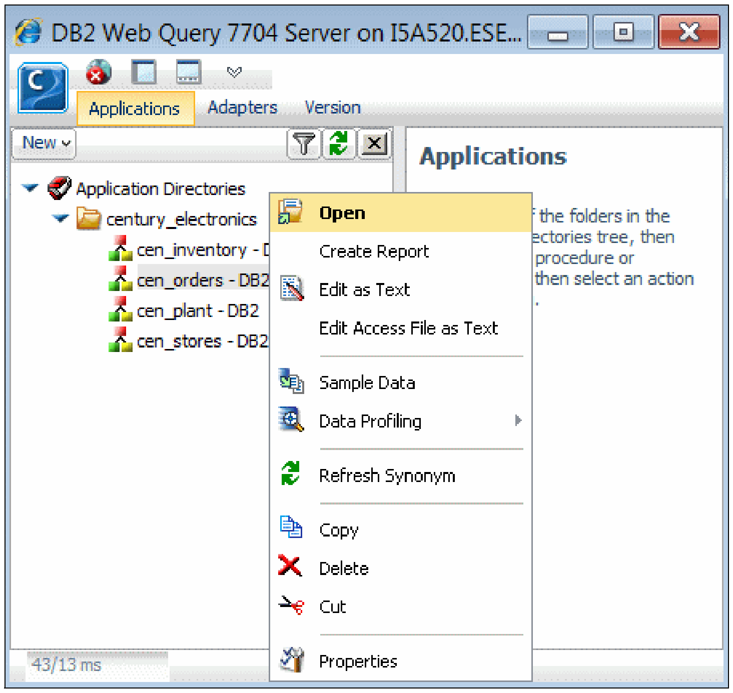


Figure 3-106 Web Interface - open metadata

3. The synonym editor opens up. Right-click the segment name (here it is **ORDERS)** and select **Insert** → **Filter** (Figure 3-107).

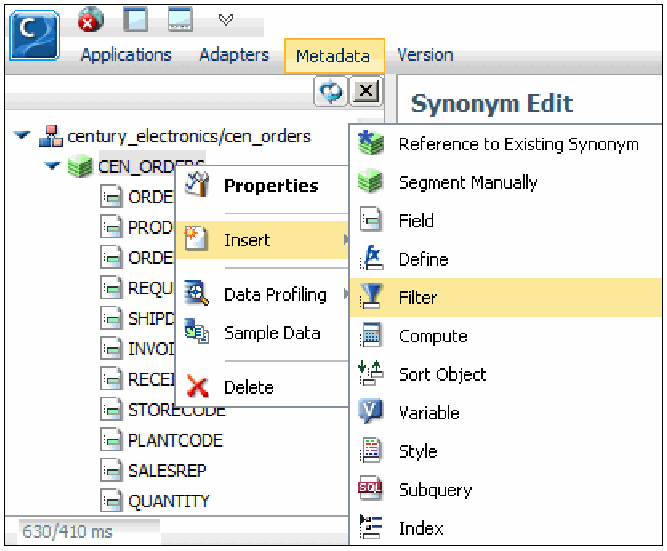
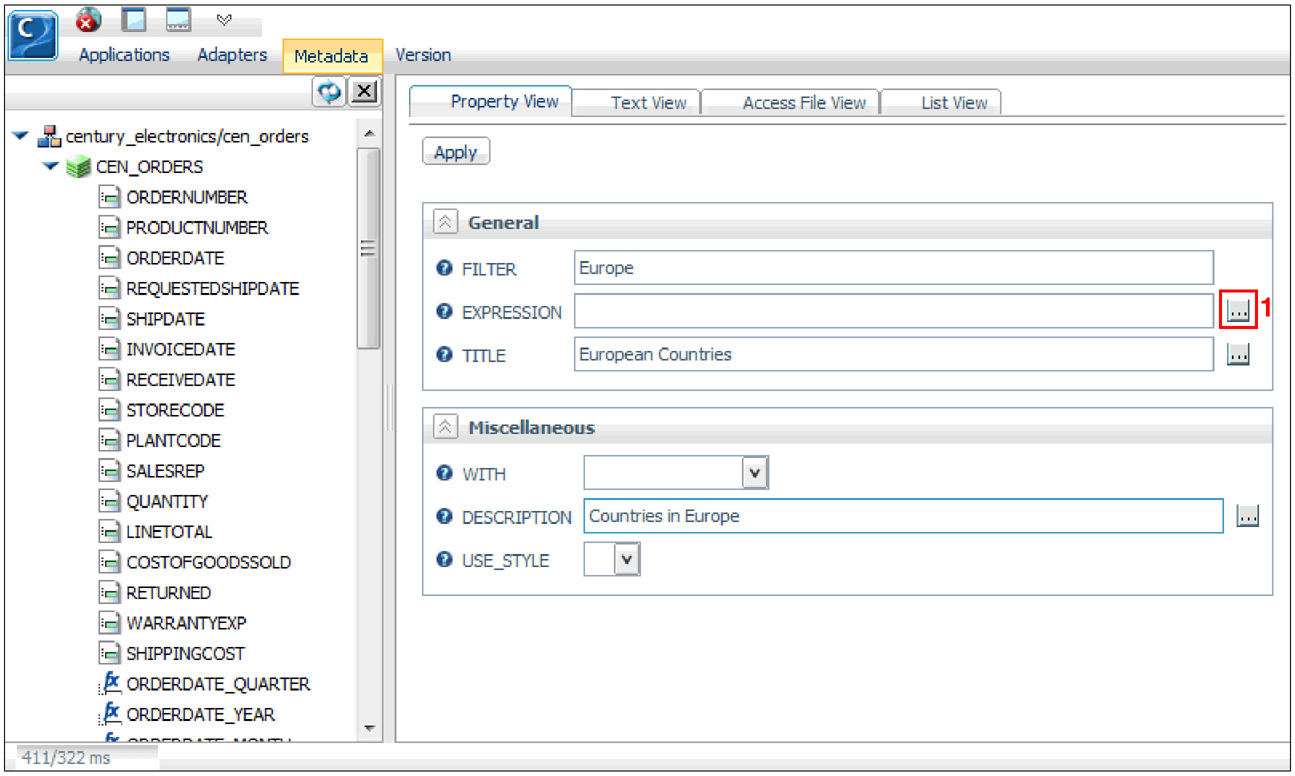


Figure 3-107 Web Interface - Insert Filter

4. In the right-hand side of the pane, you get the **Property View** for the filter you are creating. Type in a name and a title for the new filter (Figure 3-108). Here we use Europe and European Countries, since we want to create a pre-defined filter to check for European countries information in our reports; or for non-European countries, since filters can be checked for true or false at runtime. After you have entered a title, click the little dotted button on the right side of the EXPRESSION field (#1) to enter the rule you want to set.



FILTER: Europe
EXPRESSION:
TITLE: European Countries
DESCRIPTION: Countries in Europe

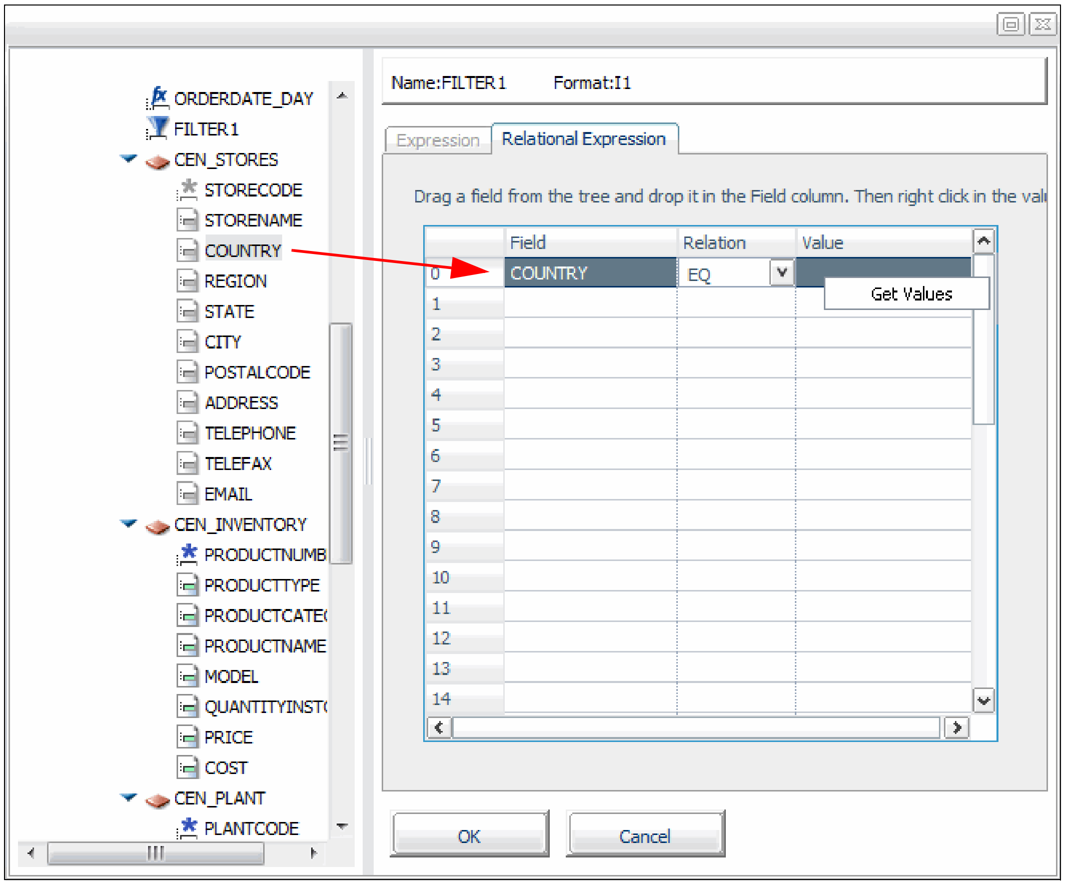
Figure 3-108 Web interface, filter properties

5. You are presented with the Relational Expression calculator. On the left side, in the list of available fields. locate the field on which you want to set the comparison (here we use COUNTRY) and drag it in the “Field” field. Select the relation you want to set, then right-click the “Value” field and select **Get Values** as shown in Figure 3-109.

Note: The following comparison operators are provided in the web interface:

- EQ
- NE
- LT
- GT
- LT
- GE

Developer Workbench provides a wider selection.


Name:FILTER1 Format:I1

Drag a field from the tree and drop it in the Field column. Then right click in the val

	Field	Relation	Value
0	COUNTRY	EQ	

Figure 3-109 Web Interface - relational expression in filter

6. In the filter panel, you will be presented with all the values that are available in the field at the moment. Select the appropriate values and add them to the right-hand side of the list. When you are done, click **OK**. Notice that in this interface, it is not possible to key in values that are not in database when building the filter (that is, if later on, you start doing business with Portugal, you have to edit your filter and add Portugal). See Figure 3-110.

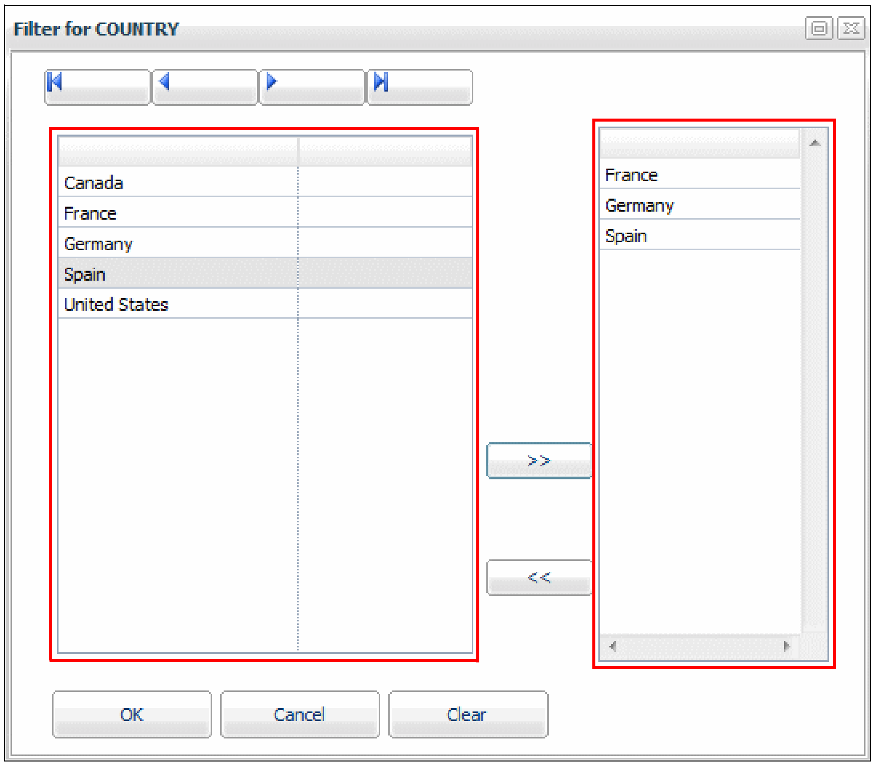
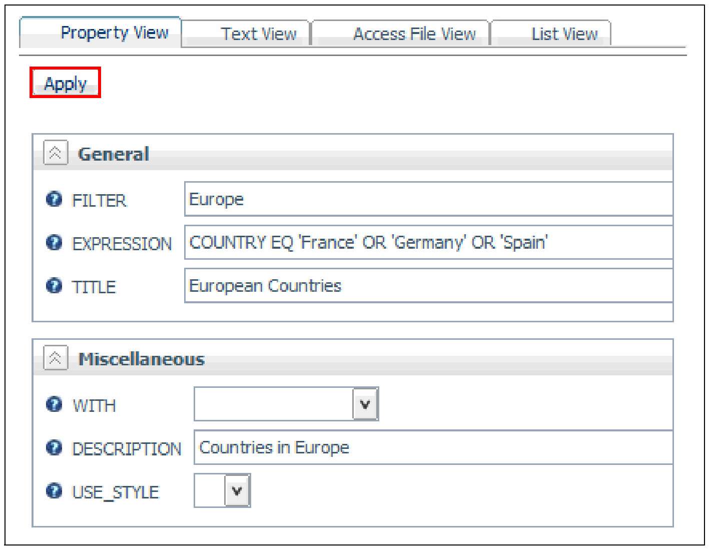


Figure 3-110 Web Interface - set relational expression in filter

7. You will be brought back to the Filter Property View pane. Select **APPLY** as shown in Figure 3-111.



FILTER	Europe
EXPRESSION	COUNTRY EQ 'France' OR 'Germany' OR 'Spain'
TITLE	European Countries
WITH	
DESCRIPTION	Countries in Europe
USE_STYLE	

Figure 3-111 Web Interface - Filter completed

8. Save your synonym as shown in Figure 3-112.

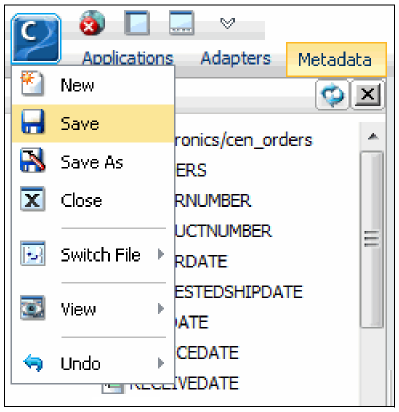


Figure 3-112 Web Interface - save synonym

9. The filter will be displayed in your list of column names when you create a report. Drag the filter to the Selection criteria pane. A `WHERE filter is true` statement is generated by default. This report will now include only the countries that are defined in the Europe filter (Figure 3-119).

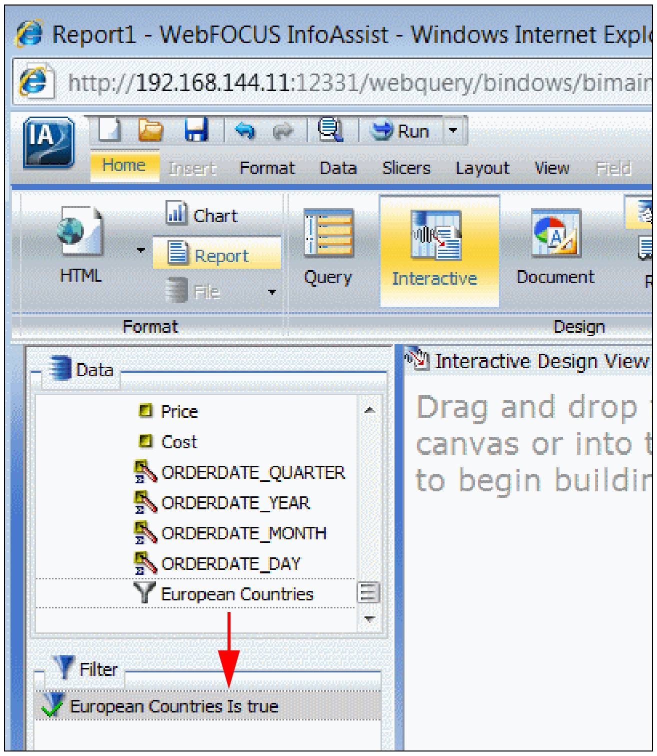
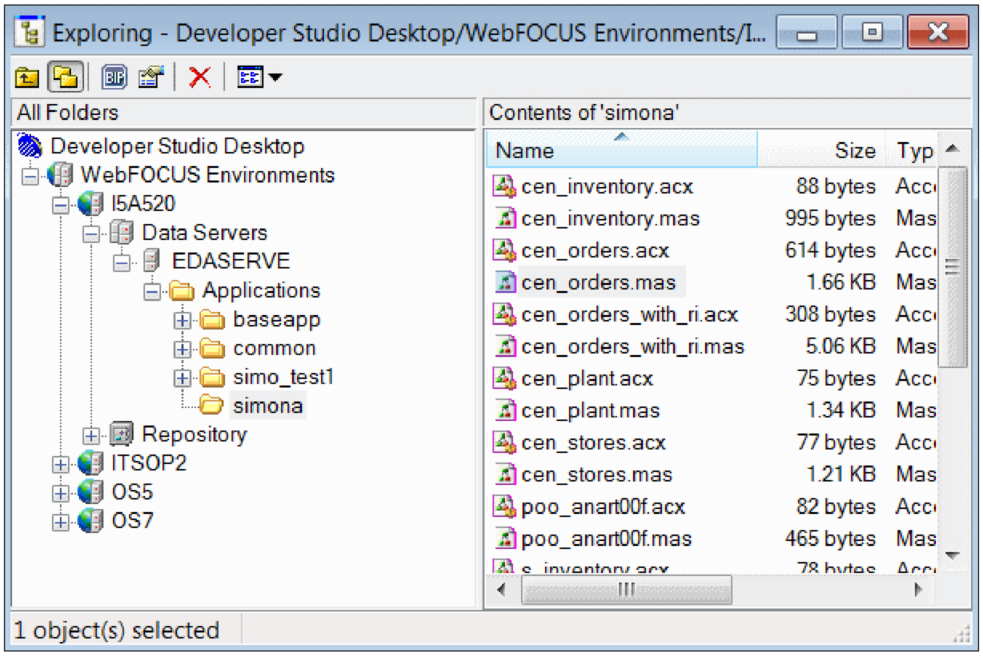


Figure 3-113 Using predefined filter in a report

3.11.2 Creating filters with Developer Workbench

To create a filter in a synonym using Developer Workbench, proceed as follows:

1. Open Developer Workbench, select the system on which to work, and validate with your userid and password. Remember that the user must be part of the DevWorkbench group (use **Administration** → **Security Center** in the web interface to set up). Select **Data Servers** → **EDASERVE** → **Applications** → ***<yourfolder>*** as shown in Figure 3-114. Double-click the synonym where you want to define the filter.


Exploring - Developer Studio Desktop/WebFOCUS Environments/I...

All Folders

- Developer Studio Desktop
  - WebFOCUS Environments
    - I5A520
      - Data Servers
        - EDASERVE
          - Applications
            - baseapp
            - common
            - simo_test1
            - simona
      - Repository
    - ITSOP2
    - OS5
    - OS7

Contents of 'simona'

Name	Size	Typ
cen_inventory.acx	88 bytes	Acc
cen_inventory.mas	995 bytes	Mas
cen_orders.acx	614 bytes	Acc
cen_orders.mas	1.66 KB	Mas
cen_orders_with_ri.acx	308 bytes	Acc
cen_orders_with_ri.mas	5.06 KB	Mas
cen_plant.acx	75 bytes	Acc
cen_plant.mas	1.34 KB	Mas
cen_stores.acx	77 bytes	Acc
cen_stores.mas	1.21 KB	Mas
poo_anart00f.acx	82 bytes	Acc
poo_anart00f.mas	465 bytes	Mas
s_inventory.acx	78 bytes	Acc

1 object(s) selected

Figure 3-114 Developer Workbench - add filter to synonym

2. Double-click the synonym where you want to add a new filter.
3. Right-click the master file name segment that contains the field that you would like to filter and select **Insert** → **Filter,** as shown in Figure 3-115, or alternatively by selecting the symbol highlighted. In this case, we want to filter on the Country field, which exists in our STORES segment.

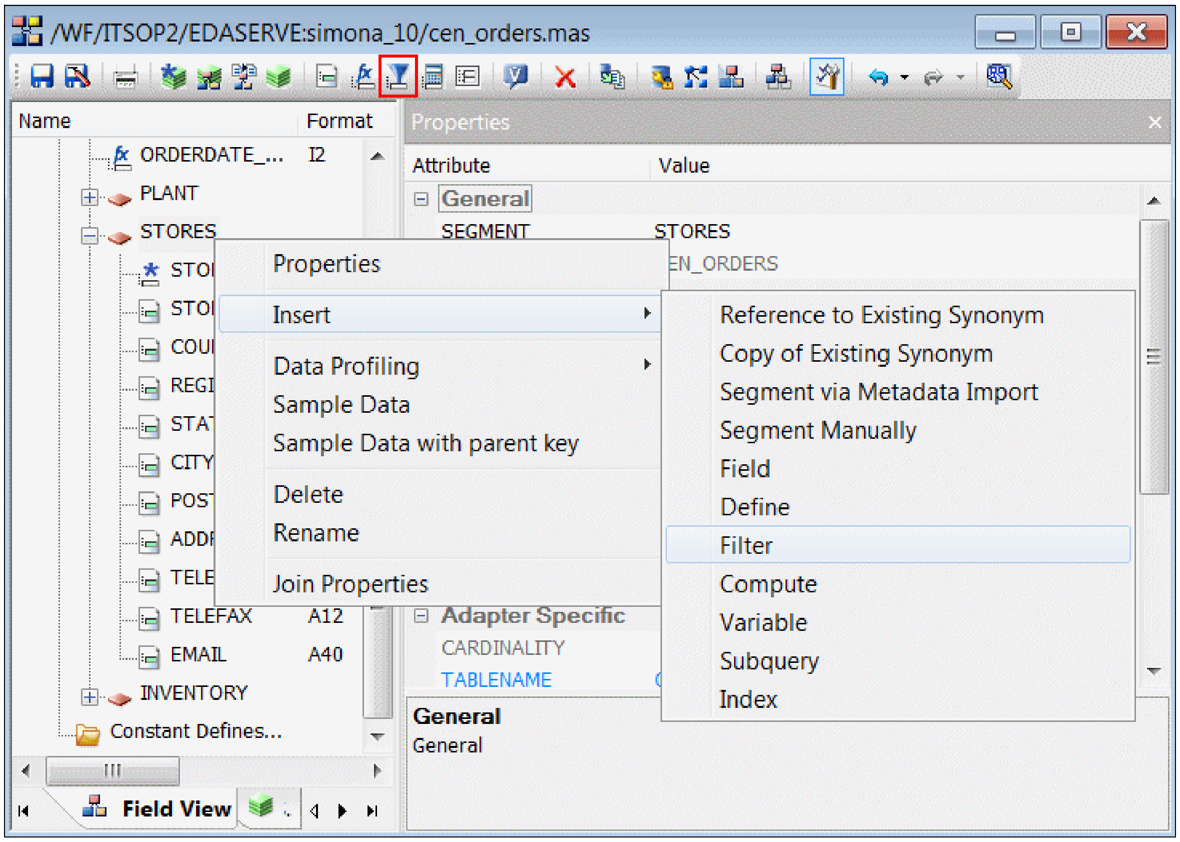


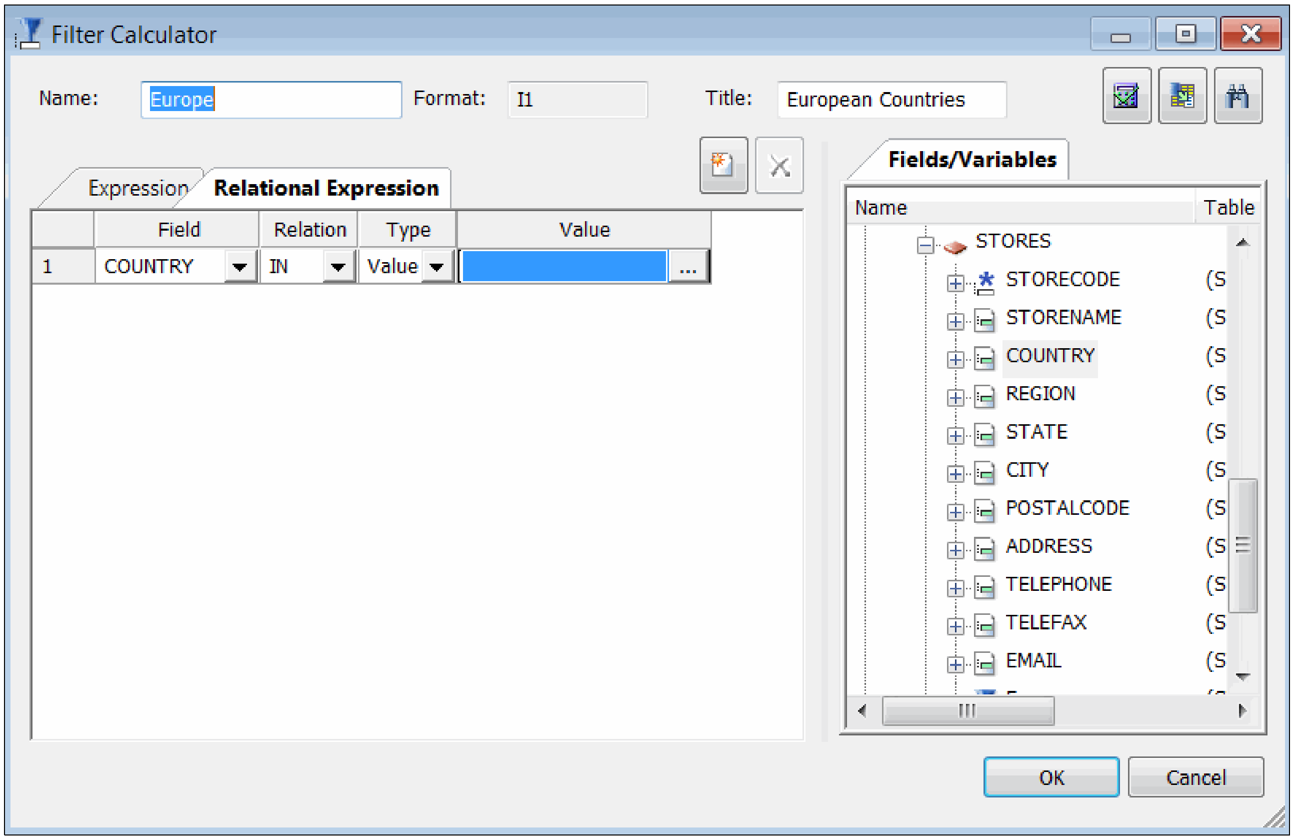
Figure 3-115 Developer Workbench - Define Filter

4. In the Filter Calculator window (Figure 3-116) go to Column, type `Europe`, and define it as an I1 (integer one long) column. This means that we can test it for true or false in our reports. Double-click **COUNTRY** under the STORES segment to add it to the filter. Change the Relations to IN because we must provide multiple countries for the value. Click the list button on the value field.

Note: The following comparison operators are provided in Developer Workbench:

- EQ
- NE
- LT
- GT
- LT
- GE
- CONTAINS
- OMITS
- LIKE
- UNLIKE
- IN
- INCLUDES
- EXCLUDES

This selection is wider than that available in the web interface.

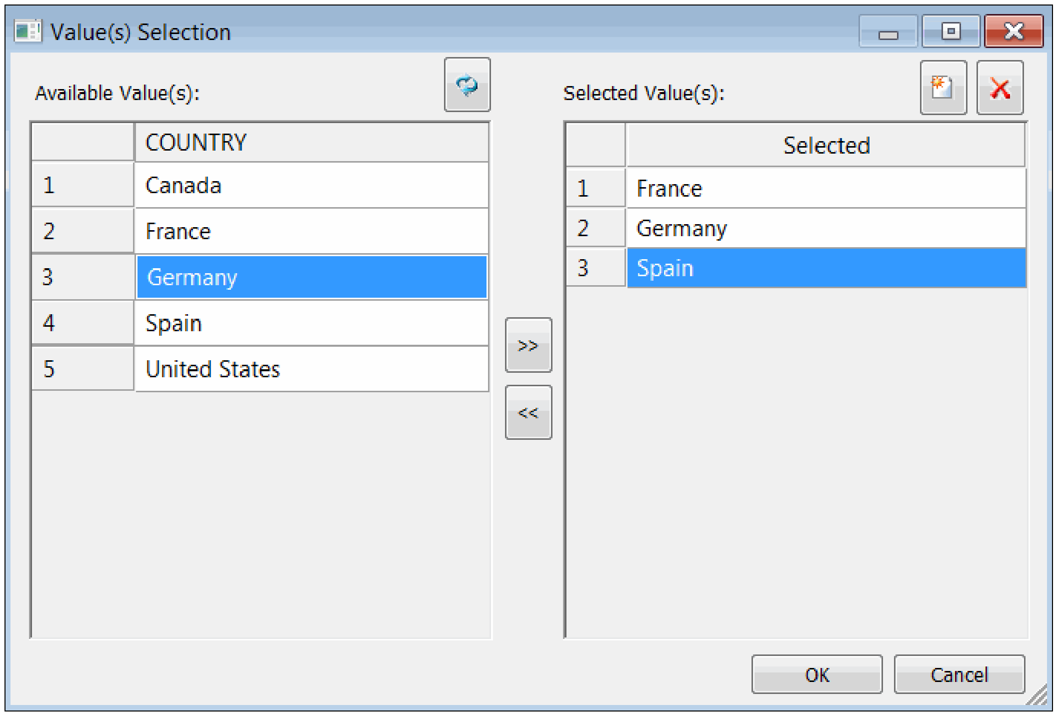


Name: Europe
Format: I1
Title: European Countries

	Field	Relation	Type	Value
1	COUNTRY	IN	Value	

Figure 3-116   Developer Workbench - Creating a filter

5. Double-click the countries or use the > button to move the selections to the right-hand side in the Value(s) Selection window (Figure 3-117). Click **OK** to close the Value(s) Selection window.

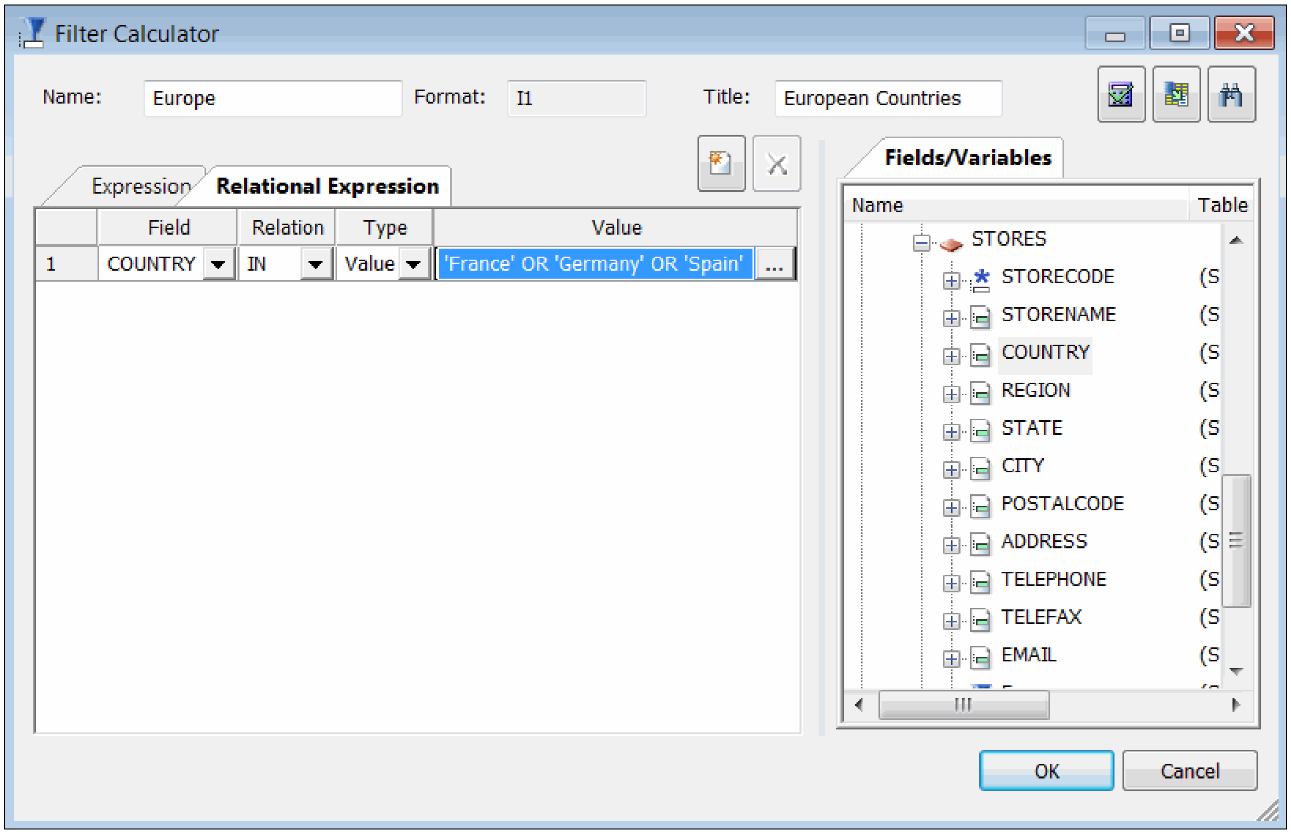


	COUNTRY
1	Canada
2	France
3	Germany
4	Spain
5	United States

	Selected
1	France
2	Germany
3	Spain

Figure 3-117 Developer Workbench - Values in a filter

6. If necessary, you can add more elements by typing directly in the VALUE box. For example, you know that your company is going to open new stores in Portugal and want your metadata to be ready for that occurrence, so you add **OR 'Portugal'** to the list of values to be checked. Click **OK** to close the Filter Calculator window (Figure 3-118).



	Field	Relation	Type	Value
1	COUNTRY	IN	Value	'France' OR 'Germany' OR 'Spain'

Figure 3-118   Developer Workbench - Creating a Filter Detail

7. The filter will be displayed in your list of column names when you create a report. Drag the filter to the Selection criteria pane. A `WHERE filter is true` statement is generated by default. This report will now include only the countries that are defined in the Europe filter (Figure 3-119).

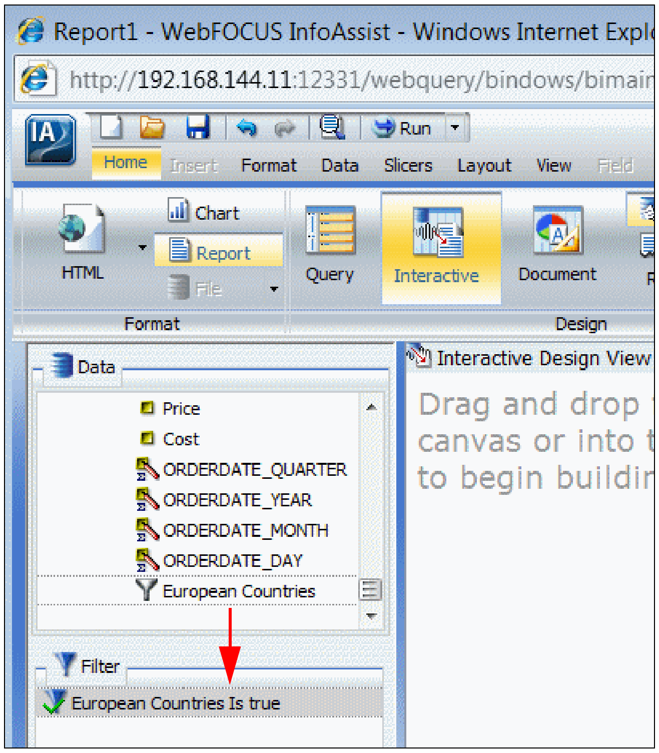


Figure 3-119 Using predefined filter

3.12 Defining dimensions: InfoMini and OLAP

Within synonyms, it is possible to define “hierarchies” or “dimensions” to be used:

- In reports to enable pre-defined paths for drill down in data
- To be able to include new columns according to end users need (OLAP function - refer to Chapter 14, “Assignment #7: Implementing OLAP” on page 429)
- To include pre-chained filters in reports (using InfoMini slicers refer to 12.4, “Filter in metadata: Europe Revenue and Profit report” on page 372) at runtime.

Before a user can OLAP enable a report, a developer must define hierarchies that represent the data. This hierarchy is automatically used by DB2 Web Query whenever a developer asks to OLAP enable a report that references tables that contain these hierarchies. *Hierarchies*, also known as *dimensions*, can involve elements from multiple tables, although the norm is to have a single hierarchy composed of columns from a single table.

Proceed as follows:

1. Start Developer Workbench.
2. Expand **WebFOCUS Environments** → ***system name*** → **Data Servers** → **EDASERVE** → **Applications** → ***<yourfolder>***.
3. Right-click the **cen_orders.mas** table and select **Edit in Synonym Editor** (Figure 3-120).

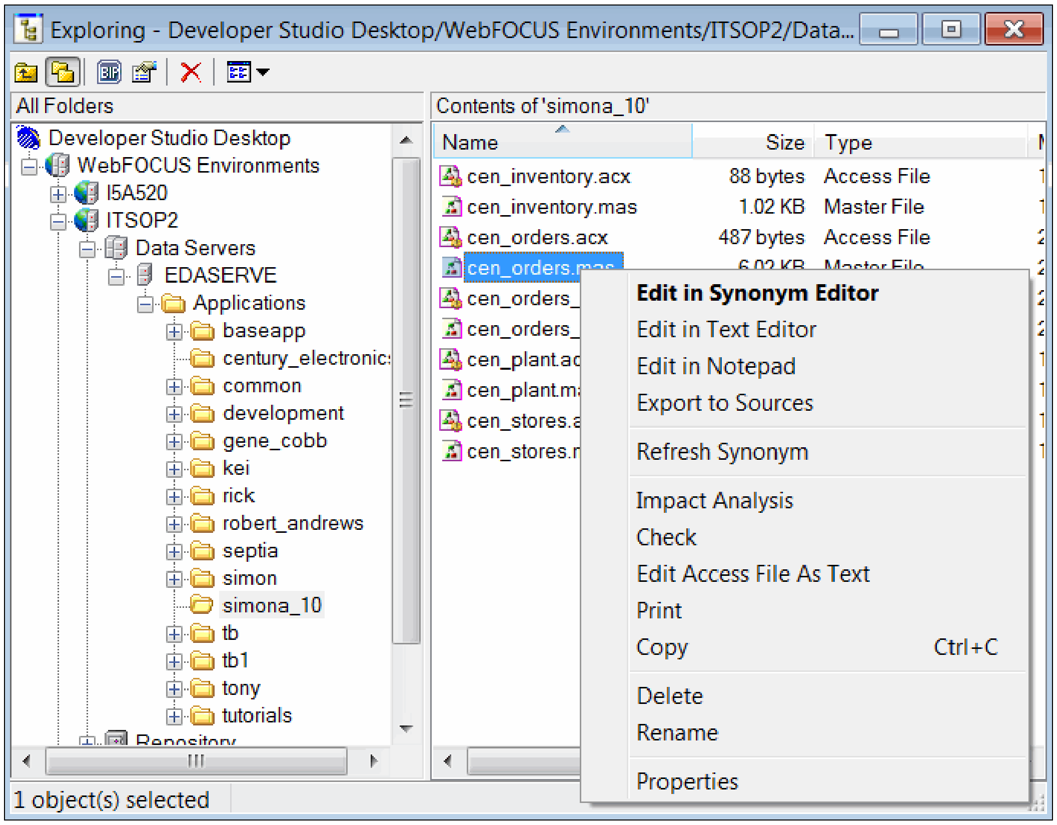


Figure 3-120 Developer Workbench, edit synonym

4. In the Synonym Editor, select the Dimension Builder icon in the tool bar, highlighted at # 1 in Figure 3-121. You are presented with the Dimension Builder pane, highlighted at #2.

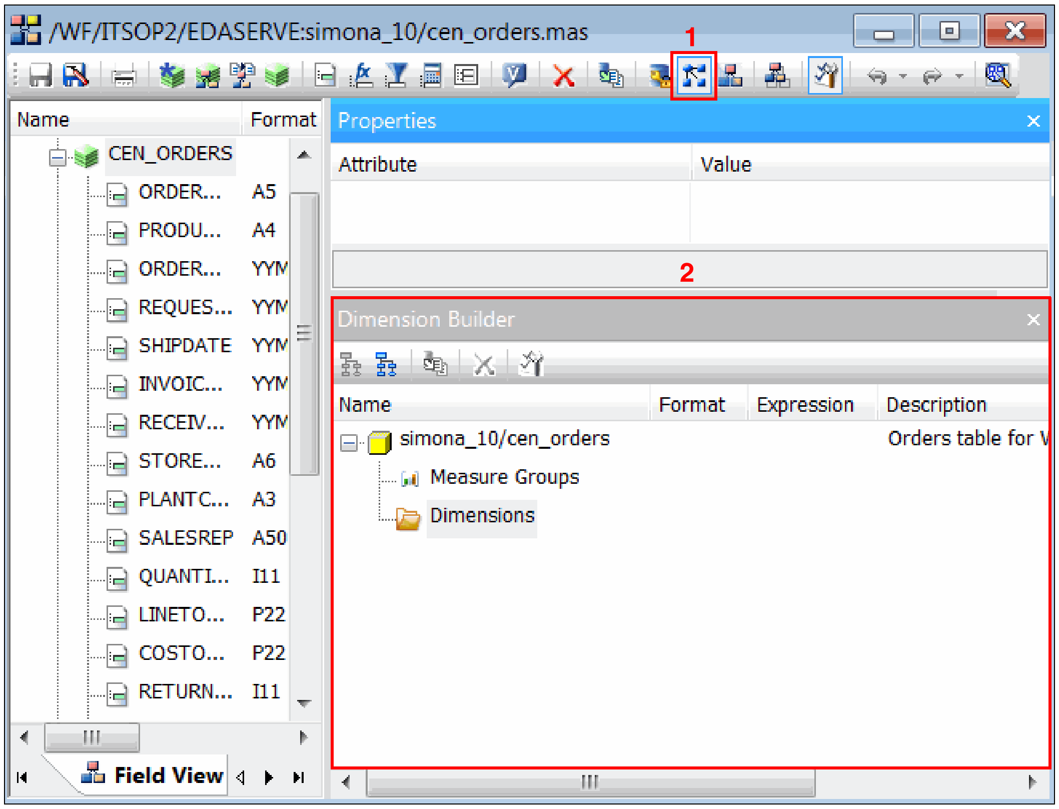


Figure 3-121 Developer Workbench - Dimension Builder

5. Right-click **Dimension** and select **Insert** → **New Time Dimension**. as shown in Figure 3-122.

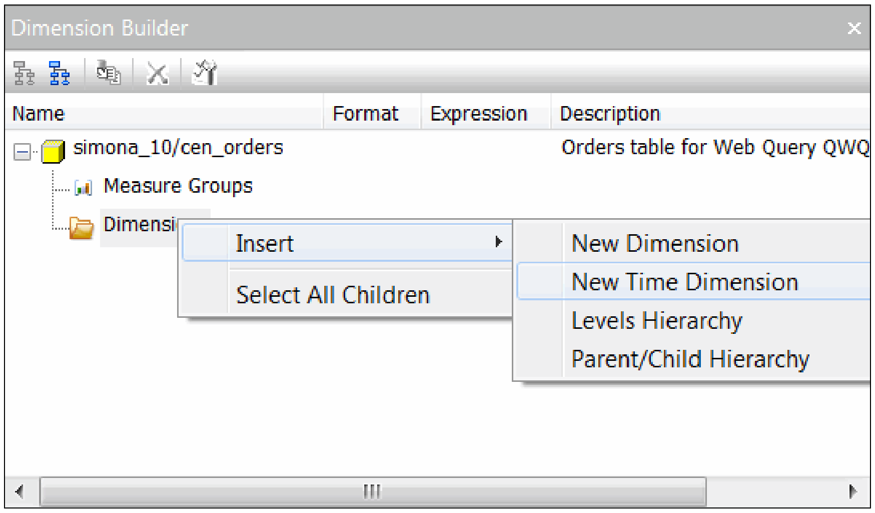


Figure 3-122 Dimension Builder, New Time Dimension

6. You are presented with a list of the segments defined in your synonym. Select the segment that holds the date field you want to use. In our example, we use **ORDERS**, with its ORDERDATE field (Figure 3-123). Click **OK**.

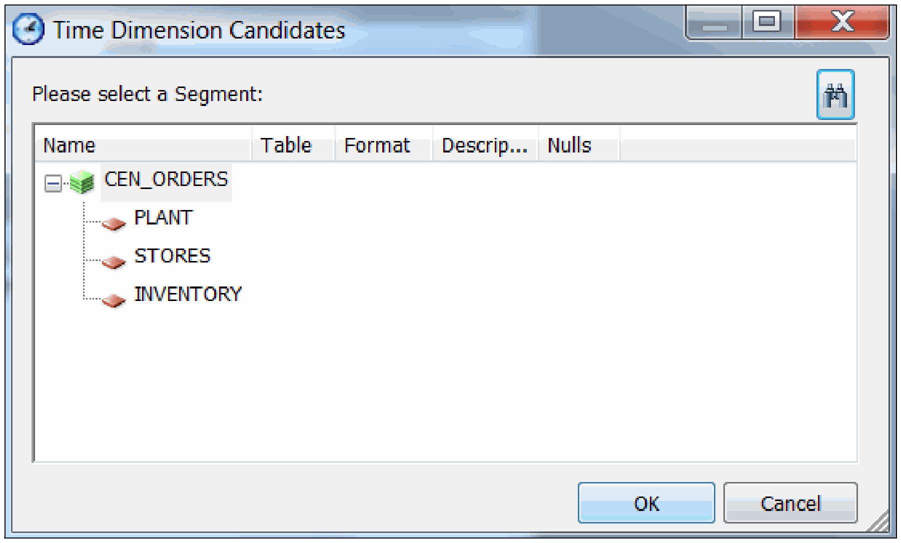
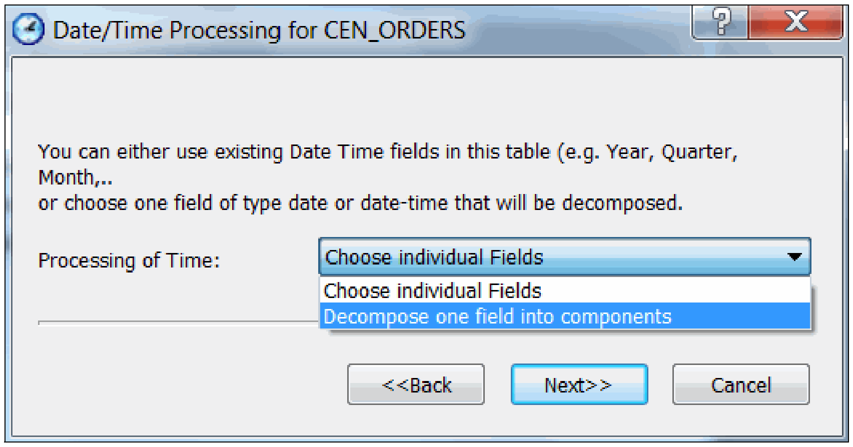


Figure 3-123 Time dimension candidates

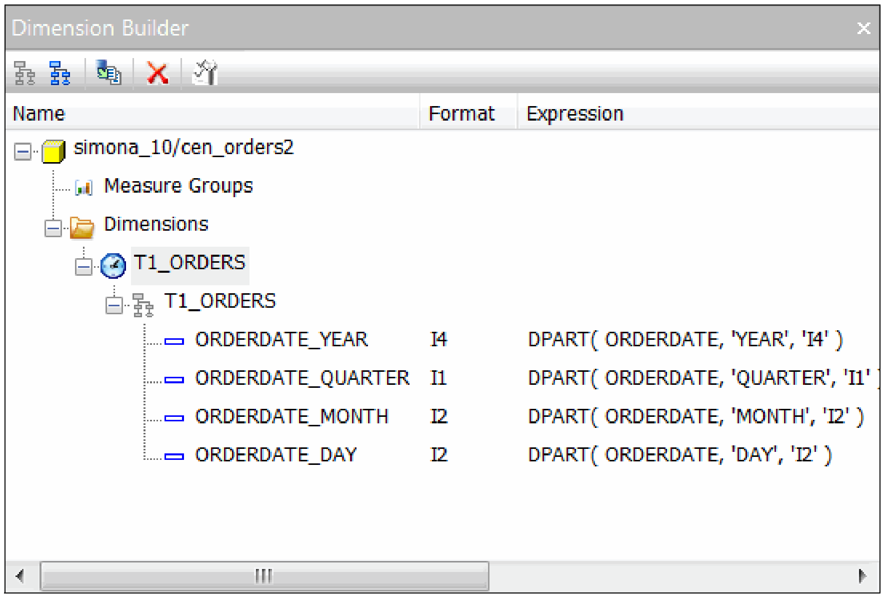
7. You are presented with a panel where you can set if you want to choose individual fields (where the basic date components are already decomposed - see 3.7, "Date decomposition" on page 78) or if you want one single field to be decomposed into basic elements. Here we take the second choice (Figure 3-124). Select **Next**.



You can either use existing Date Time fields in this table (e.g. Year, Quarter, Month,..
or choose one field of type date or date-time that will be decomposed.

Figure 3-124 Date Dimension - Decompose date field

8. You will notice that a full time decomposed dimension is populated, as shown in Figure 3-125.

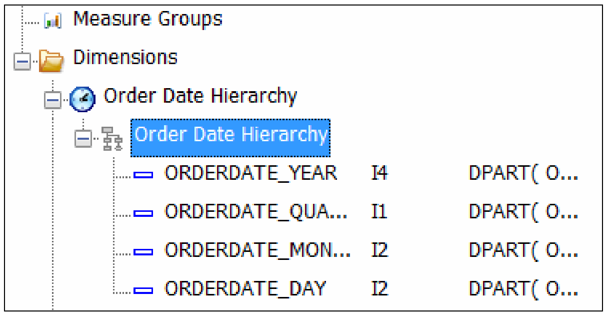


Dimension Builder

Name	Format	Expression
simona_10/cen_orders2		
Measure Groups		
Dimensions		
T1_ORDERS		
T1_ORDERS		
ORDERDATE_YEAR	I4	DPART(ORDERDATE, 'YEAR', 'I4')
ORDERDATE_QUARTER	I1	DPART(ORDERDATE, 'QUARTER', 'I1'
ORDERDATE_MONTH	I2	DPART(ORDERDATE, 'MONTH', 'I2')
ORDERDATE_DAY	I2	DPART(ORDERDATE, 'DAY', 'I2')

Figure 3-125 Time Dimension

9. Right-click the original label, select **Rename**, and rename it to suit your needs. Here we use **Order Date Hierarchy**. Repeat the process for the second level title (Figure 3-126).



Measure Groups		
Dimensions		
Order Date Hierarchy		
Order Date Hierarchy		
ORDERDATE_YEAR	I4	DPART(O...
ORDERDATE_QUA...	I1	DPART(O...
ORDERDATE_MON...	I2	DPART(O...
ORDERDATE_DAY	I2	DPART(O...

Figure 3-126 Rename Hierarchy

10. Right-click **Dimensions** and select **Levels Hierarchy** as shown in Figure 3-127.

    There are two types of hierarchies. One is based on levels and the other is based on parent-child relationships. We use level hierarchies. The level hierarchy is the example that we have used so far, that is, country, regions, state, and city are a level hierarchy.

    For more information about the different types of hierarchies, see the Developer Workbench help text.

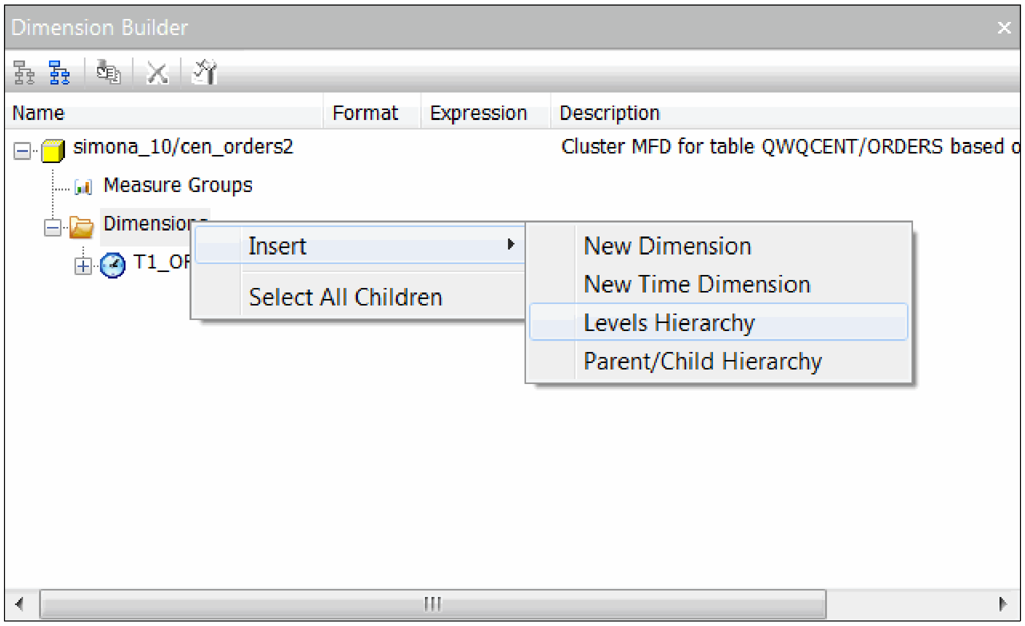


Figure 3-127 Insert new dimension

11. Right-click the new hierarchy and select **Rename** (Figure 3-128). Here we use **Product Info**. Repeat the step at the Dimension level.

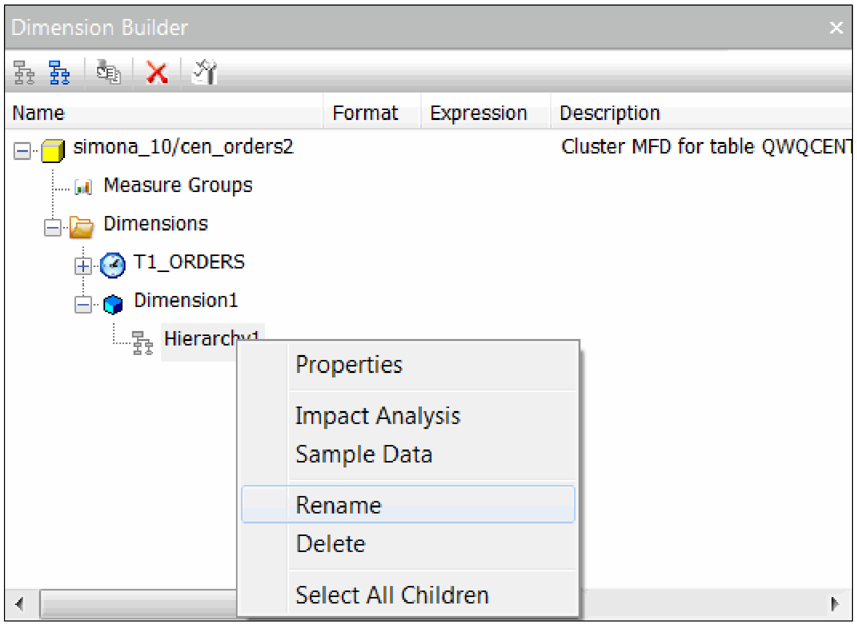
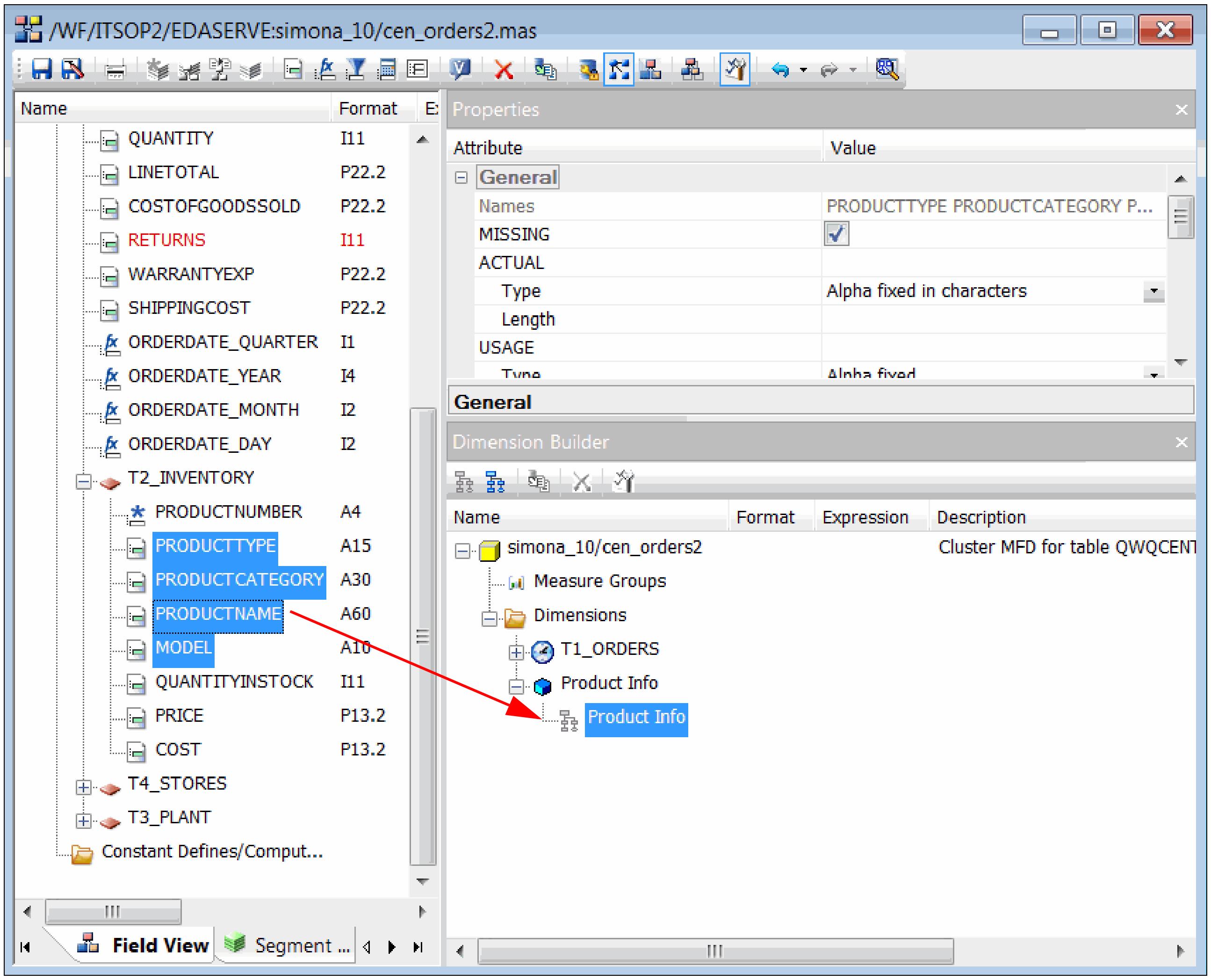


Figure 3-128 Rename a hierarchy

12. In the left-hand side of the Edit Synonym window, select the segment in which the required fields are located; here we use **INVENTORY**. Hold the CTR key and select the field you want to be in the dimension in the exact order you want them to appear (here the sequence is **Product Type, Product Category, Model, Product Name**). Press the left mouse button and drag all fields to the hierarchy on the right-hand side of the panel; see Figure 3-129. Alternatively, you can drag each field, one by one.


/WF/ITSOP2/EDASERVE:simona_10/cen_orders2.mas

Name	Format
QUANTITY	I11
LINETOTAL	P22.2
COSTOFGOODSSOLD	P22.2
RETURNS	I11
WARRANTYEXP	P22.2
SHIPPINGCOST	P22.2
ORDERDATE_QUARTER	I1
ORDERDATE_YEAR	I4
ORDERDATE_MONTH	I2
ORDERDATE_DAY	I2
T2_INVENTORY	
PRODUCTNUMBER	A4
PRODUCTTYPE	A15
PRODUCTCATEGORY	A30
PRODUCTNAME	A60
MODEL	A10
QUANTITYINSTOCK	I11
PRICE	P13.2
COST	P13.2
T4_STORES	
T3_PLANT	
Constant Defines/Comput...	

Properties

Attribute	Value
General	
Names	PRODUCTTYPE PRODUCTCATEGORY P...
MISSING	✓
ACTUAL	
Type	Alpha fixed in characters
Length	
USAGE	
Type	Alpha fixed

Dimension Builder

Name	Format	Expression	Description
simona_10/cen_orders2			Cluster MFD for table QWQCENT
Measure Groups			
Dimensions			
T1_ORDERS			
Product Info			
Product Info			

Figure 3-129 Drag fields to hierarchy

13. You can repeat the process from step 10 and build as many dimensions as you need. when you are finished save your metadata.

You have now completed all the definitions that are required for OLAP-enabling your reports. Typically, this is done once by the IT department. After this, any user with authority to create reports can choose to OLAP-enable their report.

You can see how easy it is to define a hierarchy in your data and create the necessary metadata to OLAP-enable your reports for detailed analysis.

Note: Developer Workbench, a DB2 Web Query optional feature, is a prerequisite for creating OLAP-enabled reports. Developer Workbench is the only interface for defining the hierarchies that are required by OLAP at the time this book was written. Eventually a Web based interface for building dimensions may be available.

3.13 Segmenting and securing Metadata: DBA

In DB2 Web Query V2, all metadata is kept in `Applications` (a directory in the IFS) that is related to the `report folder` where they are used. All metadata that must be seen and used by everybody can still be hosted in the base application, Common (`/qibm/UserData/qwebqry/apps/Common`), which is associated with the Common folder.

Having metadata in private folders ensures the following benefits:

- Only those allowed to interact (use, modify/create) with a folder can see associated metadata, thus ensuring an optimal protection of *sensitive* data.
- It is easy to extract (save) metadata from a system to import (restore) either on the same system for safety reasons or on another system for duplicating the environment.
- Identifying the desired metadata is easier when developing reports.

There are situations where you may want to hide information from certain users, so, even if they are allowed to run a report, they do not get to see data for certain columns or for certain rows. It is possible to define security rules at the metadata level, so that they are applied for every sort of reports. This is done using the DBA function, available in the Developer Workbench tool.

Notes:

- An alternative security approach based on authorization tables, SQL views, and stored procedures can be implemented. This may be easier and quicker to implement than DBA functionalities native in DB2 Web Query.
- For more information, see Gene Cobb's article "Using SQL Views and Stored Procedures to control Security with DB2 Web Query," available from the DB2 for i Wiki Article page:

  `https://www.ibm.com/developerworks/mydeveloperworks/wikis/home?lang=en#/wiki/W516d8b60d32c_4fc5_a811_5f3d840bf524/page/Articles`

Here we will see how to set up security rules in metadata using the Developer Workbench tool. This is performed using the DBA function which provides facility for setting up a reporting environment with granular security restrictions such as these:

- Segment (File)
- Field
- Value (Row level security)
- No Print (Column level security)

We will set two kind of rules:

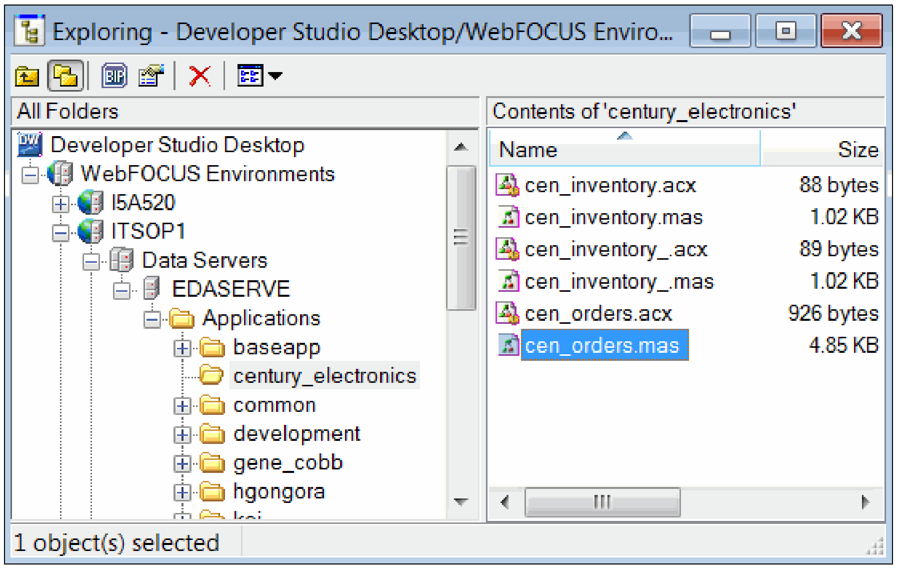
1. Field level, No Print: To prevent a user from seeing data coming from a field
2. Value, row level: To to authorize a determined user only to data coming from certain rows

3.13.1 Field level security, No Print

Proceed as follows:

1. Open Developer Workbench, select the system on which to work, and validate with your userid and password. Remember that the user must be part of the DevWorkbench group (use **Administration** → **Security Center** in the web interface to set up).

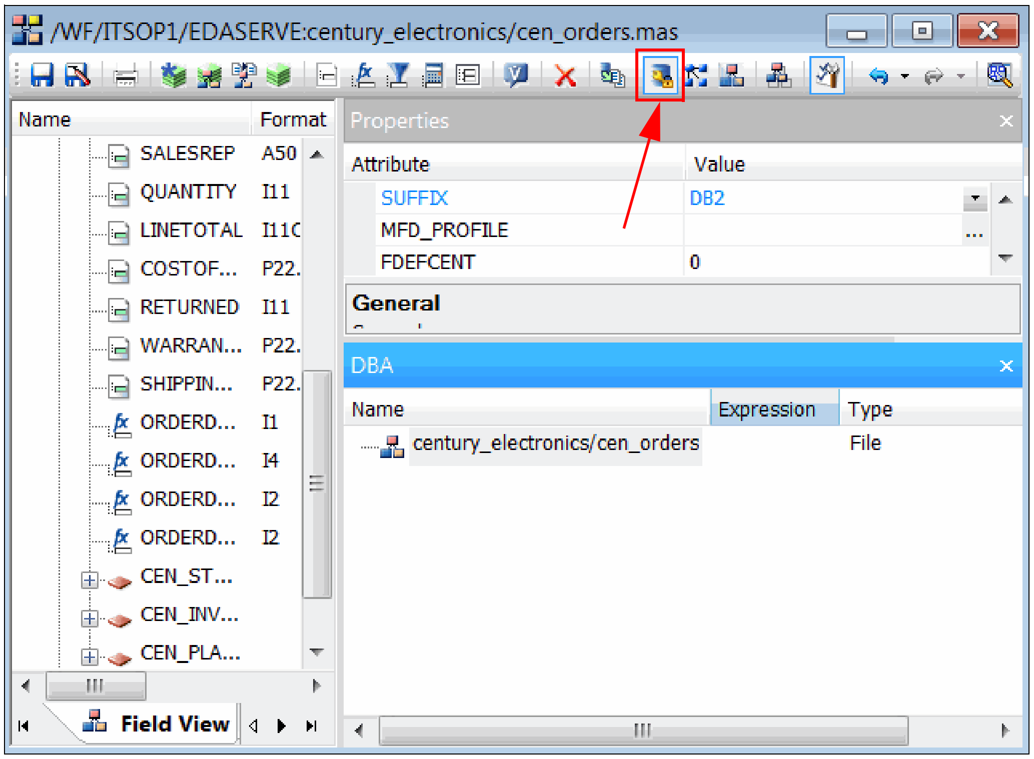
Select **Data Servers** → **EDASERVE** → **Applications** → ***<yourfolder>*** as shown in Figure 3-130. Double-click the synonym where you want to create a security rule to open it; here we use **CEN_ORDERS**.



Name	Size
cen_inventory.acx	88 bytes
cen_inventory.mas	1.02 KB
cen_inventory_.acx	89 bytes
cen_inventory_.mas	1.02 KB
cen_orders.acx	926 bytes
cen_orders.mas	4.85 KB

Figure 3-130 DevWorkbench set security - Select metadata - Step 1

2. Once in the synonym, select the icon representing the DBA function on the tool bar, as shown in Figure 3-131.



Name	Format
SALESREP	A50
QUANTITY	I11
LINETOTAL	I11C
COSTOF...	P22.
RETURNED	I11
WARRAN...	P22.
SHIPPIN...	P22.
ORDERD...	I1
ORDERD...	I4
ORDERD...	I2
ORDERD...	I2
CEN_ST...	
CEN_INV...	
CEN_PLA...	

Attribute	Value
SUFFIX	DB2
MFD_PROFILE	
FDEFCENT	0

Name	Expression	Type
century_electronics/cen_orders		File

Figure 3-131 DevWorkbench set security - Select metadata - Step 2

3. A new panel will be opened, where it is possible to the security rules. Right-click the metadata name and select **Insert** → **DBA** (Figure 3-132).

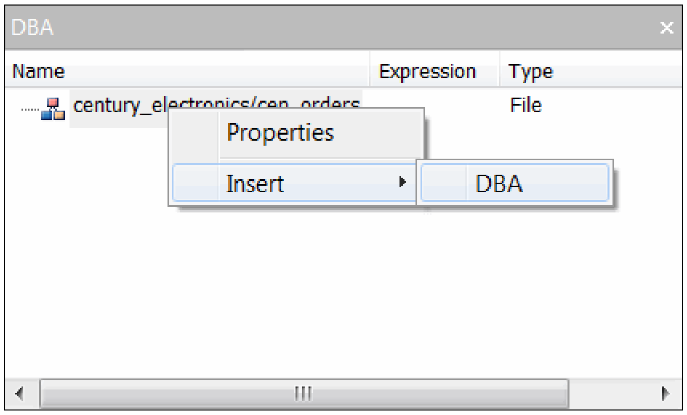


Figure 3-132 DevWorkbench set security - Select metadata - Step 3

4. The rule has the same name as the user that creates it. Now you can start defining rules. Right-click the DBA name and select **Insert** → **User** (Figure 3-133).

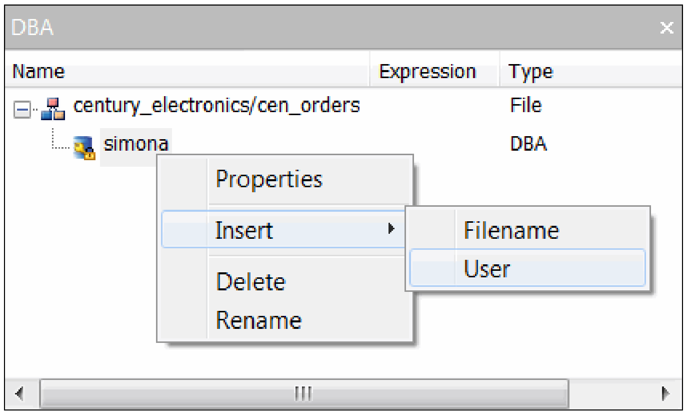


Figure 3-133 DevWorkbench set security - Select metadata - Step 4

5. You will see that a generic USER0001 is added, right-click it, and select **Rename** (Figure 3-134). Change the name to that of the user you want to manage; in our example, it will be COBBG.

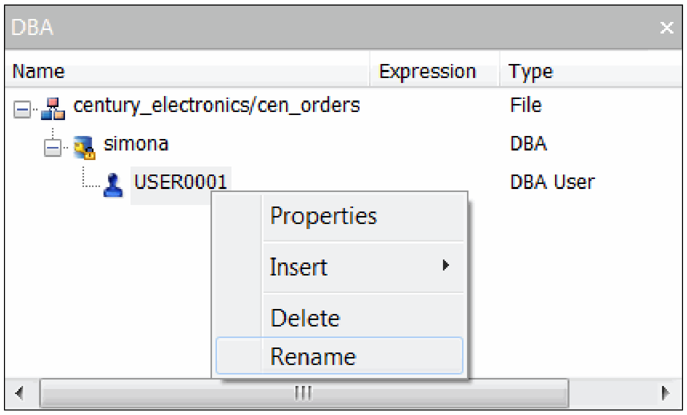


Figure 3-134 DevWorkbench set security - Select metadata - Step 5

6. Now we will define a READ rule for user COBBG. Right-click the user name and select **Insert** → **Read Access** (Figure 3-135).

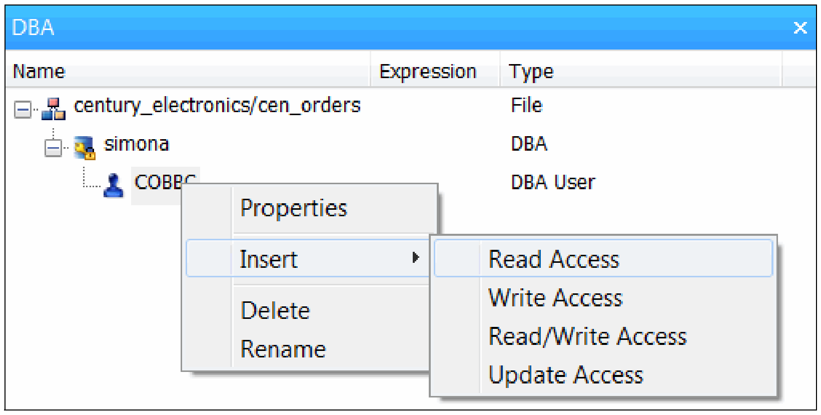


Figure 3-135 DevWorkbench set security - Select metadata - Step 6

7. Right-click **Read** and select **Insert** → **Noprint Restriction** (Figure 3-136).

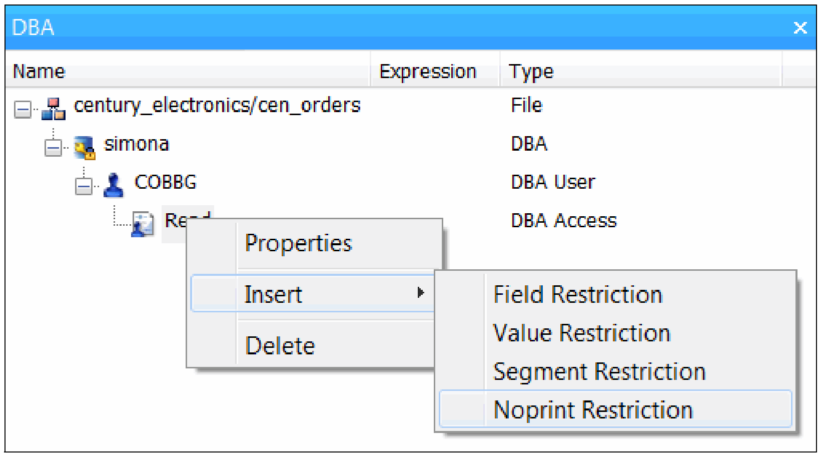


Figure 3-136 DevWorkbench set security - Select metadata - Step 7

8. The first field of the segment is referenced automatically. To change it go to the “Properties” pane, above the DBA one, and select the field you want to manage, **LINETOTAL** in our example (Figure 3-137). The bottom part of the pane will be updated to reflect the selection made.

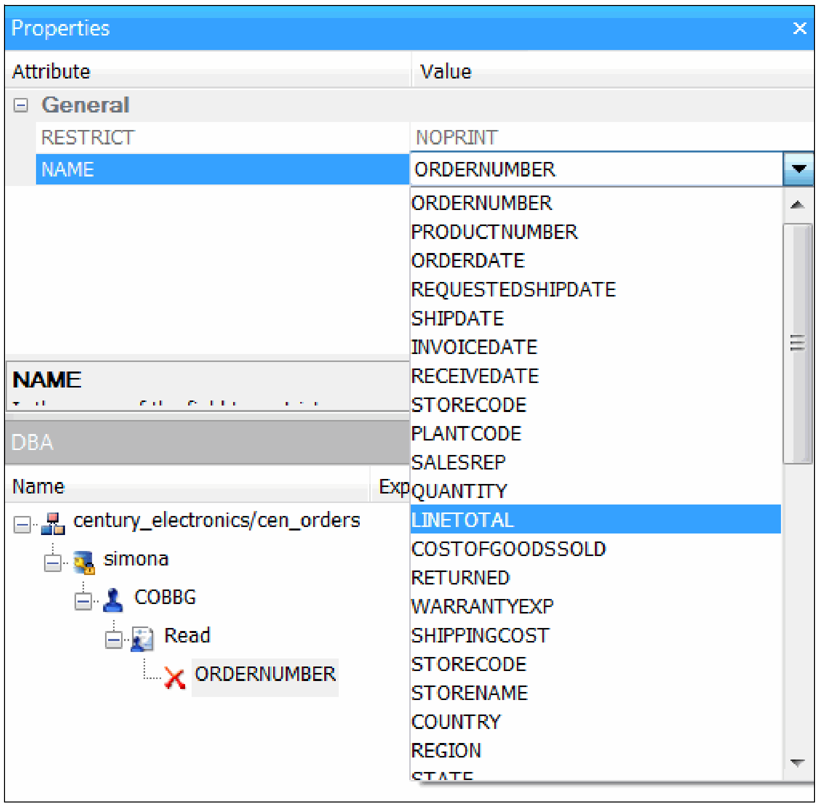
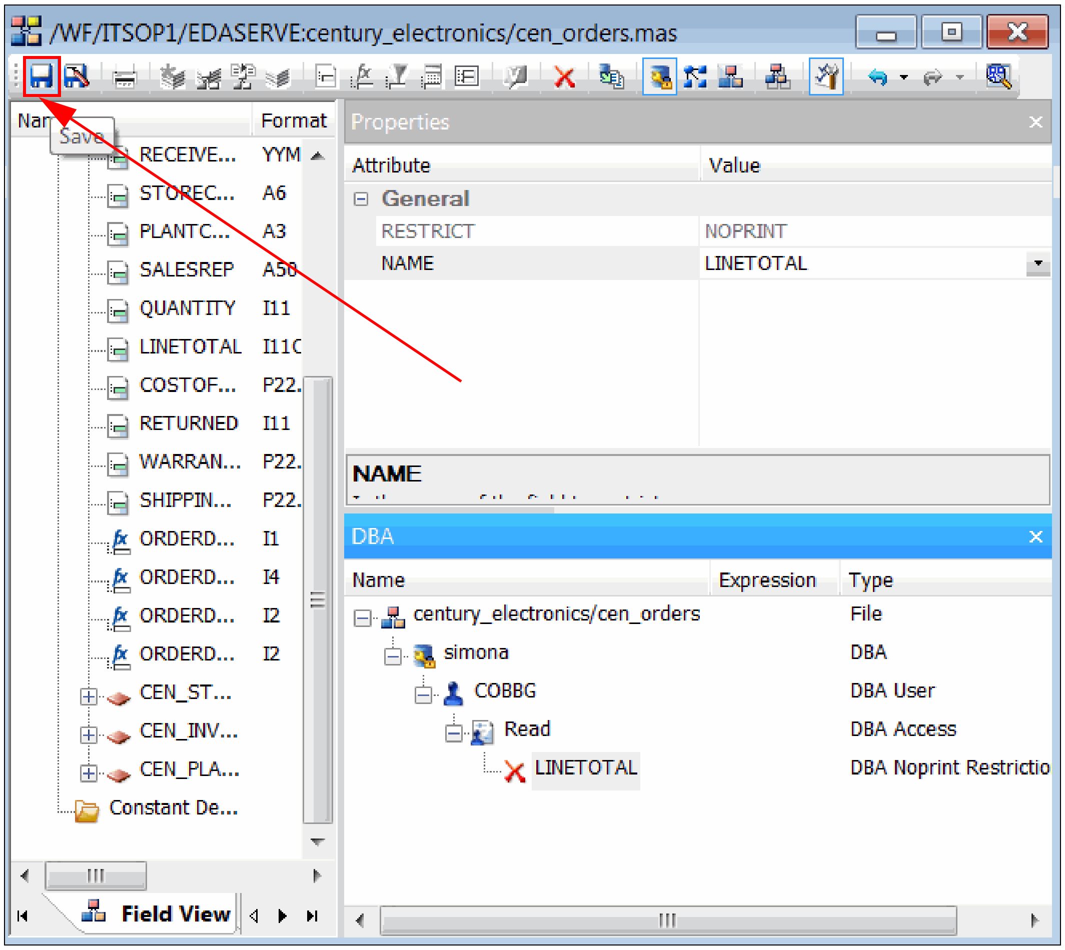


Figure 3-137 DevWorkbench set security - Select metadata - Step 8

9. The NOPRINT rule prevents user COBBG from reading the content of the selected field. The whole rule is shown in Figure 3-138. Save the metadata.



/WF/ITSOP1/EDASERVE:century_electronics/cen_orders.mas

Name	Format
RECEIVE...	YYM
STOREC...	A6
PLANTC...	A3
SALESREP	A50
QUANTITY	I11
LINETOTAL	I11C
COSTOF...	P22.
RETURNED	I11
WARRAN...	P22.
SHIPPIN...	P22.
ORDERD...	I1
ORDERD...	I4
ORDERD...	I2
ORDERD...	I2
CEN_ST...	
CEN_INV...	
CEN_PLA...	
Constant De...	

Properties

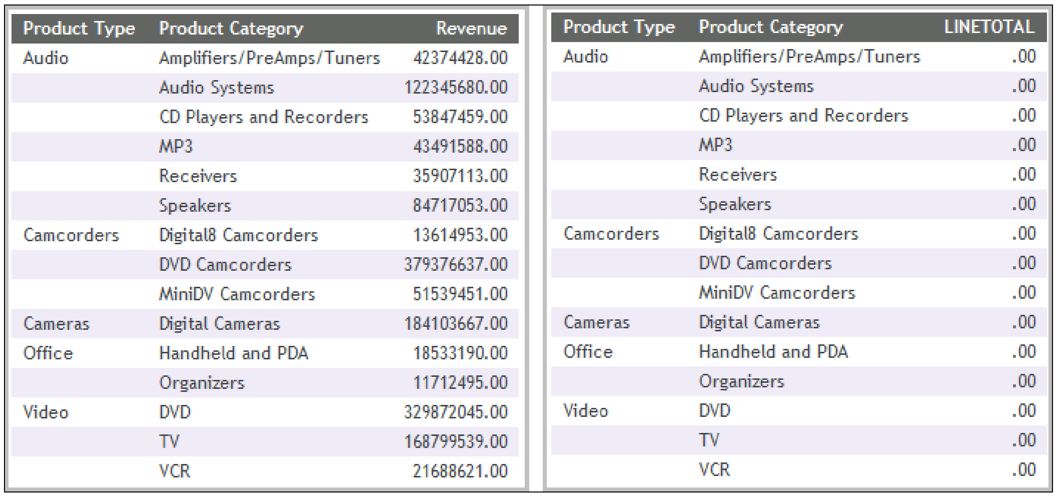
Attribute	Value
General	
RESTRICT	NOPRINT
NAME	LINETOTAL

DBA

Name	Expression	Type
century_electronics/cen_orders		File
simona		DBA
COBBG		DBA User
Read		DBA Access
LINETOTAL		DBA Noprint Restrictio

Figure 3-138 DevWorkbench set security - Select metadata - Step 9

10. Now running the same report, user COBBG would get a column with no values for LINETOTAL, where other users get the actual values, as shown in Figure 3-139.



Product Type	Product Category	Revenue
Audio	Amplifiers/PreAmps/Tuners	42374428.00
	Audio Systems	122345680.00
	CD Players and Recorders	53847459.00
	MP3	43491588.00
	Receivers	35907113.00
	Speakers	84717053.00
Camcorders	Digital8 Camcorders	13614953.00
	DVD Camcorders	379376637.00
	MiniDV Camcorders	51539451.00
Cameras	Digital Cameras	184103667.00
Office	Handheld and PDA	18533190.00
	Organizers	11712495.00
Video	DVD	329872045.00
	TV	168799539.00
	VCR	21688621.00

Product Type	Product Category	LINETOTAL
Audio	Amplifiers/PreAmps/Tuners	.00
	Audio Systems	.00
	CD Players and Recorders	.00
	MP3	.00
	Receivers	.00
	Speakers	.00
Camcorders	Digital8 Camcorders	.00
	DVD Camcorders	.00
	MiniDV Camcorders	.00
Cameras	Digital Cameras	.00
Office	Handheld and PDA	.00
	Organizers	.00
Video	DVD	.00
	TV	.00
	VCR	.00

Figure 3-139 DevWorkbench set security - Data shielded

To define the same rule for another user, you have to repeat the process, starting from #4.

3.13.2 Row level security, Value

Proceed as follows:

1. Open Developer Workbench, select the system on which to work, and validate with your userid and password. Remember that the user must be part of the DevWorkbench group (use **Administration** → **Security Center** in the web interface to set up). Select **Data Servers** → **EDASERVE** → **Applications** → ***<yourfoldr>*** and double-click the synonym in which you want to set the security rule to edit it.
2. In the synonym editor, select the DBA icon, as shown in Figure 3-140.

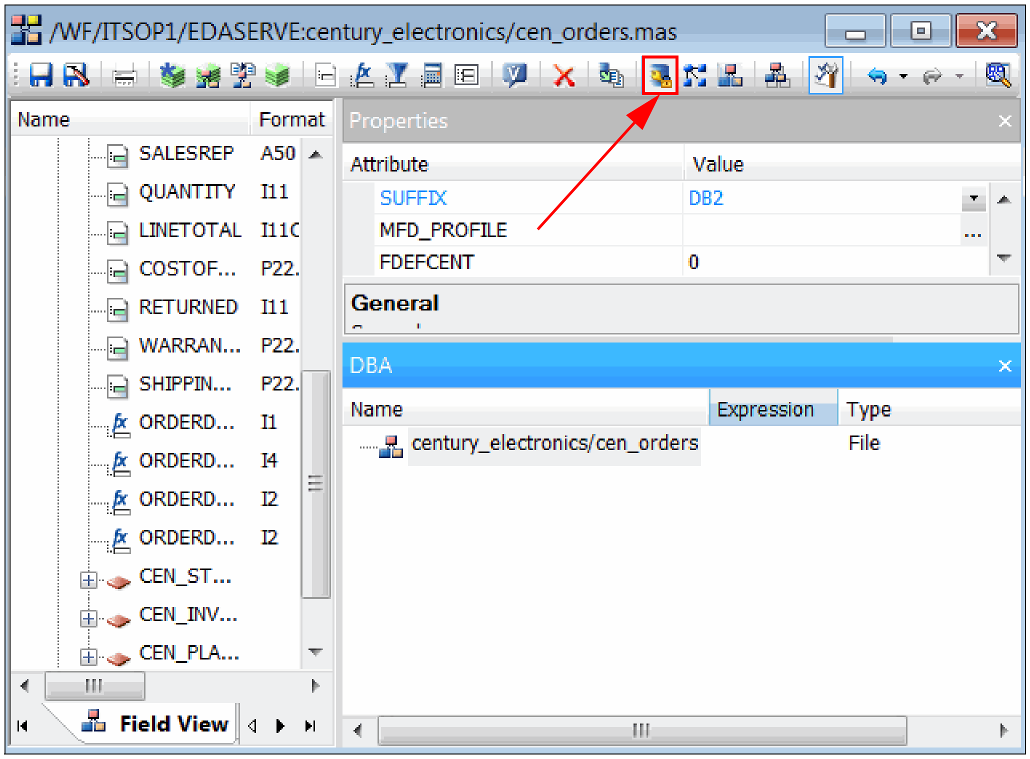


Figure 3-140 Developer Workbench, DBA

3. In the bottom right side of the synonym editor, the DBA panel appears. Right-click the synonym name and select **Insert** → **DBA** (Figure 3-141).

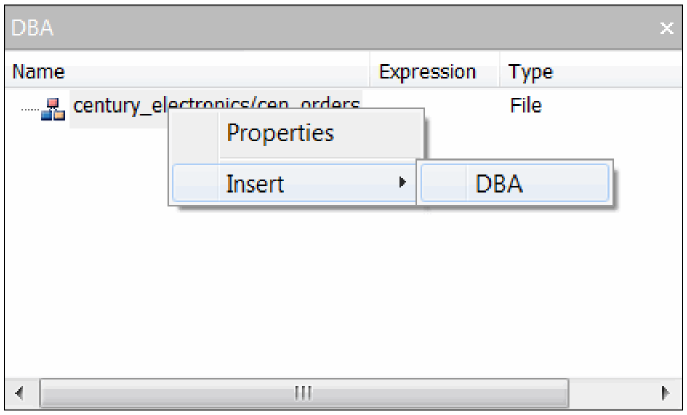


Figure 3-141 DBA - value rule - Step 1

4. A new rule is created, with the same name as the user profile you are using. Right-click the DBA rule and select **Insert** → **User**, as shown in Figure 3-142.

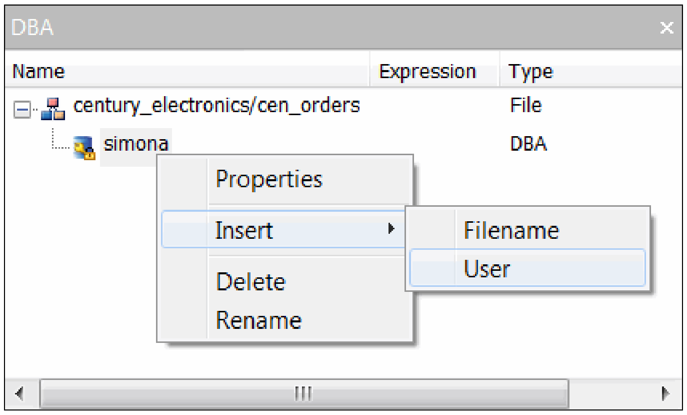


Figure 3-142 DBA - value rule - Step 2

5. A new entry is added, named “USER0001”. Right-click it and select **Rename** to change it to the name of the user for which you want to set the security rule. In this example, we will use “SEPTIA” (Figure 3-143).

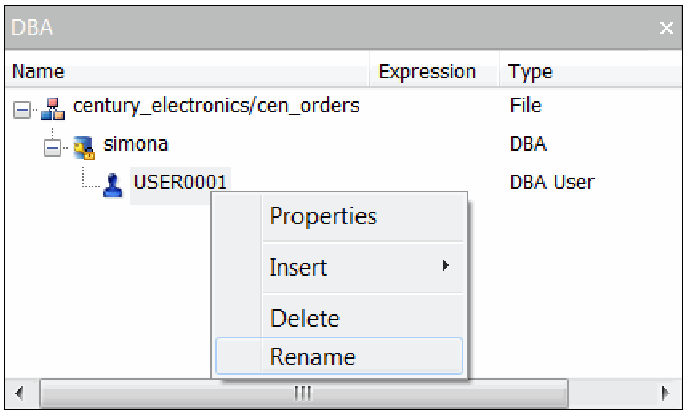


Figure 3-143 DBA - value rule - Step 3

6. Right-click the name of the rule and select **Insert** → **Read Access** (Figure 3-144).

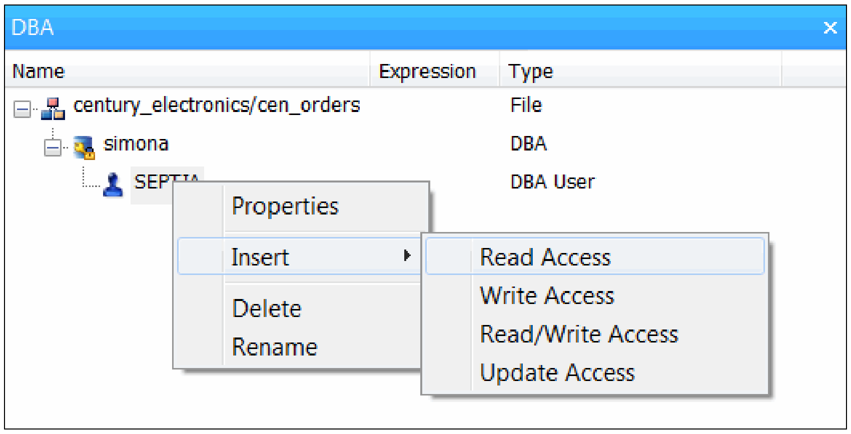


Figure 3-144 DBA - value rule - Step 4

7. Right-click **Read** and select **Insert** → **Value Restriction** (Figure 3-145).

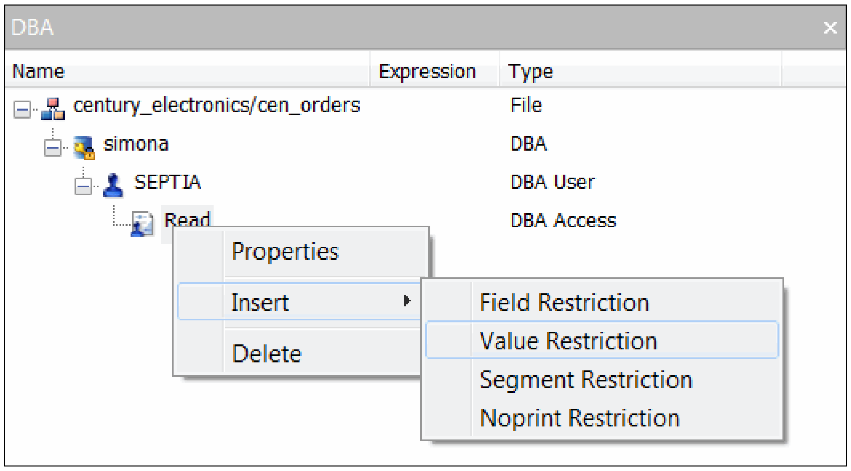


Figure 3-145 DBA - value rule - Step 5

8. A new item is inserted, represented by the function symbol and named "SYSTEM". Select it and move to the panel above. Use the little dotted button on the side of the VALUE field as shown in Figure 3-146.

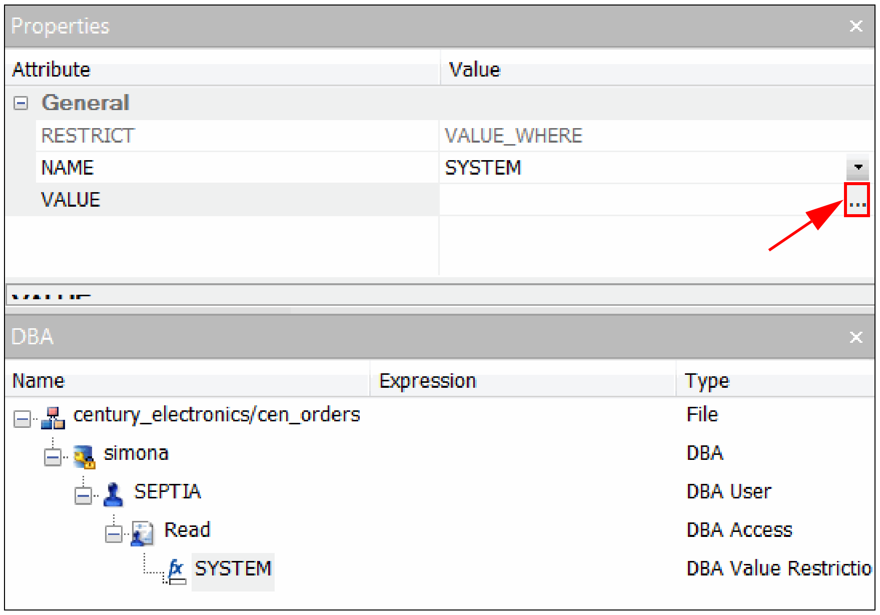
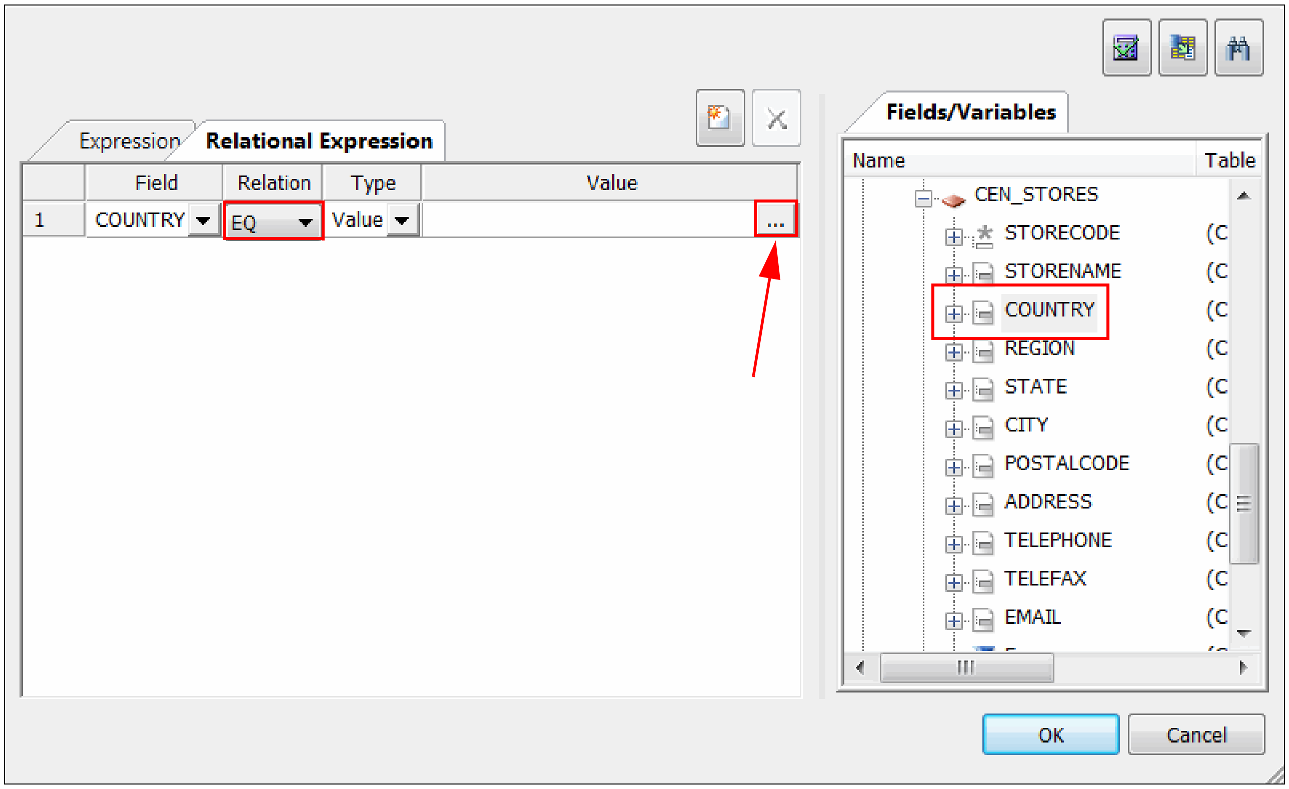


Figure 3-146 DBA - value rule - Step 6

9. You are presented with a panel where you can set up the rule. It offers you two choices. Expression is used to set up rules that requires operations or functions to be applied to data, Relational Expression is used to set up comparison to fields data. In our example, we want to check contain of a field and restrict usage of all the rows that have the selected value. Double-click the field you want to check, COUNTRY in our example. It will be inserted in the left-hand side of the panel. Select the comparison operator, EQ in our example, and ask for a list of values by clicking the dotted button as shown in Figure 3-147.

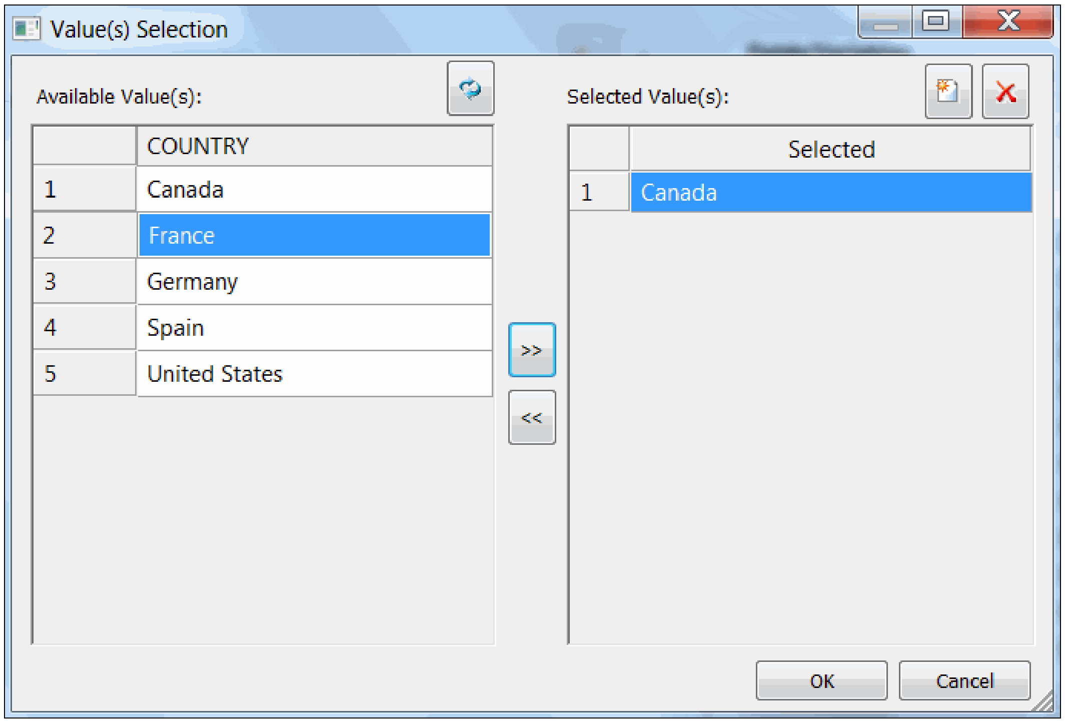


	Field	Relation	Type	Value
1	COUNTRY	EQ	Value	...

CEN_STORES
STORECODE (C
STORENAME (C
COUNTRY (C
REGION (C
STATE (C
CITY (C
POSTALCODE (C
ADDRESS (C
TELEPHONE (C
TELEFAX (C
EMAIL (C

Figure 3-147 DBA - value rule - Step 7

10. In the Value selection pane select the value for which you want to restrict access, Canada in our example, and then click **OK** (Figure 3-148).



Available Value(s):

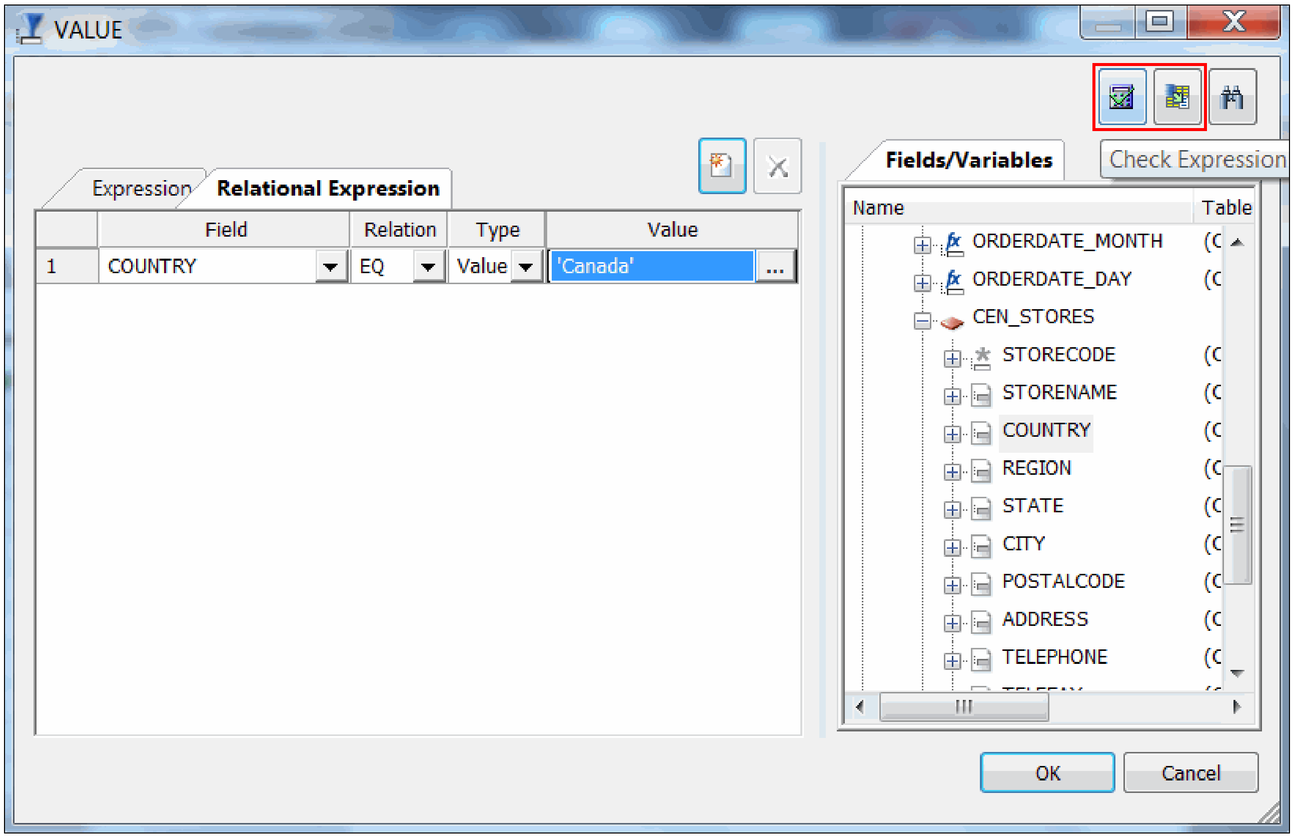
	COUNTRY
1	Canada
2	France
3	Germany
4	Spain
5	United States

Selected Value(s):

	Selected
1	Canada

Figure 3-148 DBA - value rule - Step 8

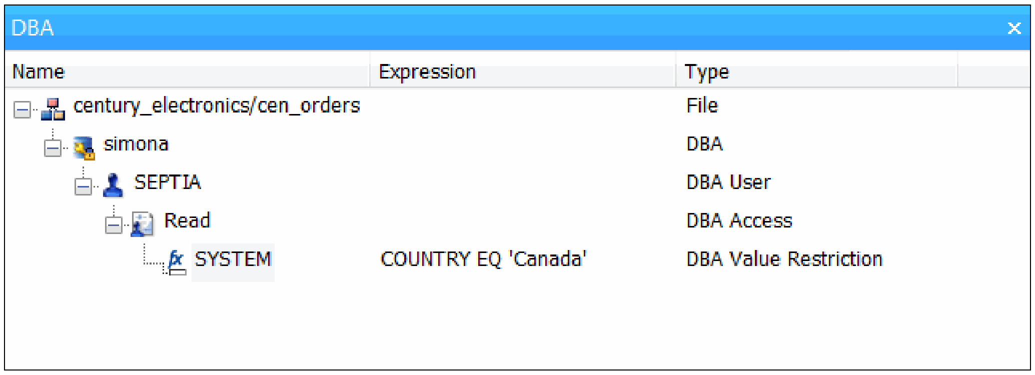
1. You will be brought back to the previous pane where the settings can be verified. If necessary, you can use the button on the top right-hand corner of the pane to check expression, sample data and find. Select **OK**.



	Field	Relation	Type	Value
1	COUNTRY	EQ	Value	'Canada'

Figure 3-149   DBA - value rule - Step 9

2. In the main DBA pane, you can see the whole rule (Figure 3-150).



DBA

Name	Expression	Type
century_electronics/cen_orders		File
simona		DBA
SEPTIA		DBA User
Read		DBA Access
SYSTEM	COUNTRY EQ 'Canada'	DBA Value Restriction

Figure 3-150   DBA - value rule completed

21.3 Using the adapter for JD Edwards EnterpriseOne

The adapter for JD Edwards EnterpriseOne allows DB2 Web Query to access JD Edwards EnterpriseOne data sources. With this adapter, data in the JD Edwards EnterpriseOne DBMS is displayed using rules contained in dictionary files, thereby ensuring that valid information is returned to the requesting program.

21.3.1 Preparing the JD Edwards EnterpriseOne environment

Although no environment preparation steps are required, ensure that your system complies with all software specifications.

Do not add this JD Edwards adapter to a Reporting Server which already contains a configured JD Edwards World adapter. If you would like to add this adapter to that server configuration, you must remove the existing JD Edwards World adapter first.

The only software requirement is to use a JD Edwards EnterpriseOne Application installation configured with DB2.

21.3.2 Overview of the setup process

The setup process for the adapter for JD Edwards EnterpriseOne is composed of the following basic steps:

1. Configure the adapter for JD Edwards EnterpriseOne. Define what type of security to implement.
2. Refresh the metadata repository. You will need to perform this step initially, and repeat it only if there are changes in the JD Edwards metadata tables. This occurs infrequently at most sites.
3. Refresh the security extract. This step is required only if Group or Role based security is configured in step 1. This captures the current JD Edwards EnterpriseOne rules for the adapter to enforce.
4. Create the JD Edwards EnterpriseOne synonyms. Synonyms are required for Web Query reporting against a data source.

21.3.3 Configuring the adapter for JD Edwards EnterpriseOne

In this chapter, we explain how to configure the Adapter for JD Edwards EnterpriseOne. Follow these steps:

1. Log on to DB2 Web Query as the DB2 Web Query (user profile with QWQADMIN group) administrator ID.

   The administrator is the user that configures and manages the adapter configuration; other users are not permitted to manage and configure adapters.

2. In the Reporting area, create folder “Assignment 15 - Creating JD Edwards reports”. In this tutorial, the folder is created under the Century Electronics folder as in Figure 21-6.

   If you already have this folder created, you can skip this step.

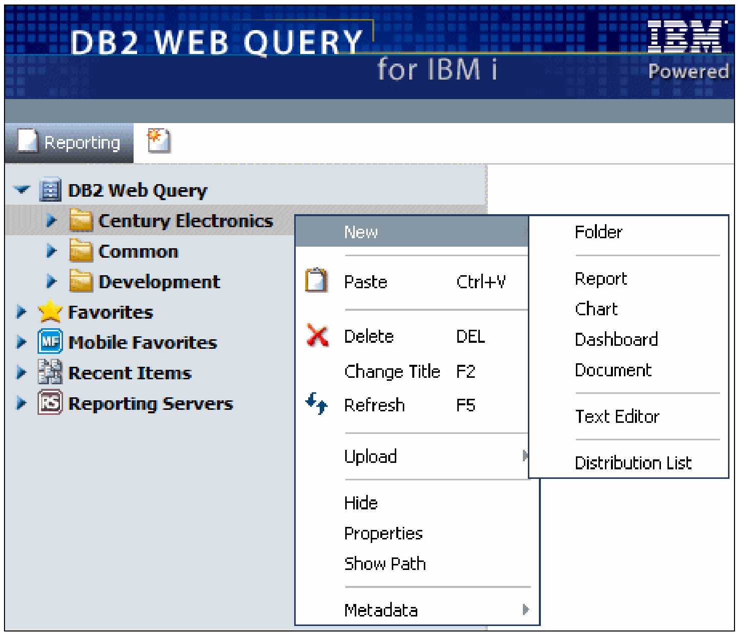


Figure 21-6 Menu to create folder

3. Expand **DB2 Web Query** → **Century Electronics** → **Assignment 15 - Creating JD Edwards reports**. Right-click the **Assignment 15 - Creating JD Edwards reports** folder, and select **Metadata** → **Edit** as in Figure 21-7.

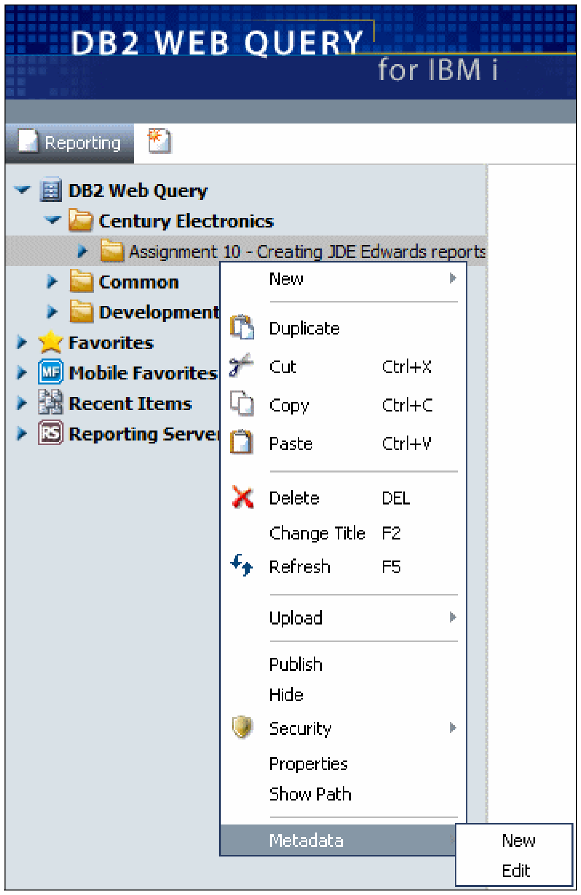


Figure 21-7 Menu to edit metadata

4. In the new opened window, click the **Adapter** to open the Adapter tab, as in Figure 21-8

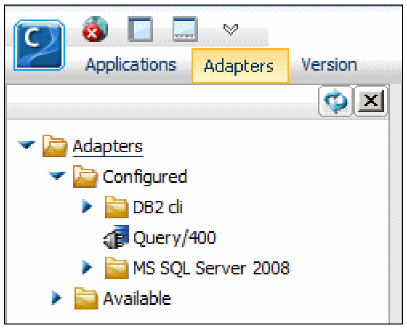
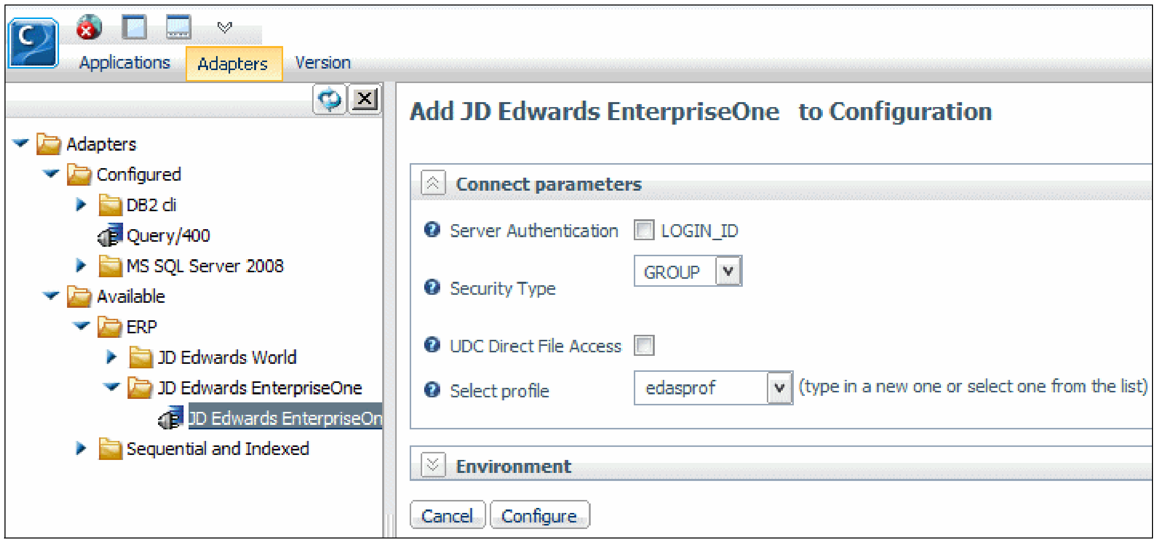


Figure 21-8 Open the Adapter tab

5. In the left-hand Adapter navigation pane, expand the Available folder.
6. Expand the ERP folder.
7. Expand the JD Edwards EnterpriseOne folder.

8. Double-click **JD Edwards EnterpriseOne**; the configuration window displays as shown in Figure 21-9.



Add JD Edwards EnterpriseOne to Configuration

Connect parameters

Server Authentication ☐ LOGIN_ID

Security Type: GROUP

UDC Direct File Access ☐

Select profile: edasprof (type in a new one or select one from the list)

Figure 21-9 JD Edward EnterpriseOne configuration

9. Select the connection parameters:

   a. Server Authentication:

      Check this box if the reporting server is secured. This option applies when every JD Edwards EnterpriseOne user has a user ID on the reporting server system as is the case in Web Query.

   b. Security Type:

      When you configure the adapter for JD Edwards EnterpriseOne, you must choose if your JDE environment is configured to use role-, group-based security, or no security (NONE).

   c. UDC Direct File Access:

      When you select this check box, you give users access to the User Defined Code Direct File.

   d. Select Profile:

      You must choose edasprof.

10. Click **Configure**.

    You will receive a confirmation message.

    Important: The reporting server agents will be stopped. You need to confirm that no Web Query jobs are running before clicking OK. Restarting the Reporting Server disconnects any users currently working in DB2 Web Query.

11. Click **OK**.

21.3.4 Refreshing the metadata repository

The Metadata repository contains the dictionary information for the JD Edwards EnterpriseOne tables.

You must refresh the repository the first time you set up the adapter and repeat the process each time the JD Edwards EnterpriseOne dictionary tables change.

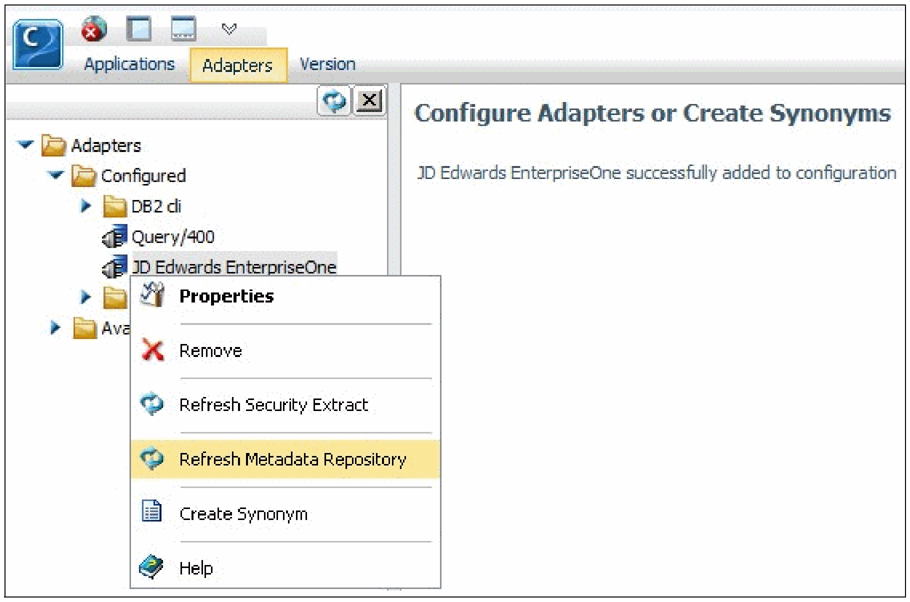
How to refresh the metadata repository

You can refresh the metadata repository from the Adapters list in the navigation pane on the web page:

1. Right-click the configured JD Edwards EnterpriseOne adapter and select Refresh Metadata Repository as in Figure 21-10.

   You will need to perform this step initially, and repeat it only if there are changes in the metadata for tables. This occurs infrequently at most sites.

   The Refresh Metadata Repository pane opens. The JDE tables required for this procedure are listed in the first column.



Configure Adapters or Create Synonyms

JD Edwards EnterpriseOne successfully added to configuration

Figure 21-10 Menu to refresh metadata repository

2. Enter the Library name of the library containing the specified objects. The UDC library can be any arbitrary name, for example, UDCDIC.
3. Click **Refresh Now** to refresh the metadata repository.

   Once the refresh has completed, the metadata repository has been successfully refreshed.

21.3.5 Refreshing the security extract

The security extract is a snapshot of the security rules defined in JD Edwards EnterpriseOne.

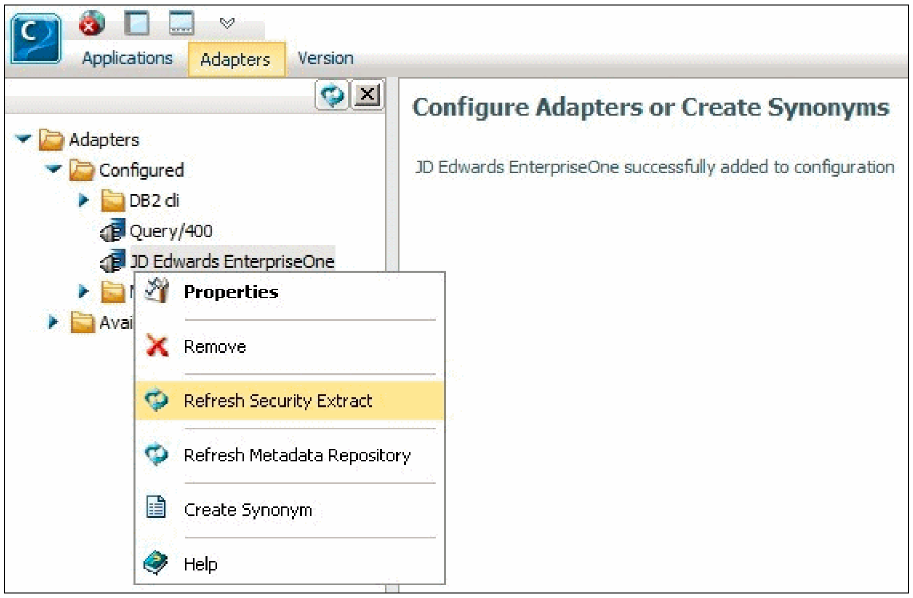
The adapter uses this extract to enforce the restrictions when reporting against JD Edwards EnterpriseOne data. You must refresh the security extract as often as the security rules change in the JD Edwards EnterpriseOne application.

How to refresh the security extract

In order to refresh the security extract, navigate to the Adapters list in the navigation pane on the web page:

1. Right-click the configured JD Edwards Enterprise One adapter and select Refresh Security Extract as in Figure 21-11.

> **Note:** You will need to perform this step upon initial configuration of the adapter, and repeat it only if there are changes in the JD Edwards EnterpriseOne security rules.



Configure Adapters or Create Synonyms

JD Edwards EnterpriseOne successfully added to configuration

Figure 21-11 Menu to refresh security extract

2. Enter the Library name of the library containing the specified objects.

3. Click **Submit** to refresh the security extract.

21.3.6 Creating the JD Edwards EnterpriseOne synonyms

To report against JD Edwards EnterpriseOne data, you must first create synonyms.

How to create the JD Edwards EnterpriseOne synonyms

Follow these steps to create the synonyms:

1. Right-click **JD Edwards EnterpriseOne**, and select **Create Synonym**.
2. Click the DB2 CLI connection that points to your JD Edwards EnterpriseOne data tables.
3. Select the restrictions you would like to apply when searching for synonym candidates. Restriction options included are restrict object type, further restricting Tables, Views, Aliases, and MQTs.
4. Click **Next**.
5. Select the parameters you would like the synonym to include.

   Parameters options include With foreign keys, One-part name, Application, Prefix, Suffix, and Overwrite existing synonyms.

a. With foreign keys:

   Select the **With foreign keys** check box to include within this synonym every table related to the current table by a foreign key. The resulting multi-table synonym describes all of these tables' foreign key relationships.

b. One-part name:

   On the IBM i platform, the One-part name check box is unchecked by default. The unchecked behavior generates a table name that includes the explicit name of the library containing the table. For example, if you specified a library on the first Create Synonym pane, a qualified name like the following is automatically created in the Access File:

   TABLENAME=MYLIB/MYTABLE

   With this explicit type of entry in the Access File, at runtime, the library is directly located and searched for the table name. If you select the check box, the explicit library name is not stored in the metadata (Access File). When the synonym is generated, the library portion of the table name is omitted from the Access File, and appears like this:

   TABLENAME=MYTABLE

   With this type of entry in the Access File, at runtime, the library path of the user is searched until the table name is located.

c. Application:

   This defaults to the first application folder in the application path.

d. Prefix/Suffix:

   If you have tables with identical table names, assign a prefix or a suffix to distinguish them. For example, if you have identically named human resources and payroll tables, assign the prefix HR to distinguish the synonyms for the human resources tables. Note that the resulting synonym name cannot exceed 64 characters. If all tables and views have unique names, leave prefix and suffix fields blank.

e. Overwrite existing synonym:

   To specify that this synonym should overwrite any earlier synonym with the same fully qualified name, select the Overwrite existing synonyms check box.

   Note: The connected user must have operating system write privileges in order to recreate a synonym.

6. Select the check box next to table(s) that you want to create synonyms for.
7. Click **Create synonym**.
8. Add JD Edwards dictionary information to the synonym:

   a. Select date format:

      The options are: YMD, YYMD, DMY, MDY, MDYY, DMYY, MYY, YYM. (YYMD is the default setting.) The selected format will be used only if the field is described as a DATE in the Data Dictionary.

   b. UDC:

      Check the UDC box to ensure that UDC description fields are generated as DEFINEs (virtual fields) in the synonym. Checked (ON) is the default setting.

   c. Combine UDC:

      Check this box to Combine User Defined Code. Unchecked (OFF) is the default setting.

9. Click **Continue**.

   The synonym has been successfully created.

21.4 Developing a JD Edwards report

This section will highlight the JD Edwards synonym using Info Assist. We will use the F0911 table (Account Ledger) for this section of the tutorial and assumes you have completed all the prior sections of this tutorial.

Benefits of JD Edwards adapter

These are the benefits of the JD Edwards adapter:

- Proper decimal notation
- Automatic UDC lookups
- Julian to Gregorian date conversion
- User friendly column titles
- Leveraging Presumptive Joins (World only) and Security definitions

This tutorial will highlight the first four benefits.

Developing a JD Edwards Report

Here is how to develop a JD Edwards Report:

1. Log on to DB2 Web Query.
2. Expand **DB2 Web Query** → **Century Electronics** → **Assignment 150 - Creating JD Edwards reports**. Right-click the **Assignment 15 - Creating JD Edwards reports** folder, and select **New** → **Report** as in Figure 21-12.

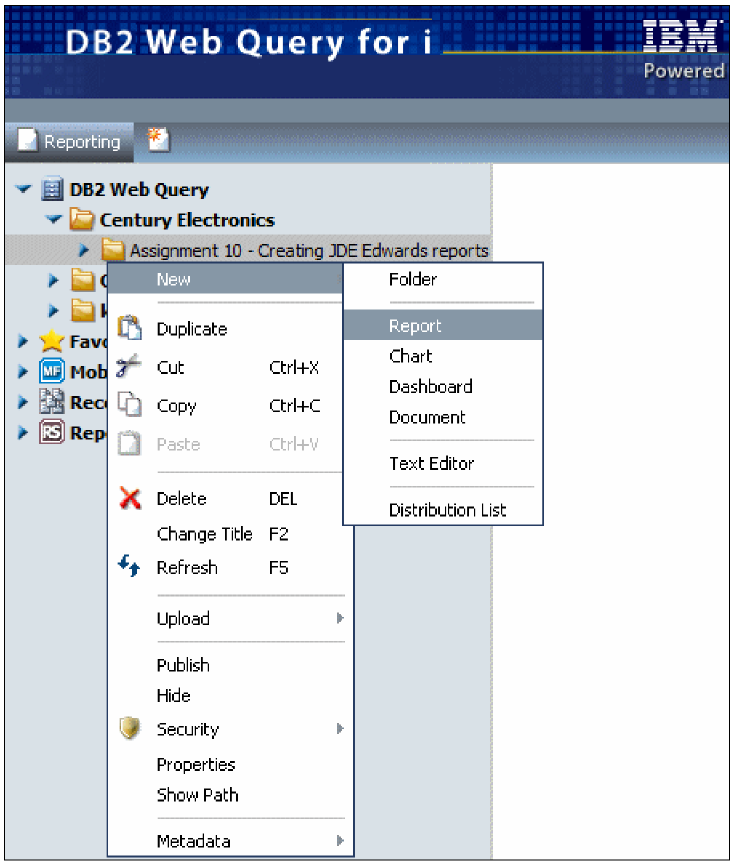
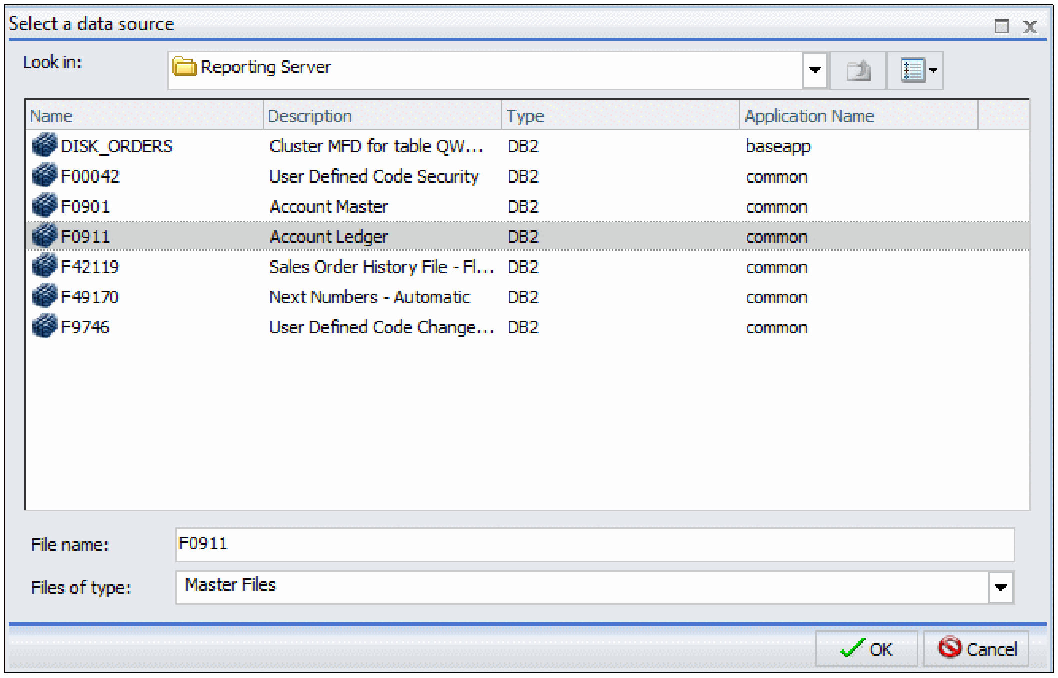


Figure 21-12 Menu to Create Report

3. Select the JD Edwards F0911 synonym in the **Select data source** dialog window. See Figure 21-13.



Name	Description	Type	Application Name
DISK_ORDERS	Cluster MFD for table QW…	DB2	baseapp
F00042	User Defined Code Security	DB2	common
F0901	Account Master	DB2	common
F0911	Account Ledger	DB2	common
F42119	Sales Order History File - Fl…	DB2	common
F49170	Next Numbers - Automatic	DB2	common
F9746	User Defined Code Change…	DB2	common

File name: F0911

Files of type: Master Files

Figure 21-13 Select Synonym

By default, you are presented with the Logical view of the field list from F0911. The Logical view arranges your fields by Dimension and Measures. Your fields are sorted such that all your numeric fields are grouped under a heading called Measures and all the character fields are grouped under Dimensions, as in Figure 21-14.

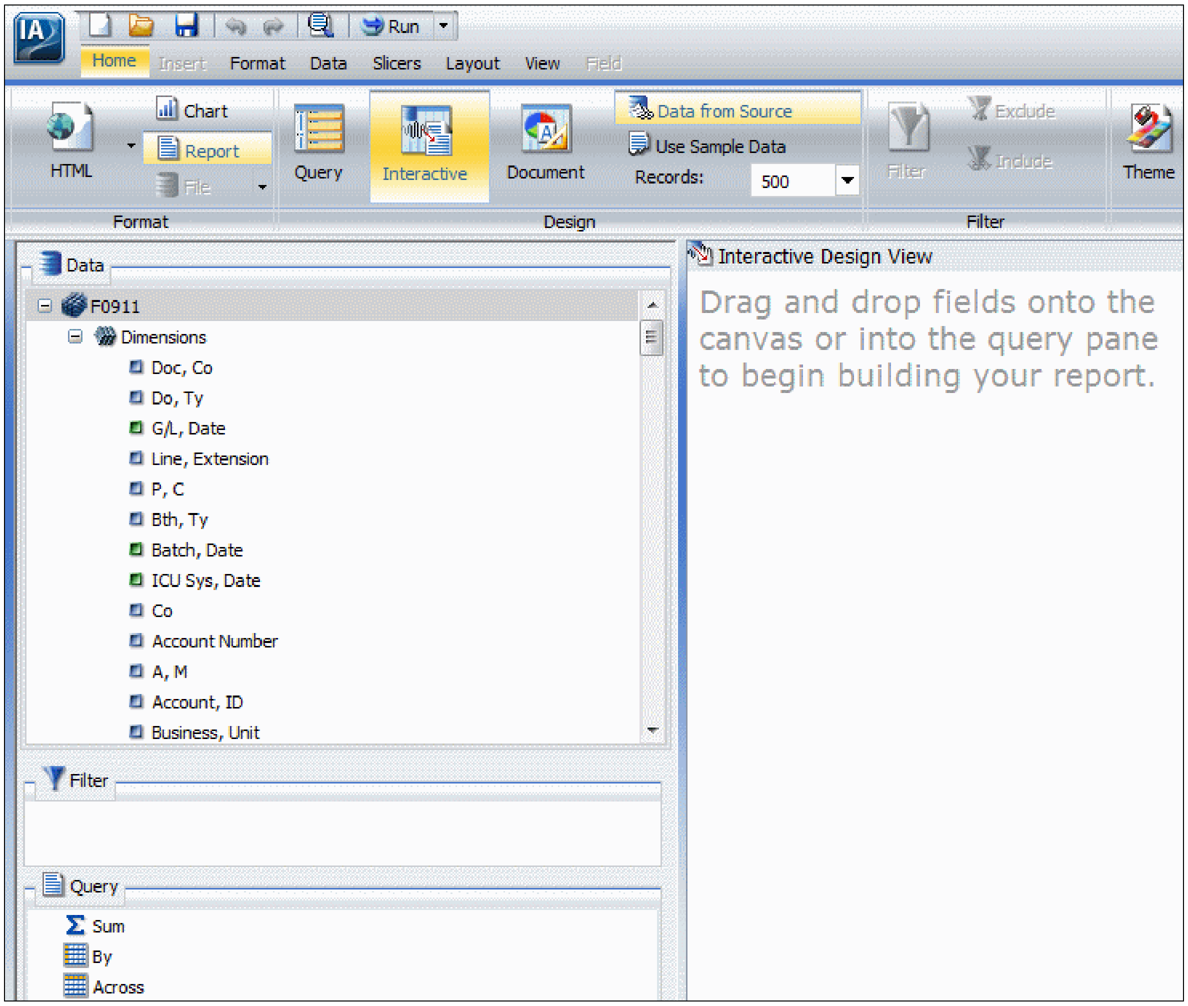


Figure 21-14 Logical view of the field list

4. If you want display filed names, you can set this by clicking the **Logical** icon on the View ribbon and selecting **Field**. See Figure 21-15.

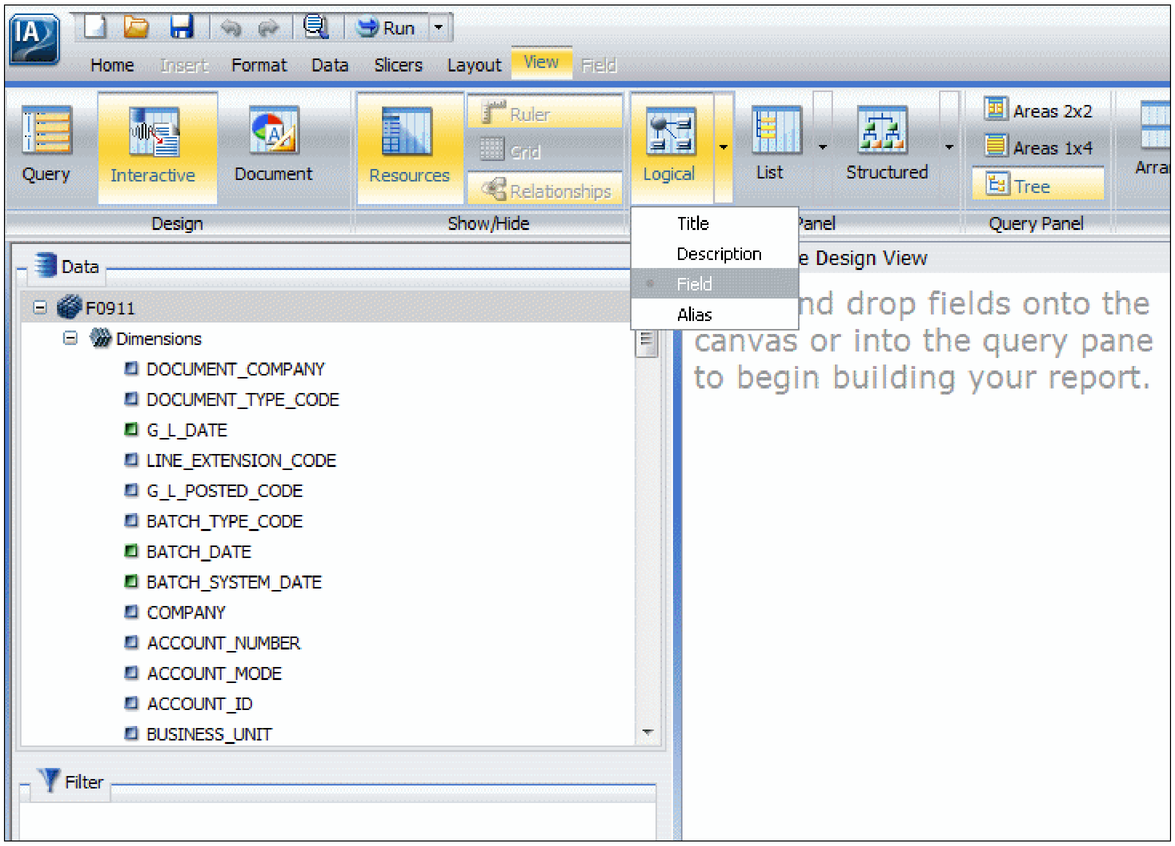
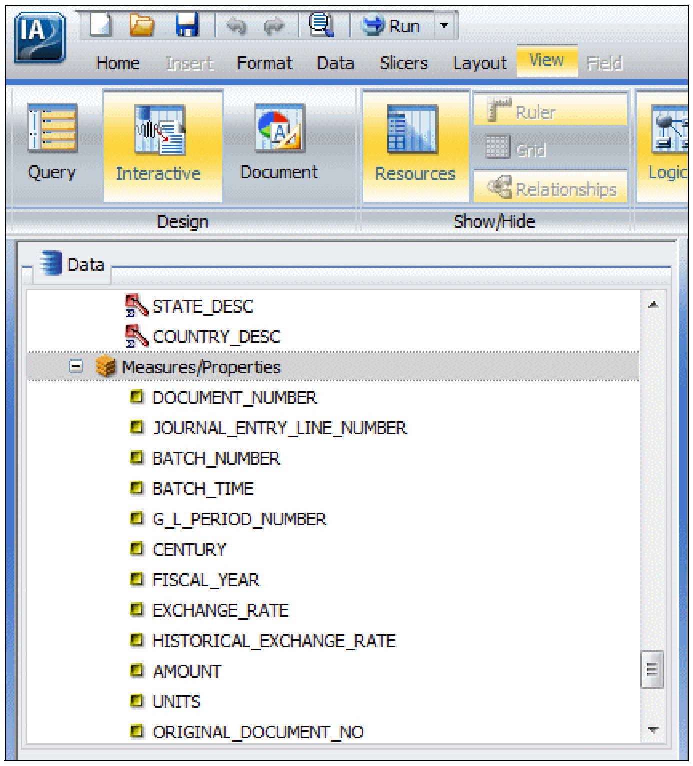


Figure 21-15 Display field names

5. Scroll down the field list and notice the UDC (User Defined Codes) fields. These represent the descriptive text for key fields in the file.

6. Scroll down further and notice the Measures associated with this file, as in Figure 21-16.

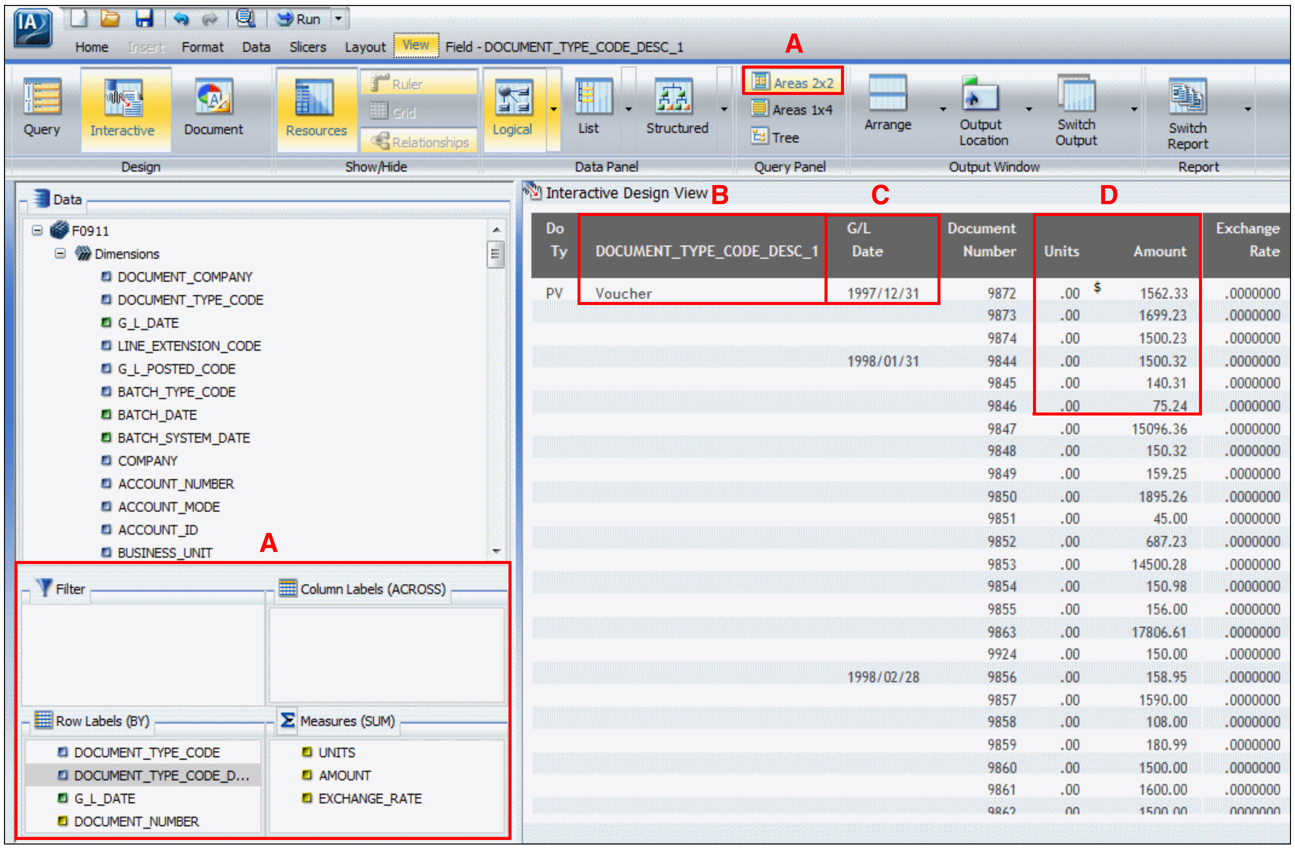


STATE_DESC
COUNTRY_DESC
Measures/Properties
DOCUMENT_NUMBER
JOURNAL_ENTRY_LINE_NUMBER
BATCH_NUMBER
BATCH_TIME
G_L_PERIOD_NUMBER
CENTURY
FISCAL_YEAR
EXCHANGE_RATE
HISTORICAL_EXCHANGE_RATE
AMOUNT
UNITS
ORIGINAL_DOCUMENT_NO

Figure 21-16 Measures fields

7. Select some of each of the aforementioned fields and build a report. You can double-click fields, or drag and drop them on to your Interactive Design View panel to build the report. Info Assist will put dimensions as Sort fields and measures as Measure fields where you can aggregate as required. The other option is to drag fields into a specific area of the Query Panel.

   Figure 21-17 shows the Interactive Design view using Data from Source. It represents what your report will look like at runtime using data from the JD Edwards data source.



Do Ty	DOCUMENT_TYPE_CODE_DESC_1	G/L Date	Document Number	Units	Amount	Exchange Rate
PV	Voucher	1997/12/31	9872	.00	1562.33	.0000000
			9873	.00	1699.23	.0000000
			9874	.00	1500.23	.0000000
		1998/01/31	9844	.00	1500.32	.0000000
			9845	.00	140.31	.0000000
			9846	.00	75.24	.0000000
			9847	.00	15096.36	.0000000
			9848	.00	150.32	.0000000
			9849	.00	159.25	.0000000
			9850	.00	1895.26	.0000000
			9851	.00	45.00	.0000000
			9852	.00	687.23	.0000000
			9853	.00	14500.28	.0000000
			9854	.00	150.98	.0000000
			9855	.00	156.00	.0000000
			9863	.00	17806.61	.0000000
			9924	.00	150.00	.0000000
		1998/02/28	9856	.00	158.95	.0000000
			9857	.00	1590.00	.0000000
			9858	.00	108.00	.0000000
			9859	.00	180.99	.0000000
			9860	.00	1500.00	.0000000
			9861	.00	1600.00	.0000000
			9862	.00	1500.00	.0000000

Figure 21-17 Build a report

   a. Notice that the Query Panel to be 2x2 has been changed by selecting the icon from the Query Panel group on the View ribbon. The icon on the View ribbon and the Query Panel group are both highlighted with boxes labeled A in Figure 21-17.

   b. Notice the UDC field, DOCUMENT_TYPE_CODE_DESC_1, is provided. In this case, it is giving the descriptive name for Document Type. This is highlighted in the box labeled B in Figure 21-17.

   c. Notice that the Date is converted to a Gregorian date YYMD. This is highlighted in the box labeled C in Figure 21-17.

   d. Notice the decimal precision on the numeric columns as well as the friendly column titles. This is highlighted in the box labeled D in Figure 21-17.

8. There are a couple of tasks you can perform to enhance your report before running it:
   a. UDC field titles can be changed by right-clicking the UDC field in the Interactive Design View and selecting the **Change Title…** option as in Figure 21-18.

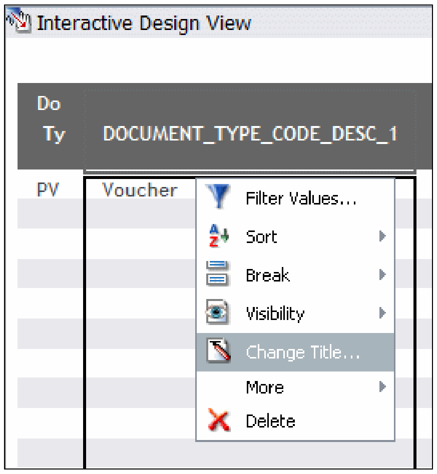
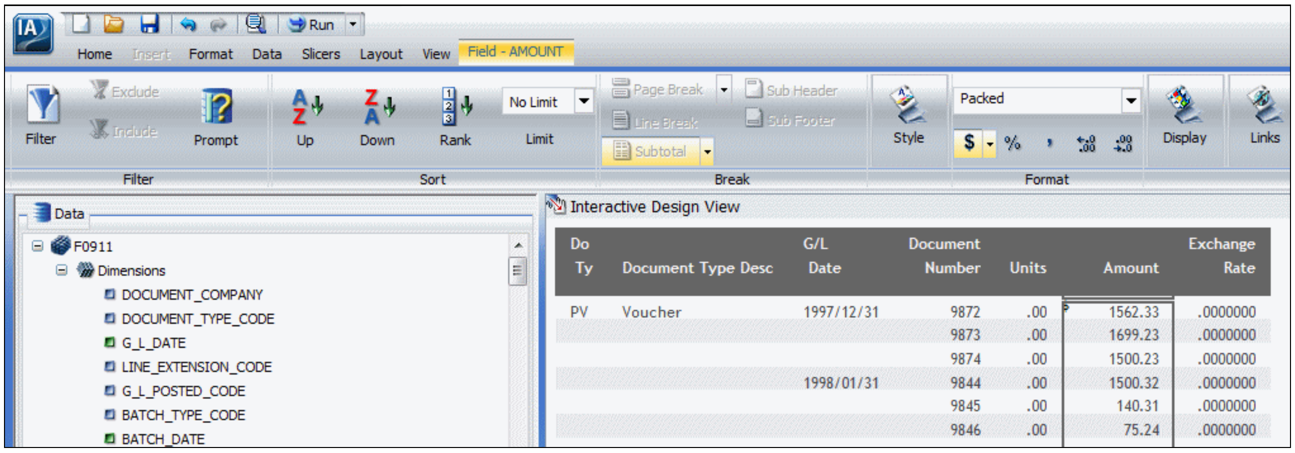


Figure 21-18 Menu to change title

A dialog box will appear to allow you to enter in a new column title.

   b. Currency fields can be easily formatted with floating currency symbols. Select a numeric field on the Interactive Design View. Let's select Amount. Notice the Field ribbon appears for Amount, as in Figure 21-19.



Do Ty	Document Type Desc	G/L Date	Document Number	Units	Amount	Exchange Rate
PV	Voucher	1997/12/31	9872	.00	1562.33	.0000000
			9873	.00	1699.23	.0000000
			9874	.00	1500.23	.0000000
		1998/01/31	9844	.00	1500.32	.0000000
			9845	.00	140.31	.0000000
			9846	.00	75.24	.0000000

Figure 21-19 Select Amount field

Use the Format group on the Field ribbon to add floating currency and commas by clicking the respective icons. An item that is highlighted in yellow means the option is turned on. See Figure 21-20.

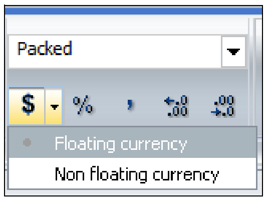


Figure 21-20 Add floating currency

To do more advance formatting, click the drop-down and select **More options…** See Figure 21-21.

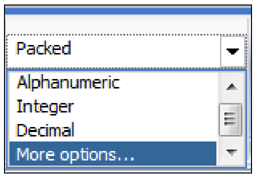
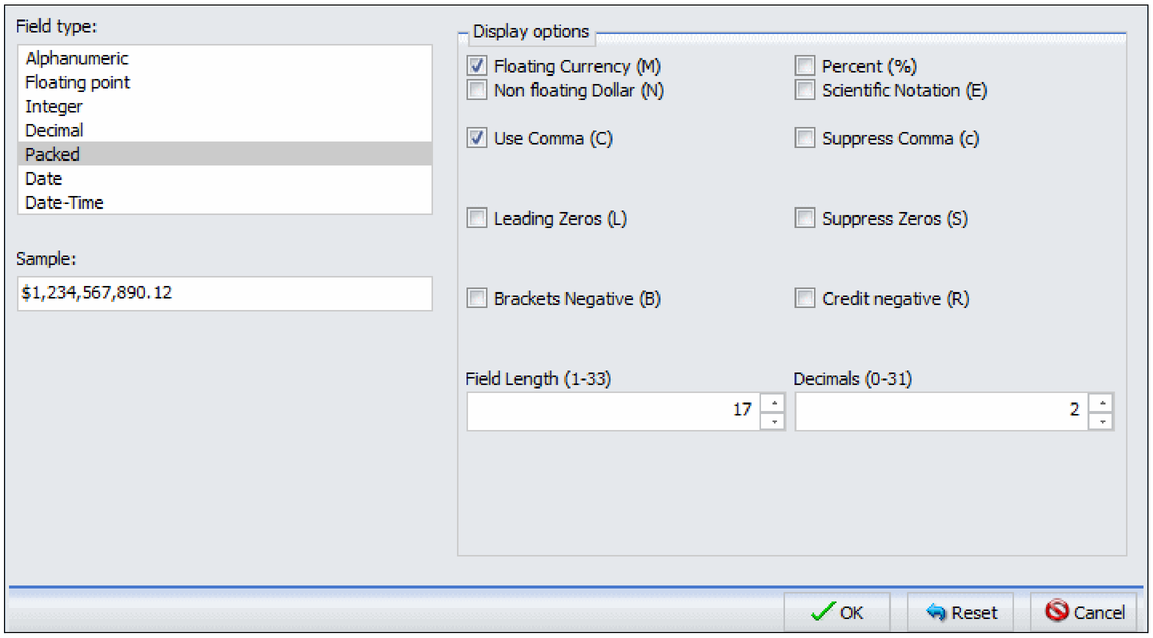


Figure 21-21 Menu to more options

Change decimal precision to 2. See Figure 21-22.



Sample: $1,234,567,890.12

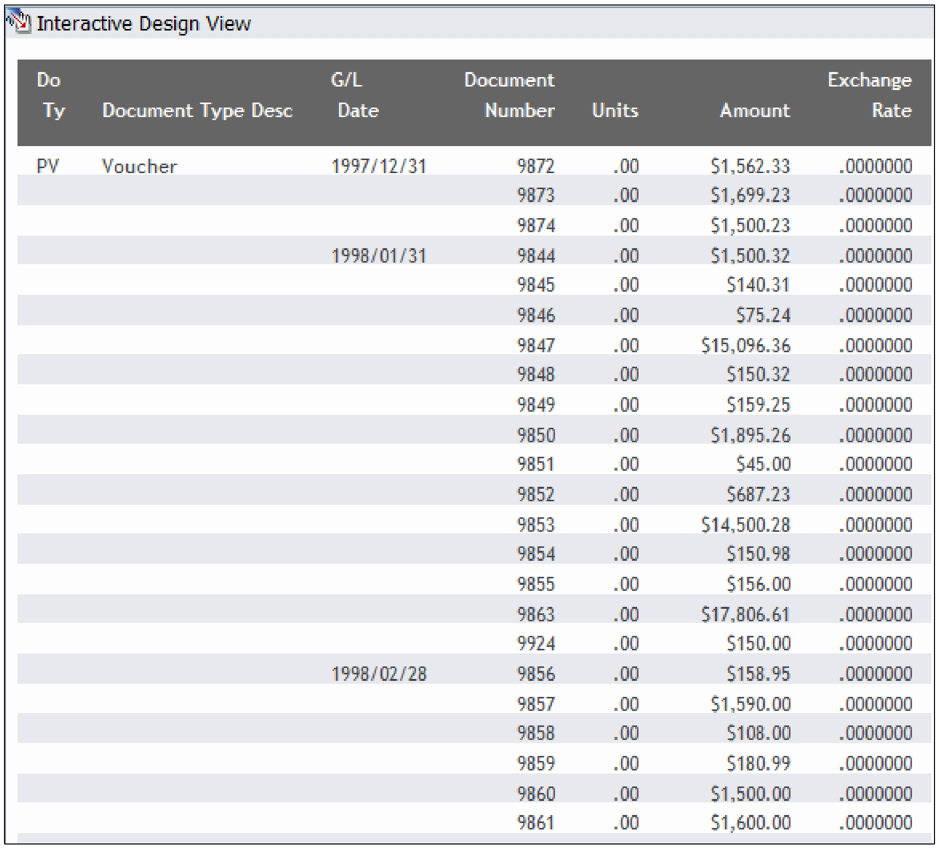
- [✓] Floating Currency (M)
- [ ] Percent (%)
- [ ] Non floating Dollar (N)
- [ ] Scientific Notation (E)
- [✓] Use Comma (C)
- [ ] Suppress Comma (c)
- [ ] Leading Zeros (L)
- [ ] Suppress Zeros (S)
- [ ] Brackets Negative (B)
- [ ] Credit negative (R)

Field Length (1-33): 17

Decimals (0-31): 2

Figure 21-22 Window to change field format

The Interactive Design View dynamically reflects this change, as in Figure 21-23.



Interactive Design View

Do Ty	Document Type Desc	G/L Date	Document Number	Units	Amount	Exchange Rate
PV	Voucher	1997/12/31	9872	.00	$1,562.33	.0000000
			9873	.00	$1,699.23	.0000000
			9874	.00	$1,500.23	.0000000
		1998/01/31	9844	.00	$1,500.32	.0000000
			9845	.00	$140.31	.0000000
			9846	.00	$75.24	.0000000
			9847	.00	$15,096.36	.0000000
			9848	.00	$150.32	.0000000
			9849	.00	$159.25	.0000000
			9850	.00	$1,895.26	.0000000
			9851	.00	$45.00	.0000000
			9852	.00	$687.23	.0000000
			9853	.00	$14,500.28	.0000000
			9854	.00	$150.98	.0000000
			9855	.00	$156.00	.0000000
			9863	.00	$17,806.61	.0000000
			9924	.00	$150.00	.0000000
		1998/02/28	9856	.00	$158.95	.0000000
			9857	.00	$1,590.00	.0000000
			9858	.00	$108.00	.0000000
			9859	.00	$180.99	.0000000
			9860	.00	$1,500.00	.0000000
			9861	.00	$1,600.00	.0000000

Figure 21-23 Interactive Design View of the report

c. Add a subtotal on each DOCUMENT_TYPE_CODE_DESC_1. On the Interactive Design View, click the DOCUMENT_TYPE_CODE_DESC_1 field and click the Subtotal icon in the Break group on the DOCUMENT_TYPE_CODE_DESC_1 Field ribbon as in Figure 21-24.

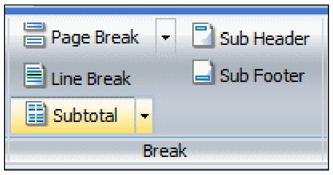
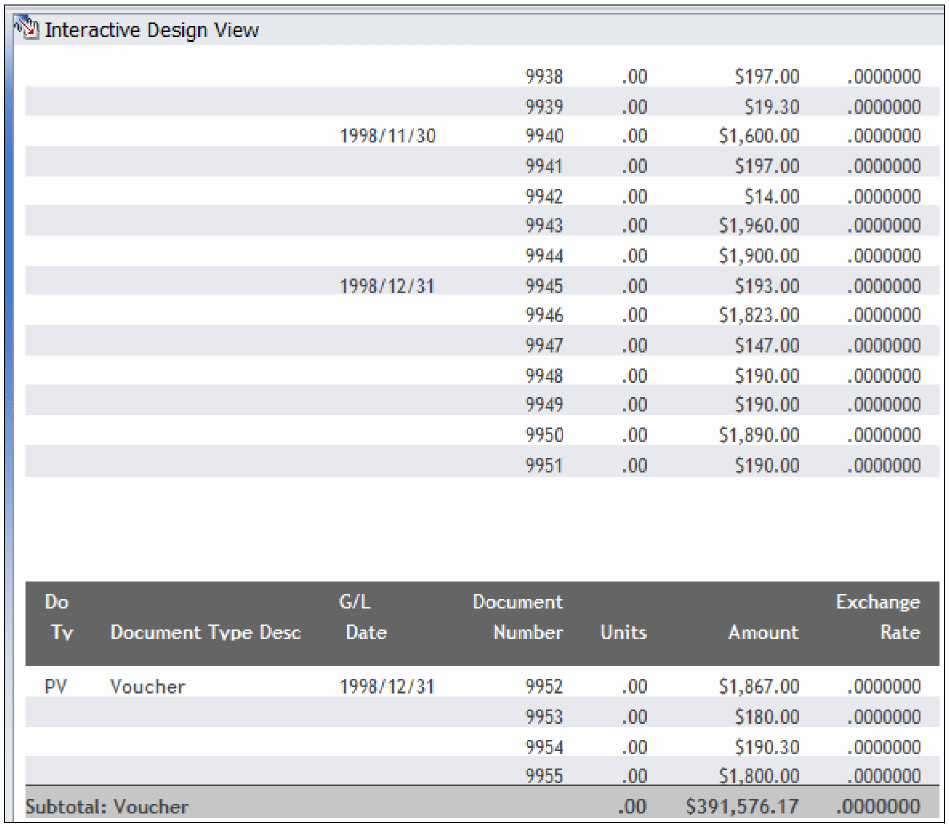


Figure 21-24 Button to add Subtotal

Notice that a subtotal is added dynamically to the Interactive Design View, as in Figure 21-25.


Interactive Design View

			9938	.00	$197.00	.0000000
			9939	.00	$19.30	.0000000
		1998/11/30	9940	.00	$1,600.00	.0000000
			9941	.00	$197.00	.0000000
			9942	.00	$14.00	.0000000
			9943	.00	$1,960.00	.0000000
			9944	.00	$1,900.00	.0000000
		1998/12/31	9945	.00	$193.00	.0000000
			9946	.00	$1,823.00	.0000000
			9947	.00	$147.00	.0000000
			9948	.00	$190.00	.0000000
			9949	.00	$190.00	.0000000
			9950	.00	$1,890.00	.0000000
			9951	.00	$190.00	.0000000

Do Ty	Document Type Desc	G/L Date	Document Number	Units	Amount	Exchange Rate
PV	Voucher	1998/12/31	9952	.00	$1,867.00	.0000000
			9953	.00	$180.00	.0000000
			9954	.00	$190.30	.0000000
			9955	.00	$1,800.00	.0000000
Subtotal: Voucher				.00	$391,576.17	.0000000

Figure 21-25 Report with subtotal

9. Save your report as **15a - JD Edwards report**.

10. Click the **Run** button on the Quick Access Toolbar to run your report, as in Figure 21-26.

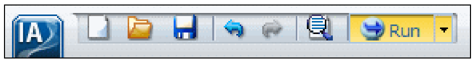
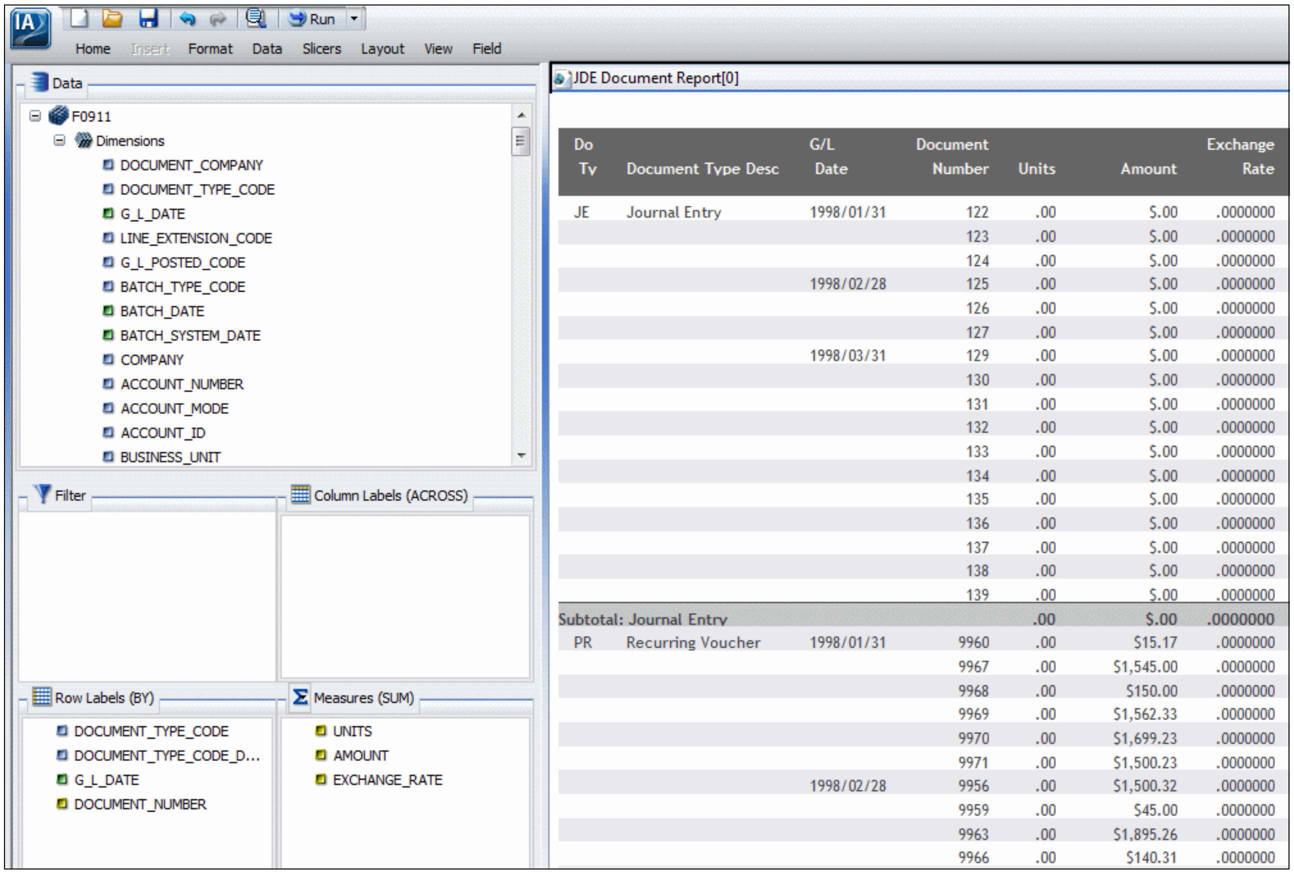


Figure 21-26 Run button

Depending on your Info Assist settings, your report output will appear in a tab or window. In this example, Info Assist directs the output to a single tab. See Figure 21-27.



Do Ty	Document Type Desc	G/L Date	Document Number	Units	Amount	Exchange Rate
JE	Journal Entry	1998/01/31	122	.00	$.00	.0000000
			123	.00	$.00	.0000000
			124	.00	$.00	.0000000
		1998/02/28	125	.00	$.00	.0000000
			126	.00	$.00	.0000000
			127	.00	$.00	.0000000
		1998/03/31	129	.00	$.00	.0000000
			130	.00	$.00	.0000000
			131	.00	$.00	.0000000
			132	.00	$.00	.0000000
			133	.00	$.00	.0000000
			134	.00	$.00	.0000000
			135	.00	$.00	.0000000
			136	.00	$.00	.0000000
			137	.00	$.00	.0000000
			138	.00	$.00	.0000000
			139	.00	$.00	.0000000
Subtotal: Journal Entry				.00	$.00	.0000000
PR	Recurring Voucher	1998/01/31	9960	.00	$15.17	.0000000
			9967	.00	$1,545.00	.0000000
			9968	.00	$150.00	.0000000
			9969	.00	$1,562.33	.0000000
			9970	.00	$1,699.23	.0000000
			9971	.00	$1,500.23	.0000000
		1998/02/28	9956	.00	$1,500.32	.0000000
			9959	.00	$45.00	.0000000
			9963	.00	$1,895.26	.0000000
			9966	.00	$140.31	.0000000

Figure 21-27 Run the report

This tutorial guides you through your first JD Edwards report. As you can see, configuring the JD Edwards adapter, managing JD Edward synonyms, and developing a report with Web Query Info Assist is very similar to any other data source in Web Query.

22

The Century Challenge BI solution: Postmortem

In this chapter, we look back on the Century Challenge BI solution and provide the results of the challenge.

22.1 Summary

In the previous chapters in Part 2, you completed a variety of assignments related to the Century Challenge BI solution. In each chapter you were given assignments by a skeptical executive who did not believe in the IBM i plaform, the DB2 for i database, or the DB2 Web Query product. In the end, you delivered on all of the assignments and in many cases, you exceeded Dan's expectations. All the reports, charts, dashboards, and scheduling requirements were delivered, and all within the 2-month period.

As a result of your efforts, Dan decided that you and Mel were correct: that the IBM i platform is powerful, modern, secure, stable, reliable, and leading edge. And that it would be foolish for a business to ever move off of such a platform. He humbly accepted defeat in the challenge, but in the process, he became an ardent and enthusiastic supporter of the IBM i. In fact, a few short months later, he placed the order for a new Power 7 model, and initiated projects to look at migrating other business applications and databases *to* the IBM i platform. He fired Fitzgerald (his caddy at the country club), citing an abundance of bad IT advice and lack of moral support. He even aspires to speak at midrange user conferences in the future to present a case studies on Century Challenge BI solution and the migration projects. What a difference two months makes!

Figure 22-1 The "new" Executive Dan (beard now gone)

Part 3

Miscellaneous topics

In this part of the book, we discuss miscellaneous topics regarding DB2 Web Query such as these:

- Migration considerations
- An IBM i Business Intelligence solution
- An adapter for Microsoft SQL Server

23

Performance case study

Minimizing the processing time for report requests and maintaining the accuracy and integrity of the report data are major goals for most organizations. In this chapter, we discuss DB2 Web Query performance considerations and recommendations. We include a case study that is intended to demonstrate steps that you can take to tune a DB2 Web Query report.

23.1 Performance basics

When the objective is to obtain optimal report performance, keep in mind the following DB2 Web Query characteristics regarding how it approaches performance optimization:

- When a DB2 Web Query report is run, the tasks of row selection, joining, ordering, and aggregating must be performed. These tasks can be carried out by the DB2 for i database engine, the DB2 Web Query Reporting Server, or a combination of both.
- DB2 Web Query attempts to translate the source code of a DB2 Web Query report into equivalent SQL statements. When the report is run, the SQL statement is submitted to the database engine for processing.

  Attention: SQL translation does not occur for reports that go through the Query/400 or DB Heritage Files adapter. Those adapters use different methods to access the data.

- Like most other relational databases management systems (RDBMS), the DB2 for i database engine knows its own data. Its optimizer component uses complex costing algorithms to determine the ideal access plan, which indexes to use, and so on, so that it can retrieve the data in the most efficient manner. To aid in its decision-making processes, it maintains various forms of statistics about the data. In addition, much of the database code lies below the Machine Interface (MI) layer of IBM i, which is an attribute that can yield great efficiencies during the query optimization and execution steps.
- For accomplishing all processing tasks in an efficient manner, the database engine should be preferred over DB2 Web Query. Therefore, the goal of DB2 Web Query optimization is to push as much of this processing, decision making, and execution as possible down to the database level.

If a request is not optimized, the DB2 Web Query Reporting Server performs the tasks such as joining, sorting, and aggregating. In these cases, the database engine retrieves all the rows that are necessary for the request based on the translation. DB2 Web Query then completes that processing necessary to prepare the data for the report. This can have the following results:

- Higher processing costs for both the database engine and DB2 Web Query
- Potentially higher network communication costs since larger-than-necessary result sets are moved around the network
- Higher personnel costs since time is wasted waiting for inefficient queries to execute and return the requested data

23.2 DB2 CLI adapter performance

In this section, we discuss performance considerations that are specific to the DB2 call-level interface (CLI) adapter.

23.2.1 Report request process flow

As a foundation for understanding DB2 Web Query performance factors, you must first understand the overall process flow when reporting from relational data. It is important to comprehend this process flow so that you can see where the different phases of optimization occur when a report request is made and how you can influence them.

The process of running a DB2 Web Query request consists of two phases:

- The execution phase
- The report production phase

Figure 23-1 illustrates the flow of the request through these two phases.

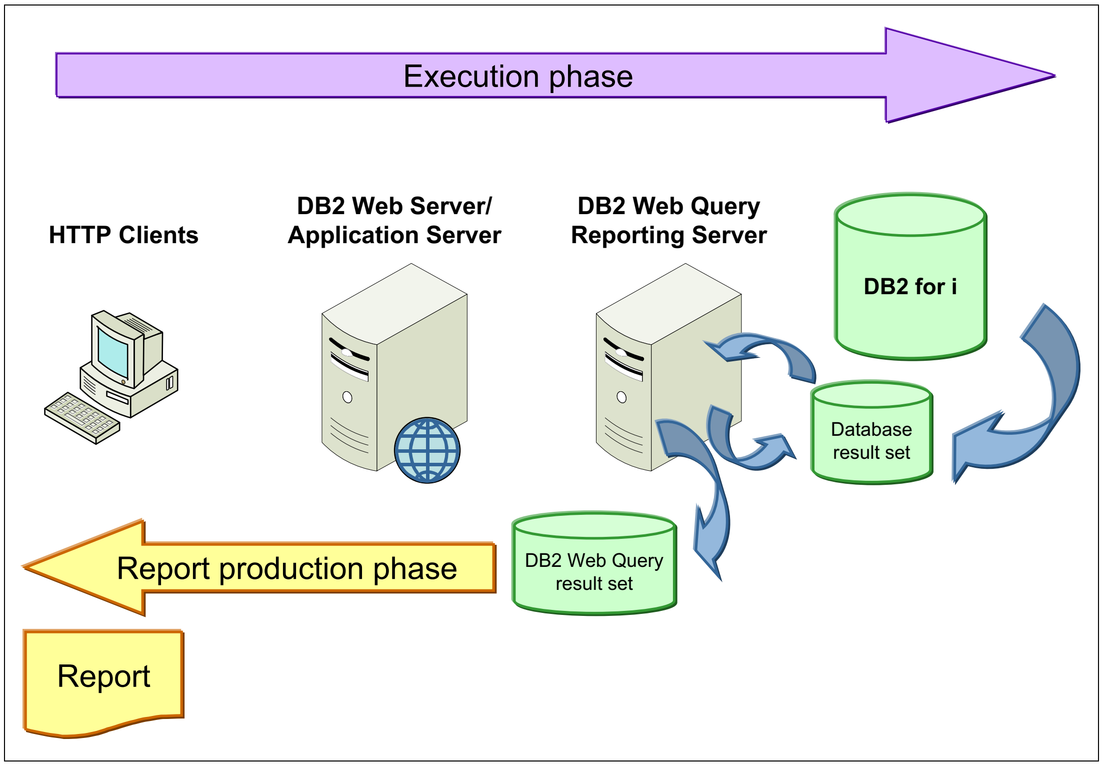


Figure 23-1 Flow of DB2 Web Query report run request

Report execution phase

The following steps occur during the execution phase of a report request:

1. The Web browser client sends the request to the Web server or application server for processing.
2. The Web server or application server routes the request to the DB2 Web Query Reporting Server.

3. The Reporting Server component handles the following tasks during this phase:
   a. Reads and parses the DB2 Web Query metadata
   b. Parses the procedure (checks the syntax of the source code)
   c. Sends a DESCRIBE request to DB2 for i, so that the Reporting Server can create metadata for the result set that will be returned
   d. Passes the procedure to the data adapter for processing
4. The DB2 Web Query data adapter component handles the following tasks during this phase:
   a. Analyzes the DB2 Web Query master file for the specific SQL module and dialect to use, and to retrieve the SQL column names
   b. Analyzes the DB2 Web Query access file for the specific table to access and which connection to use
   c. Translates (optimizes) the DB2 Web Query request to the appropriate SQL statement or statements
   d. Passes the SELECT statement or statements to the DB2 for i database engine
5. The database engine component handles the following tasks during this phase:
   a. Analyzes and optimizes the SQL statement or statements
   b. Chooses the appropriate access path and retrieval method
   c. Retrieves the data
   d. Creates the database result set
6. The report execution phase ends.

Report production phase

At the completion of the report execution phase, the report production phase begins. During the report production phase, the database engine component sends a row and an SQL status code to the Reporting Server.

Note: The database engine is now interfacing with the database result set and not the native relational data on DB2 for i.

The Reporting Server component handles the following tasks during this phase:

1. Reads the row from the database result set and processes any remaining actions on that row (IF/WHERE, DEFINEs, and so on, that were not translated by the database engine)
2. Converts nonstandard data into DB2 Web Query format, making it available to DB2 Web Query
3. Puts the valid row into the DB2 Web Query result set
4. Asks the database engine for the next row (FETCH)
5. Repeats tasks 1–4 (loop) until the end of the database result set (SQL status code +100)
6. Processes the DB2 Web Query result set, applies style sheets, and formats the report output as requested by the instructions in the procedure
7. Displays the report via a browser or native program based on the type of output produced (for example, Adobe or Excel)

23.2.2 Adapter processing and optimization

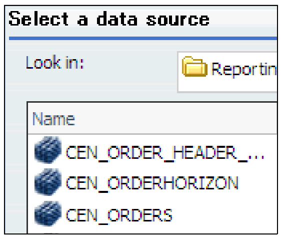
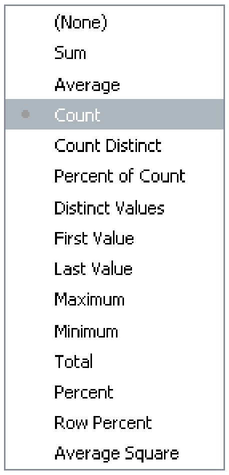
When reporting from relational data sources, the data adapter attempts to translate the source code of a DB2 Web Query procedure into equivalent SQL statements. *Adapter optimization* is the degree to which a DB2 Web Query request is translated to pure SQL statements to be handled by the database engine. Maximizing the adapter optimization should be a major goal of any DB2 Web Query application. If all the commands in a DB2 Web Query report request are translated into the equivalent operation in SQL, then all joining, sorting, or aggregating functions are handled by the database engine. When this occurs, the request is considered (from a DB2 Web Query perspective) to be totally optimized.

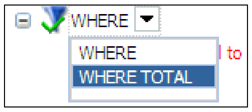
Table 23-1 lists the DB2 Web Query operations and the SQL equivalents.

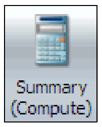
Table 23-1 DB2 Web Query operations and SQL equivalent

Operation	SQL equivalent

Retrieve rows for columns specified.	SELECT
	SELECT DISTINCT(….)

Identify the table from which to report.  Select a data source Look in: Reportin Name CEN_ORDER_HEADER_... CEN_ORDERHORIZON CEN_ORDERS	FROM
Simple aggregation functions.  (None) Sum Average Count Count Distinct Percent of Count Distinct Values First Value Last Value Maximum Minimum Total Percent Row Percent Average Square	SUM, COUNT, AVG, MIN, MAX
Retrieve specific rows. WHERE Country Equal to	WHERE

Retrieve specific aggregated rows.

WHERE
WHERE
WHERE TOTAL
	HAVING

Operation	SQL equivalent
Ordering rows.	ORDER BY
Create logical table structures (joins).	FROM...WHERE...
Define field	FUNCTON(COLUMN)
Compute field 	FUNCTON(SUM( ))

Selection and projection

Two important processes that occur during the SQL translation step of the execution phase are selection and projection. Each is discussed in this section.

- Selection:

  Selection is the process of retrieving the table rows that meet the request criteria. Selection is translated into various predicates of the SQL WHERE clause *except* those expressions listed in Table 23-2.

  Table 23-2 Selection disablers

Selection disabler	Description
Non-translatable DEFINE fields	Certain DEFINE expressions can be translated to SQL as part of aggregation or record selection operations. In some circumstances, they can be used as part of a define-based join. The following examples are of translatable DEFINE expressions: ► For selection, the DEFINE expression must be an arithmetic valued expression, a character string valued expression, or a logical expression. ► For aggregation, the DEFINE expression must be an arithmetic or character string valued (COUNT) expression. ► These expressions are based on what is built (for example, on the right side of the equal sign) in the DEFINE expression, *not* on the format of the field that is being defined. Table 23-3 provides more information about the DEFINE field expressions that can be translated to SQL.
Non-translatable DATE fields	DATE fields with formats other than YMD or YYMD.
Spanned DEFINE fields	DEFINE fields that span more than one segment in a joined structure.

Table 23-3 lists the DEFINE expressions that can be translated to SQL.

Table 23-3 DEFINE expressions that can be translated to SQL

Expression	SQL translatable conditions

Arithmetic expressions
`NEWSAL/D12.2 =`
`((CURR_SAL + OTIME_SAL) x 1.1) - 100;`
	► Real-field operands of numeric data types (I, P, D, F)
	► Numeric constants
	► Arithmetic operators (**, *, /, +, -)
	► Subtraction of one DATE field from another
	► DEFINE-field operands satisfying any of the above

Character string expressions
`FORMAL_NAME/A36 =`
`LAST_NAME | ', ' | FIRSTNAME;`
	► Real-field operands of alphanumeric data types (TEXT field formats are not supported with DEFINE expressions.)
	► String constants
	► String concatenation operators
	► DEFINE-field operands satisfying any of the above

Logical expressions
(Expressions that are evaluated as true, 1, or false, 0.)
 	
`SALES_FLAG/I1 =`
`(DIV_CODE EQ 'SALES') OR (COMMISSION GT 0);`
 	
`QUOTA_CLUB/I1 =`
`(SALES_FLAG) AND (UNITS_SOLD GT 100);`
	► Real-field operands of any DB2 Web Query-supported data type (including DATE fields)
	► Constants with same data type as fields in the predicate
	► Relational operators (EQ, NE, GT, LT, GE, LE)
	► Logical operators (AND, OR, NOT)
	► Arithmetic or character string expression operands (above)
	► DEFINE-field operands satisfying any of the above

For more information about creating DEFINE fields refer to “New fields: Define versus Compute” on page 91.

- Projection:

  Projection is the process of retrieving the table columns that meet the request criteria. Projection is translated to be objects of the SELECT statement, as follows:

  – Columns referenced in PRINT, SUM, or COUNT commands.
  – Columns used as objects in JOIN or DEFINE operations.
  – PRINT * and SEG.fieldname return all columns in the master file only.

  Note: SELECT * is never generated by DB2 Web Query.

It is important to understand the DB2 Web Query selection and projection operations because they are both processes that reduce the volume of data that is returned from the database, which helps to improve performance, efficiency, and report response time.

Optimization hierarchy

In addition to individual operations being optimized, a progressive optimization hierarchy affects the SQL that is being generated. Figure 23-2 shows the order of adapter optimization.

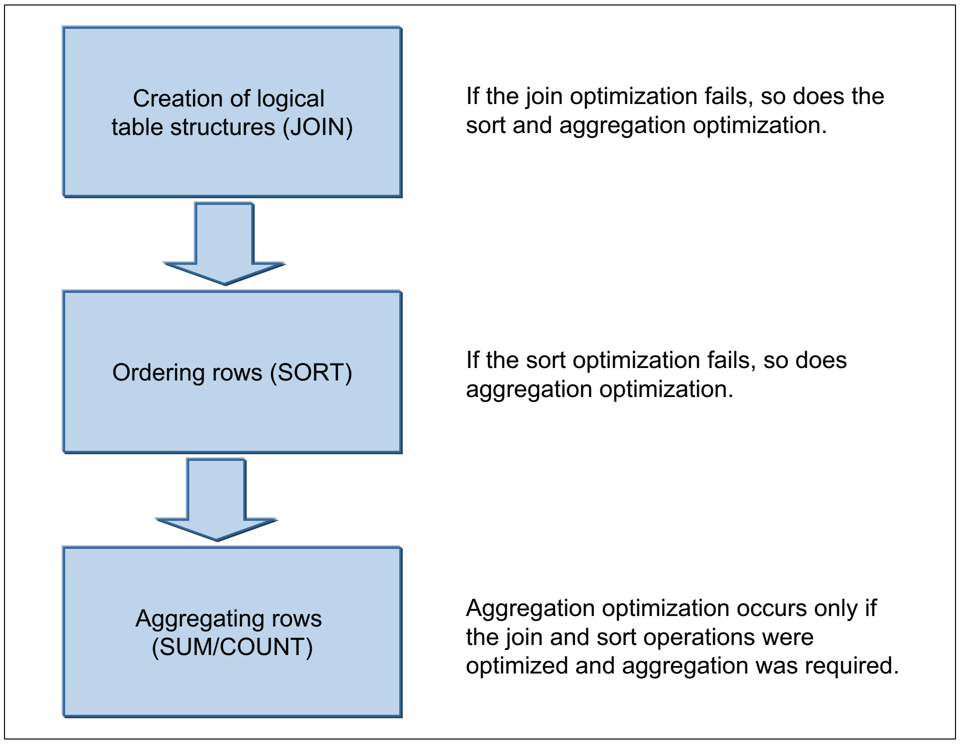

Creation of logical table structures (JOIN)
If the join optimization fails, so does the sort and aggregation optimization.
Ordering rows (SORT)
If the sort optimization fails, so does aggregation optimization.
Aggregating rows (SUM/COUNT)
Aggregation optimization occurs only if the join and sort operations were optimized and aggregation was required.

Figure 23-2 DB2 Web Query adapter optimization order

Translation to SQL: Regardless of the hierarchy in Figure 23-2, the adapter always tries to translate row selection (WHERE) and projection operations to their SQL equivalents.

This hierarchy is mentioned because it is important for you to know what to focus on first. If your report's ordering or aggregating is not being performed by the database engine and you are unable to determine why, it might be because join optimization is failing. Therefore, if you have a poorly performing report that is joining two or more tables, your analysis should begin with join optimization.

Join optimization

A join operation is a complex function that requires special attention in order to achieve good performance. Again, it must be stressed that the goal is to allow the database engine to perform as much of the optimization and processing as possible. Due to their potential complexity, this is especially true for join operations.

If the join is processed by the database engine, the following steps occur:

1. The DB2 Web Query request is optimized and is translated to a single SQL select statement.
2. The database engine retrieves the data from the tables specified in the select statement.
3. The database engine merges the rows using its chosen join implementation.
4. Screening conditions are applied, if any.
5. The columns in the request are applied.
6. Any column function values or expressions are calculated.
7. One result set is produced.

Table 23-4 illustrates the strengths of DB2 Web Query as well with the corresponding strengths of DB2 for i.

Table 23-4 Quick guide to strengths of DB2 Web Query and DB2 for i

DB2 Web Query strengths	DB2 for i strengths
DB2 for i federated joins	Joining on local tables (SELECT...FROM...WHERE)
Complex calculations	Row and column selection (SELECT…WHERE)
Sophisticated formatting	Sorting (SELECT…ORDER BY)
Generate graphs	Aggregation (SELECT…GROUP BY)

Referential integrity and constraint awareness: As of V5R3, the SQL Query Engine also has referential integrity and constraint awareness. This means that the optimizer is capable of using the referential integrity conditions to rewrite the queries to eliminate unnecessary join combinations. This can result in significant performance gains for queries with complex joining and is another compelling reason to try to achieve full DB2 Web Query optimization.

DB2 for i usually handles the join if all the DB2 Web Query commands (where possible) are translated to their SQL equivalent. However, because of the processing hierarchy, sorting and aggregation can still fail.

Conditions that prevent full DB2 Web Query optimization

After a query request is submitted, there are several conditions that cause the adapter to disable the optimization of the join operation. These *join disablers* are shown in Table 23-5. You must eliminate these conditions if you want full DB2 Web Query optimization to occur.

Table 23-5 Join disablers

Join disabler	Description
DEFINE-based JOINs (only if the DEFINE expression is not translatable)	A DB2 Web Query define-based join allows joining a cross-reference file to a host file field that was created by a DEFINE statement. As mentioned previously in "Selection and projection" on page 728, there are cases in which DEFINE fields are not translated to SQL. When this happens, and that DEFINE field is used as a join field, the join is not translated either.
DEFINE fields that span more than one segment in a joined structure	Joining tables based on DEFINE fields prevents the database engine from processing the join.
Multiplicative effect	An aggregation at any level other than the lowest level of the JOINed structure *or* a child table whose foreign key does not totally cover its primary key (causing the parent table rows to be duplicated).
Federated joins	Joining DB2 for i tables on different systems, partitions, or independent auxiliary storage pools (IASPs) is carried out by DB2 for DB2 Web Query and not the database engine.

When you create a join connection (via the join tab) between tables, DB2 Web Query might choose to generate and run multiple SQL statements to compute the correct output for your report. In some cases, DB2 Web Query can reduce these multiple SQL statements to a single SQL statement, thereby improving DB2 Web Query's performance considerably.

In particular, when the joined-in table (called the *target table* to the right of the join panel) is joined via a unique join, DB2 Web Query can process the entire report in one underlying SQL query. A *unique join* is one where given a row from the initial tables (called *host tables* to the left of the join panel) matches at most one row in the joined-in target table (on the right tab). The join must be unique over the entire set of target table joined columns. Note that no one column must be unique in the target table.

Sorting optimization

If any of the conditions listed in Table 23-6 are true, DB2 Web Query performs the sorting.

Table 23-6 Sorting and aggregating disablers

Sort disablers	Description
DB2 Web Query managed join	See Table 23-5 on page 731.
Sort on a non-translatable DEFINE field	Any request that contains a sort on a non-translatable DEFINE field.
Interface-managed join	SQL sorts the answer set by the table's primary key, causing DB2 Web Query to re-sort for the display report.

Aggregation optimization

If any of the conditions listed in Table 23-7 are true, DB2 Web Query performs the aggregation.

Table 23-7 Aggregation disablers

Aggregation disablers	Description
DB2 Web Query managed join or sort	See Table 23-5 on page 731 and Table 23-6.
DB2 Web Query managed row selection	Some WHERE clauses are not passed to the database engine.
Non-direct SQL operators	The request contains direct operators, such as PCT. and TOT., that cannot be translated into SQL.
Aggregation on a non-translatable DEFINE	Aggregating on a DEFINE field that cannot directly translate to SQL.

Other ways to influence optimization

The conditions that we discussed in the previous sections help dictate whether DB2 Web Query or the database engine performs the bulk of the report processing. In addition, there are techniques that you can use to influence this behavior and improve report response time.

Creating SQL views

During DB2 Web Query report development, you might encounter situations in which a report does not fully translate to an SQL statement. If this happens, one technique to help move toward full DB2 Web Query optimization is to create an SQL view with all of the appropriate selection, join, and aggregation syntax specified. Next create the synonym for the view, and in your report definition select that view, rather than the tables from the database description list. This instructs DB2 Web Query to use the view and push the selection, join syntax, and aggregation to the database engine. This particular technique is used to tune a problem report in 23.5, "Performance case study" on page 753.

Some of the many advantages of using SQL views are demonstrated in the following list:

- Provide the ability to specify additional join types that are not supported by the DB2 Web Query product.

  DB2 Web Query allows you to specify inner joins and left outer joins. With SQL views, you can define those and the following join types:
  - Right Outer
  - Left Exception
  - Right Exception
- Allow you to specify CASE statements to handle more complex, conditional logic.
- Provide fullselect support.

  SQL *fullselect* is the term for generating an SQL result set by combining multiple SELECT statements using the UNION, INTERSECT, or EXCEPT operators. This is a feature that helps you solve more complex business requirements.
- Provide Common Table Expression (CTE) and recursive SQL support.

  CTEs can be thought of as temporary views that exist only during the execution of an SQL statement. When defined, the CTE can be referenced multiple times in the same view. This can be used to reduce the complexity of the view, making it easier to comprehend and maintain.

  Among the V5R4 enhancements for DB2 for i was the ability for a CTE to reference itself. This feature provides the mechanism for recursive SQL, which is especially useful when querying data that is hierarchical in nature, such as bill of materials, organizational charts, and airline flight schedules.

  For more information about recursive CTEs, refer to the article "V5R4 SQL Packs a Punch," which you can download from the Web at this website:

  http://www-03.ibm.com/servers/eserver/iseries/db2/pdf/rcte_olap.pdf
- Allow business logic to reside in the database layer, making it available to all users and all SQL interfaces.
- Provide a database abstraction layer.
- Provide security granularity.

When the views are in place, you can restrict users from accessing tables directly and only allow access through the views. Because both selection and projection can be specified on the views, you can control specific rows and columns that users (or groups of users) can access.

Important: Views should not be confused with indexes. They are implemented as non-keyed logical files. This means that views, unlike indexes, have no access paths and thus no access path maintenance. If you or your database administrator (DBA) are concerned about access path maintenance, this is not an issue with views.

Establishing and using foreign key relationships

If you have foreign key relationships explicitly defined in IBM i database, you can take advantage of a DB2 Web Query feature to include with the selected table's synonym, every table related to the selected table by a foreign key. The resulting multi-table synonym describes all of this table's foreign key relationships. This can greatly simplifying the process of creating reports with joins. In fact, in this scenario, you do not even define a join in your report. All columns from the related tables are displayed on the Field Selection tab and you simply select the columns that you want to include in the report. Through the foreign key relationships, DB2 Web Query is able to understand the join syntax that must be created and generates the appropriate SQL statement.

Creating materialized query tables

Materialized query table (MQTs) provide another method of improving performance of your queries. An MQT is a table that contains the results of a previously run query, along with the query's definition. It provides a mechanism for improving the response time of complex SQL queries. While it can be thought of as a summary table, what sets an MQT apart from a regular summary table is the fact that the SQE optimizer is aware of it and its relationship to the query and base tables that were specified when it was created and populated. This means that the optimizer considers using the MQT in the access plan of subsequent similar queries if it determines that it is appropriate to do so. Because the MQT is already created, populated, joined, aggregated, and sorted, this can result in significant performance improvements for complex queries.

It is important to understand that an MQT is a table (a physical file with an object type of *FILE) that resides in a library (schema) in the System i environment. Because of this, it can be accessed directly, just like any other table on the system. While it is usually the optimizer's job to select and implement an MQT in the access plan of a query request, DB2 Web Query has the ability to create metadata against an MQT. This means that you can create DB2 Web Query reports that access an MQT directly, and not be forced to rely on the optimizer to select the MQT.

To obtain this behavior, you must first create the metadata for the MQT:

1. From the DB2 Web Query BI portal, right-click your reports folder and select **Metadata** → **New**.
2. In the browser window, from the left navigation pane "Steps", select **Synonym Or Samples,** and select the **DB2 CLI** adapter. Select **Create Synonym**.
3. In the Select Synonym Candidates for DB2 (*LOCAL) pane (Figure 23-3), specify the collection and select the **MQTs** check box. Click **Next**.

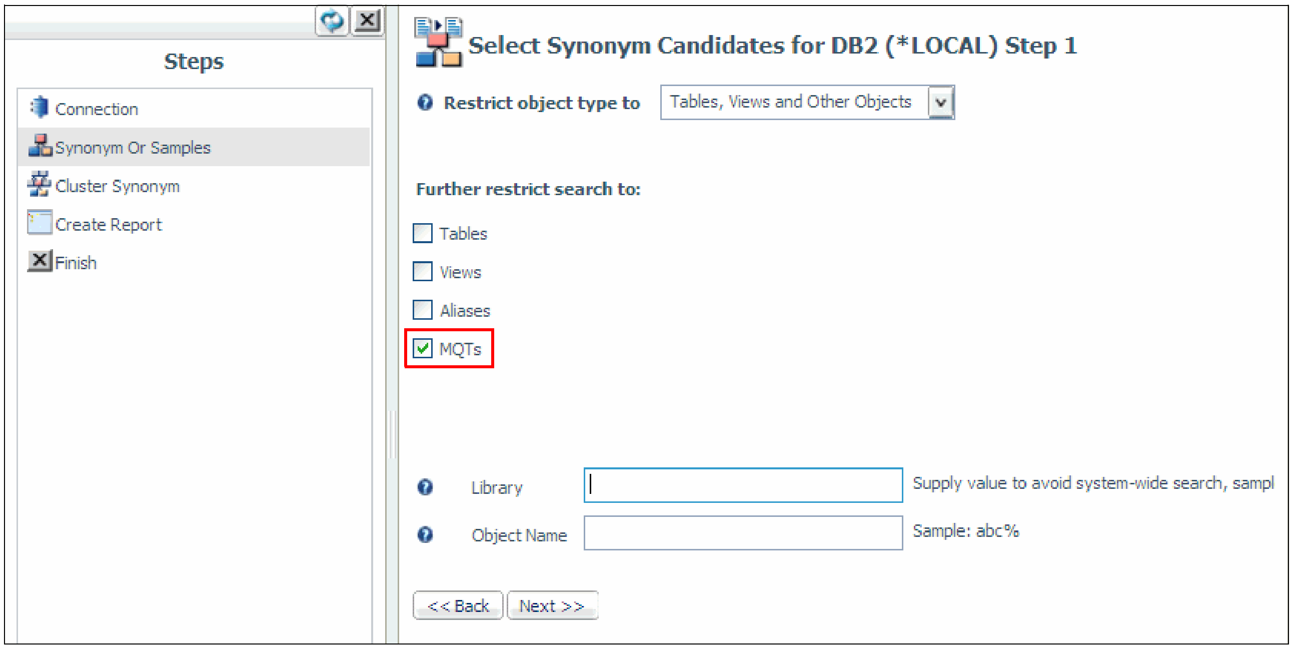


Figure 23-3   Creating a synonym for MQT: specifying the collection

4. In the Create Synonym for DB2 CLI pane, specify a prefix and suffix, select the MQTs that are listed, and click **Create synonym**.

After the synonyms are created for the MQTs, you can create reports against them as you would for any other table object on the system.

While their potential performance efficiencies are appealing for DBAs, query developers, and report users, MQTs have several attributes and limitations that must be understood prior to implementation. Probably most important, you must know that MQTs are not automatically maintained. This means that as the base tables used to populate the MQT change, the data in the MQT does not also change. If the data in your reports must be up to date, the MQT must first be manually refreshed. Therefore, before implementing MQTs, you must be willing to accept some level of data latency. This is can be acceptable for queries that report on data that is historical in nature. However, if your report requires data that is dynamic and up-to-the-minute, you might come to the conclusion that an MQT is not the right fit.

For more information about MQTs, see the white paper *Creating and using materialized query tables (MQT) in IBM DB2 for i*, which you can find on the Web at this website:

http://www-304.ibm.com/jct09002c/partnerworld/wps/servlet/ContentHandler/SROY-6UZ5E6bv

23.2.3 Remote database access considerations (including cross-system joining)

One feature of the DB2 Web Query base product is the ability to access data on remote DB2 for i systems or partitions. This feature, which is available only for the DB2 CLI adapter, gives you the option to pull data for your reports in one of the following ways:

- Local: All the data resides on the local system.
- Remote: All the data resides on one remote system or partition.
- Cross system: Data is spread across the local system or one or more remote systems or partitions.

If full DB2 Web Query optimization can be performed for Local and Remote database access, and no other external factors are negatively influencing query performance, you can expect your reports to perform reasonably well. Again, this is because DB2 Web Query can generate one SQL statement and allow the DB2 for i database engine to process the request.

However, you must be aware of the performance implications if the data is spread across two or more System i machines or partitions. In this scenario, DB2 Web Query generates a separate SQL statement for each connection and submits the statement to each connection. The answer set from each of these multiple sources is then joined together by the Reporting Server. Depending on how the joins are specified, this can be a lengthy and time-consuming process. Consider the following scenario.

A fact table (ORDERS) has 32,283 rows and is joined to a dimension table (STORES) that has 116 rows. If both of the tables are on the local (or even remote) system, DB2 Web Query generates a single SQL statement and submits it to DB2 for i. Example 23-1 shows how the statement looks.

Example 23-1 Single SQL statement

```
SELECT T2."Country",T2."State", SUM(T1."LineTotal")
FROM CENTURY/ORDERS T1,CENTURY/STORES T2
WHERE (T2."StoreCode" = T1."StoreCode")
GROUP BY T2."Country",T2."State"
ORDER BY T2."Country",T2."State"
FOR FETCH ONLY
```

Since one statement is generated and the database engine handles all the joining, grouping, and ordering, you can expect this report to run quite efficiently.

However, if the dimension table STORES resides on another system and Info Assist is used to create a nearly identical report (the only difference being a cross-system join specification to STORES), DB2 Web Query now creates two SQL statements, one for each system.

- Local system:

  Example 23-2 Local system SQL statement

  ```
  SELECT T1."StoreCode",T1."LineTotal"
  FROM CENTURY/ORDERS T1
  FOR FETCH ONLY
  ```

- Remote system:

  Example 23-3 Remote system SQL statement

  ```
  SELECT T5."Country",T5."State"
  FROM "CENTURY"/"STORES" T5
  WHERE (T5."StoreCode" = ?)
  FOR FETCH ONLY
  ```

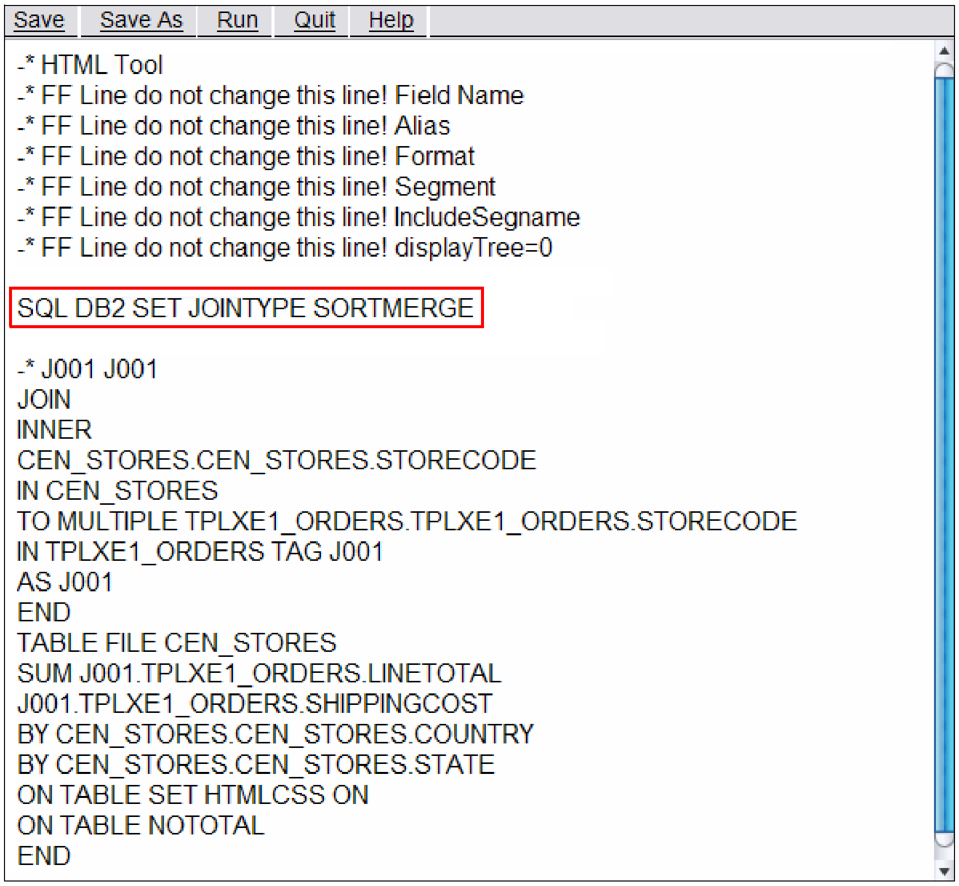
Inefficiencies occur because the remote SQL statement is submitted multiple times on the remote system. The local system fetches each row from the (local) ORDERS table result set and generates the appropriate local selection to return the matching row from the remote STORES table. With this implementation, the report takes significantly longer before the results are displayed to the browser.

If the report is modified to specify STORES (with only 116 rows) as the base table, the join fan-out is greatly reduced. This results in a report that runs significantly faster, but is still not nearly as fast as when all tables are on same system. Therefore, we generally recommend that you avoid creating reports with cross-system joins when the tables have a substantial number of rows. However, if such an implementation is required, consider the following suggestions to enhance the report's performance:

- Eliminate the report's requirement of multiple connections by making local copies of the remote tables. After all tables are on one system, join optimization can be enabled and DB2 Web Query attempts to generate a single SQL statement.
- Determine which of the tables has the smallest number of rows (after applying local selection) and make that the base table of the report. This reduces the fan-out effect of the join, thereby reducing the number of rows that must be retrieved from the other (larger) tables.
- Edit the report source and add the following line:

  ```
  SQL DB2 SET JOINTYPE SORTMERGE
  ```

  Figure 23-4 shows an example of this setting specified in the report source. This setting impacts the number of FOR FETCH lines when one of the tables is remote. In effect, it turns all of the tables into local internal tables and sort merges them.



```
-* HTML Tool
-* FF Line do not change this line! Field Name
-* FF Line do not change this line! Alias
-* FF Line do not change this line! Format
-* FF Line do not change this line! Segment
-* FF Line do not change this line! IncludeSegname
-* FF Line do not change this line! displayTree=0

SQL DB2 SET JOINTYPE SORTMERGE

-* J001 J001
JOIN
INNER
CEN_STORES.CEN_STORES.STORECODE
IN CEN_STORES
TO MULTIPLE TPLXE1_ORDERS.TPLXE1_ORDERS.STORECODE
IN TPLXE1_ORDERS TAG J001
AS J001
END
TABLE FILE CEN_STORES
SUM J001.TPLXE1_ORDERS.LINETOTAL
J001.TPLXE1_ORDERS.SHIPPINGCOST
BY CEN_STORES.CEN_STORES.COUNTRY
BY CEN_STORES.CEN_STORES.STATE
ON TABLE SET HTMLCSS ON
ON TABLE NOTOTAL
END
```

Figure 23-4 SORTMERGE setting

23.3 Query/400 adapter performance

Another feature of DB2 Web Query is the ability to run most existing Query/400 queries. This is done by using the Query/400 adapter that is provided with the base offering of 5733-QU2. This adapter functions by running the Query/400 query during both the synonym creation process and report execution. This means that if you have an existing Query/400 object that does not perform well and takes several minutes to run to completion, it requires just as much time (and perhaps more) to create the DB2 Web Query synonym. After the synonym is created and you run the new DB2 Web Query report, you can again expect it to require at least the same amount of time to run.

Because the adapter is running the existing query, you are limited in the tuning that can be done to make each of the DB2 Web Query processes perform better. In most cases, the sole tuning knob ensures that the appropriate indexes are in place for the optimizer to use when creating the access plan for the query. Refer to 23.4.3, "Indexes" on page 747, to learn about approaches for implementing an indexing strategy.

As we mentioned earlier in this chapter, Classic Query Engine (CQE) processes non-SQL database requests such as those from Query/400 queries. CQE provides index advisory information for query selection. Missing from CQE is the ability to create index advisories for joining, grouping, and ordering. Analysis and tuning, which are more manual in nature, are required for these types of requests. If your Query/400 reports are not performing to your satisfaction, make sure that the indexes exist over all join fields, order by fields, and fields that are aggregated.

23.4 DB2 for i optimization

By now you should be convinced that getting DB2 Web Query to hand off as much of the processing as possible to the database engine is a key factor in achieving optimal report performance. There are also factors within the IBM i database itself that can affect efficiency.

These types of factors are usually the responsibility of the DBA and not the users. However, a little knowledge of these factors and how they can affect efficiency enables a user to communicate effectively with the DBA.

When the users have done everything that they possibly can in the efficiency spectrum to ensure that the report definitions are as good as they can be, the following DB2 for i factors can still influence the overall efficiency of the request:

- Database design
- Query Engine used
- Indexes
- Available hardware
- Number of concurrent users

23.4.1 Database design

Good performance starts with a good database design. If your database structure is flat in nature and not normalized, the files are likely to have redundant data that is difficult to navigate, retrieve, and maintain. The complexities of retrieval and maintenance are often handled by a high-level language (HLL) program written in RPG or COBOL. This approach works when such programs are the only interface to the data and can hide the complexity from users. However, problems and confusion are likely to result when you expose your database and allow other interfaces, such as DB2 Web Query, to access and report against the data. In the following list, we highlight the shortcomings of a poorly designed database:

- Complexity:

  The application programs contain most, if not all, business rules and data relationships. Therefore, the data is difficult to query because it is hard to understand rules and relationships. Because of the complexity, programmers (and not users) are forced to develop the reports. Often the reports themselves are HLL programs that can be complex and difficult to maintain. Programmers can find themselves constantly creating new copies of reports to satisfy a seemingly endless list ad hoc reporting requirements by the user.

- Lacks flexibility:

  The database is difficult to maintain, adapt, and expand. Repeating groups must be squeezed into a single record, which places limits on the number of repeating groups that can fit into a single record format.

- Less optimal access methods:

  Performance of these reports can suffer because of the database complexity. Programmatic selection, joining, ordering, and grouping are often implemented with a row-at-a-time coding techniques. Record blocking and set-at-a-time processing are not used and, therefore, the report performance is not maximized.

If this describes your current database design, you might want to consider investing in a robust data modeling tool that can help you implement a database design that is both functional and efficient.

23.4.2 Query Engine used

As described earlier in this publication, DB2 for i employs two database engines. Both CQE and SQE are used to process database access requests. Although SQE is the engine designed for SQL access, there are still inhibitors that can prevent SQE processing for SQL requests. The SQE inhibitors for each supported version of the operating system are listed in Table 23-8.

Table 23-8 SQE inhibitors

6.1 SQE inhibitors	7.1 SQE inhibitors
▸ ICU 2.6.1 Sort Sequences ▸ Non-SQL Interfaces (OPNQRYF, Query/400, QQQQRY API) ▸ Logical File reference on FROM Clause ▸ Select/Omit Logical Files defined on tables	▸ Non-SQL Interfaces (OPNQRYF, Query/400, QQQQRY API)

Important: While as of 6.1 the SQE optimizer is still unable to utilize Select/Omit logical files when building query plans, IBM did change the default value for the Ignore_Derived_Index QAQQINI parameter to enable more SQE usage. This QAQQINI parameter was first added back in V5R3 to allow the SQL Query Engine to be used in environments where SQL statements are referencing DB2 objects created with DDS. Prior to 6.1, the default value for this parameter was *NO, causing the SQE optimizer to reroute execution of any SQL request to the Classic Query Engine (CQE) anytime that a derived logical file was encountered during the query optimization process. The default value in 6.1 for the Ignore_Derived_Index QAQQINI parameter has been changed to *YES. This value allows the SQE query optimizer to ignore any keyed logical files that contain select/omit criteria during optimization and process the query instead of rerouting execution to CQE.

Your goal as a database administrator or DB2 Web Query administrator/developer should be to ensure that each DB2 Web Query request is processed by SQE. Why is this so important? Because SQE is the IBM strategic optimizer, and as such, it will be the one on which the IBM development team focuses. Enhancements that add new features and boost performance will only be applied to SQE, not CQE. Consequently, when a query is processed by SQE you can expect vastly superior performance over the same query handled by CQE. Some of the reasons for this include:

- Enhanced optimization techniques—a smarter, faster, and more efficient optimizer
- Self-learning query optimization
- Ability to self-adjust during query execution
- Better/faster database primitives to access the data

In addition, SQE provides many more features than CQE. Among them are:

- Materialized Query Tables (MQTs)
- Maintained Temporary Indexes (MTIs)
- SQE plan cache
- Ability to cache results

Achieving SQE processing for every DB2 Web Query request is quite simple: avoid SQE Inhibitors. The following list provides the major issues to look out for in a proactive mode:

- Avoid creating metadata for DDS logical files:
  - Only create metadata over tables/physical files, SQL views, or MQTs.
  - If you need the filtering or joining provided by a DDS Logical File, create an SQL View that performs the equivalent filtering/joining.

- Watch out for Select-Omit logical files against physical files:

  Again, this is not as important in 6.1because the default behavior is for SQE to ignore Select-Omit logical files. But if you are at 5.4 of the IBM i operating system and you need to create metadata over a physical file that has Select-Omit logical files against it, you will can instruct the optimizer to ignore this derived logical file by using the IGNORE_DERIVED_INDEX parameter in the QAQQINI query options file. This process is documented in the tech tip *Maximize SQL Query Engine (SQE) Usage of Your DB2 Web Query Reports,* which can be access from the following URL:

  `http://www.mcpressonline.com/database/db2/maximize-sql-query-engine-sqe-usage-of-your-db2-web-query-reports.html`

- Use the DB2 CLI adapter whenever possible.
  - Keep in mind that both the Heritage File and Query/400 adapters use CQE.
  - Recreate Query/400 reports as new reports that use synonyms based on the DB2 CLI adapter.
  - There is no way to avoid it for multi-format files (must use Heritage File adapter).
  - Use SQL Aliases to access multiple members.

Determining which engine is used

You can also work in a reactive mode to determine which engine is handling your query requests. This can be done by collecting a database monitor and analyzing the results. Two of the most common ways to do this are as follows:

- Run the statement in Run SQL Scripts window (System i Navigator) and use Visual Explain to analyze the results.
- Start a database monitor collection, run the query, and use the System i Navigator dashboard to analyze results.

Using Run SQL Scripts window and Visual Explain

With this method, you capture the SQL statement that DB2 Web Query is producing for a report and copy and paste the statement into the Run SQL Scripts window of System i Navigator. Visual Explain is then launched against the statement to produce a graphical representation of the optimizer's chosen access plan for the query, as well as a wealth of information about the plan. Part of the information provided in this process is which query engine is used.

To implement this method, take the following steps:

1. Right-mouse the report and select **Run w/SQL Trace**.

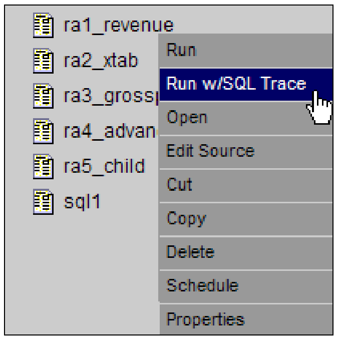
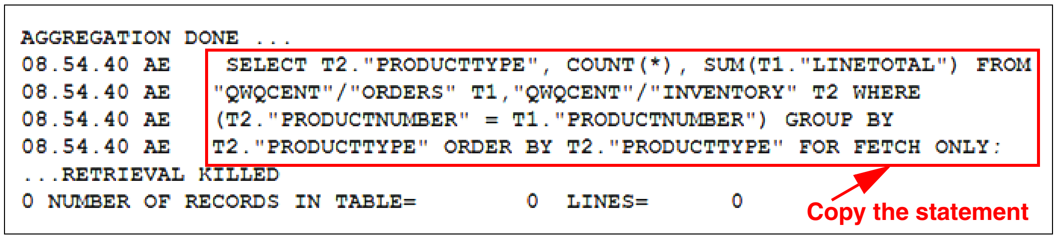


Figure 23-5 Run w/SQL Trace

Rather than running the report and displaying the results, this option displays the generated SQL statement.

2. As shown in Figure 23-6, copy this statement to your clipboard.

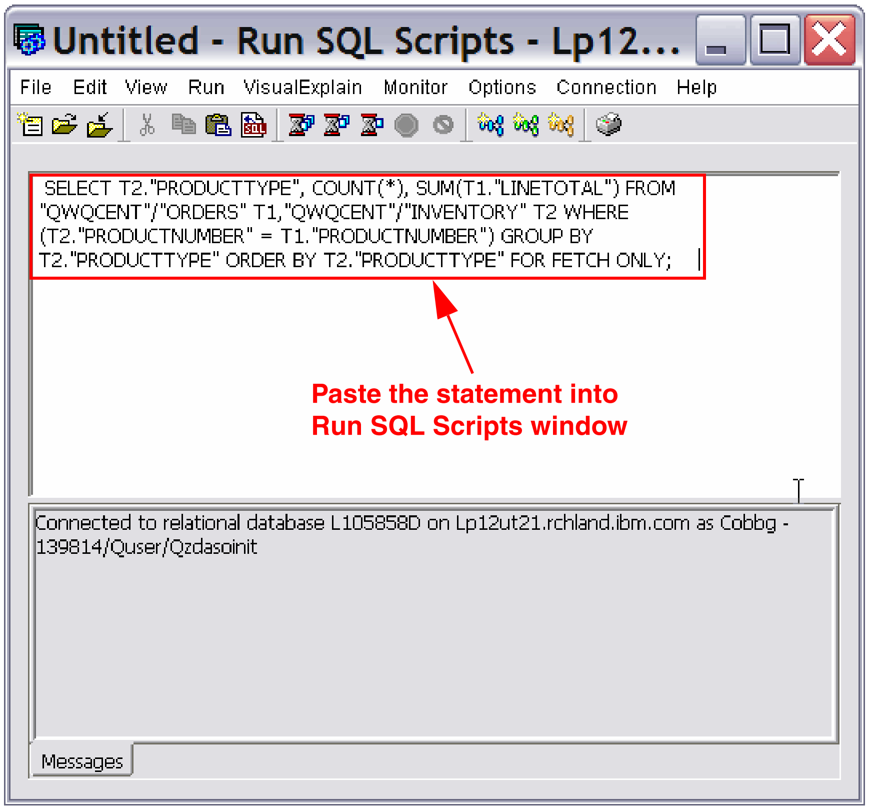


```
AGGREGATION DONE ...
08.54.40 AE    SELECT T2."PRODUCTTYPE", COUNT(*), SUM(T1."LINETOTAL") FROM
08.54.40 AE   "QWQCENT"/"ORDERS" T1,"QWQCENT"/"INVENTORY" T2 WHERE
08.54.40 AE   (T2."PRODUCTNUMBER" = T1."PRODUCTNUMBER") GROUP BY
08.54.40 AE   T2."PRODUCTTYPE" ORDER BY T2."PRODUCTTYPE" FOR FETCH ONLY;
...RETRIEVAL KILLED
0 NUMBER OF RECORDS IN TABLE=        0  LINES=      0
```

Figure 23-6 Copy the SQL statement

3. In System i Navigator, open a connection to your system and launch a Run SQL Scripts window.

4. Paste the statement from your clipboard into the Run SQL Scripts window. This is shown in Figure 23-7.



Untitled - Run SQL Scripts - Lp12...
File Edit View Run VisualExplain Monitor Options Connection Help
Paste the statement into
Run SQL Scripts window
Messages

```
SELECT T2."PRODUCTTYPE", COUNT(*), SUM(T1."LINETOTAL") FROM
"QWQCENT"/"ORDERS" T1,"QWQCENT"/"INVENTORY" T2 WHERE
(T2."PRODUCTNUMBER" = T1."PRODUCTNUMBER") GROUP BY
T2."PRODUCTTYPE" ORDER BY T2."PRODUCTTYPE" FOR FETCH ONLY;
```

Connected to relational database L105858D on Lp12ut21.rchland.ibm.com as Cobbg - 139814/Quser/Qzdasoinit

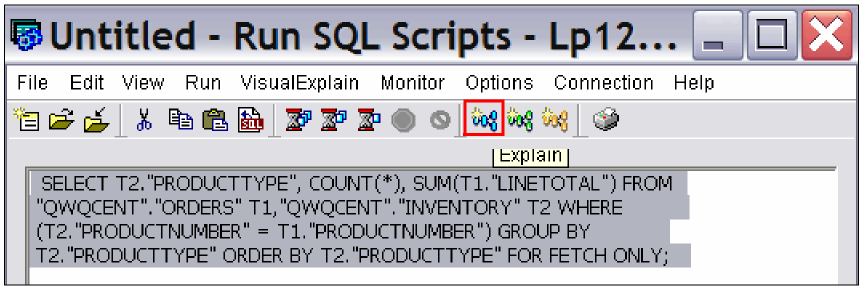
Figure 23-7 Paste the SQL statement

Tip: Depending on your configured naming convention (*SQL or *SYSTEM in the format tab of the JDBC settings for your System i Navigator connection), you may have to edit the statement after you paste it into the window. These are the naming conventions:

- The *SQL naming convention uses a period (.) between the library and object names.
- The *SYS naming convention uses a forward slash (/) between the library and object names.

This means that you may have to the replace the / character with a period (.) for all qualified object references.

5. As shown in Figure 23-8, swipe/highlight the statement and click the Explain icon in the tool bar.

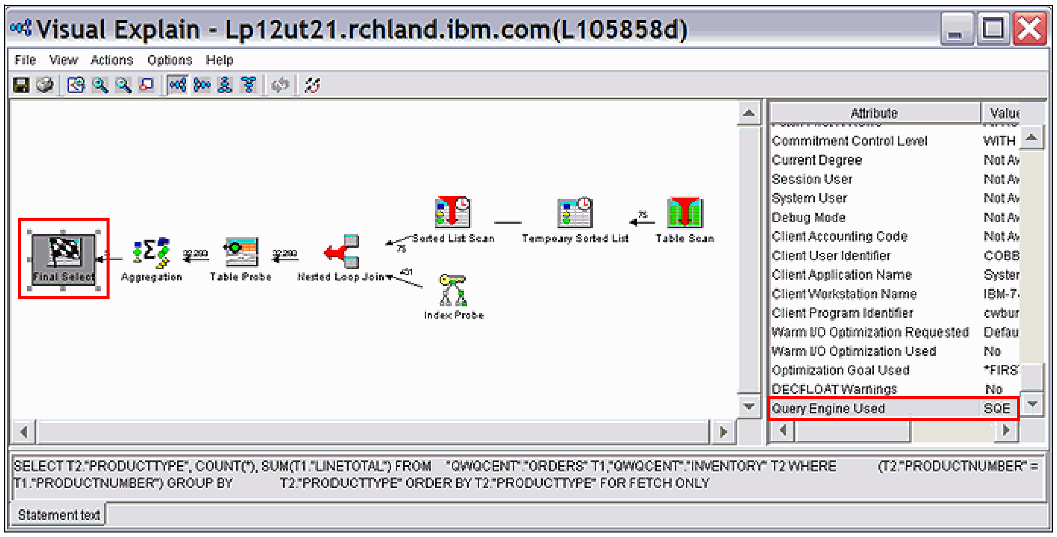


```
SELECT T2."PRODUCTTYPE", COUNT(*), SUM(T1."LINETOTAL") FROM
"QWQCENT"."ORDERS" T1,"QWQCENT"."INVENTORY" T2 WHERE
(T2."PRODUCTNUMBER" = T1."PRODUCTNUMBER") GROUP BY
T2."PRODUCTTYPE" ORDER BY T2."PRODUCTTYPE" FOR FETCH ONLY;
```

Figure 23-8 Explain the SQL statement

This action launches the Visual Explain utility. Visual Explain provides information about the optimizer’s chosen access plan. It also renders a graphical representation of the query access plan.

6. Select the Final Select icon on the plan and scroll down to the bottom of the Attribute pane on the right. This is shown in Figure 23-9.



Attribute	Value
Commitment Control Level	WITH
Current Degree	Not Av
Session User	Not Av
System User	Not Av
Debug Mode	Not Av
Client Accounting Code	Not Av
Client User Identifier	COBB
Client Application Name	Syster
Client Workstation Name	IBM-7-
Client Program Identifier	cwbur
Warm I/O Optimization Requested	Defau
Warm I/O Optimization Used	No
Optimization Goal Used	*FIRS
DECFLOAT Warnings	No
Query Engine Used	SQE

```
SELECT T2."PRODUCTTYPE", COUNT(*), SUM(T1."LINETOTAL") FROM    "QWQCENT"."ORDERS" T1,"QWQCENT"."INVENTORY" T2 WHERE        (T2."PRODUCTNUMBER" =
T1."PRODUCTNUMBER") GROUP BY          T2."PRODUCTTYPE" ORDER BY T2."PRODUCTTYPE" FOR FETCH ONLY
```

Figure 23-9 Visual Explain example

The Query Engine Used attribute appears at the bottom. The value of this attribute provides the information for which you are looking. In this case, SQE was used.

Using database monitor collection

Another technique is to run the report in DB2 Web Query after starting a an SQL monitor. You can then use the dashboard provided with the On Demand Performance Center tool to quickly determine which query engine is used.

The following steps describe this process:

1. Open a System i Navigator connection to your system and expand **Databases** → ***<name_of_your_database>***.
2. Right-click **SQL Performance Monitors** and select **New** → **SQL Performance Monitor**, as shown in Figure 23-10.

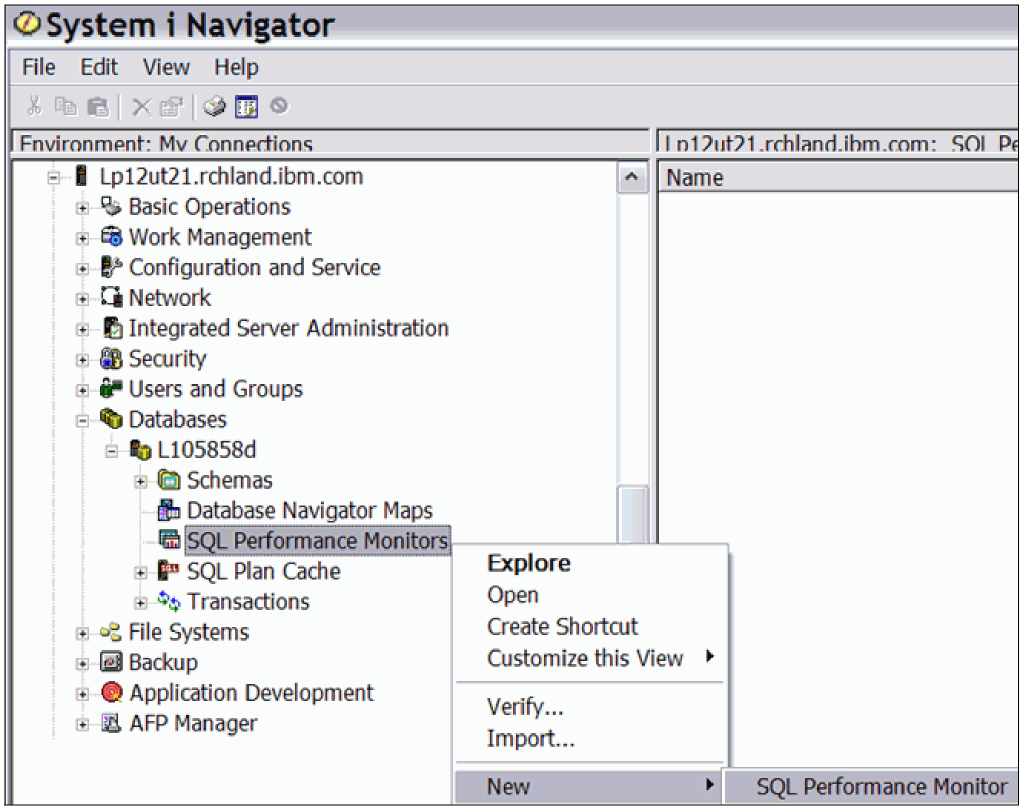
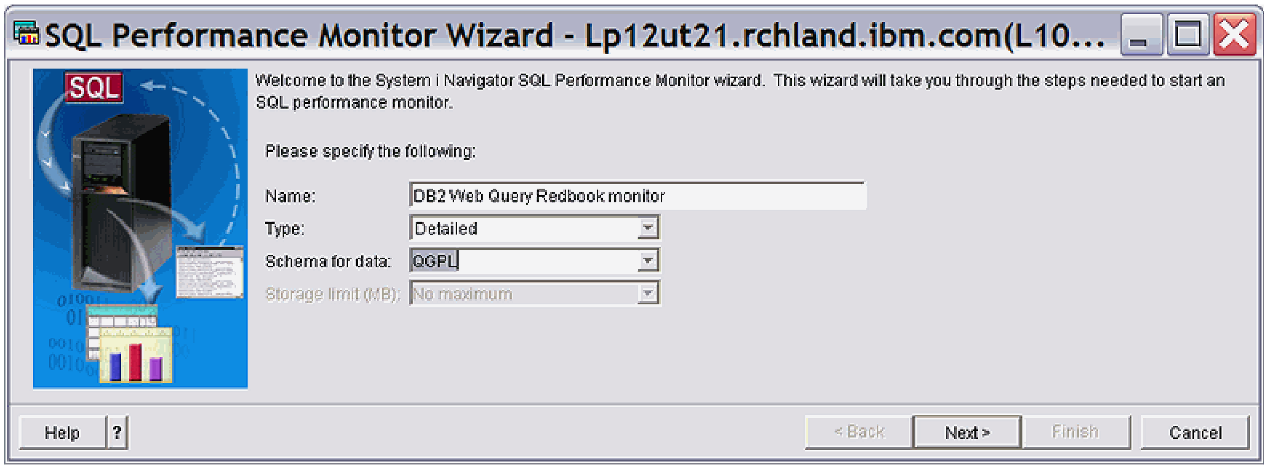


Figure 23-10 Start new SQL Performance Monitor

3. On the SQL Performance Monitor Wizard dialog window, specify the following items:
   - Name: DB2 Web Query Redbook monitor
   - Type: Detailed
   - Schema for data: QGPL

   Click **Next**. An example screen is provided in Figure 23-11.


SQL Performance Monitor Wizard - Lp12ut21.rchland.ibm.com(L10...

Welcome to the System i Navigator SQL Performance Monitor wizard. This wizard will take you through the steps needed to start an SQL performance monitor.

Please specify the following:

Name: DB2 Web Query Redbook monitor

Type: Detailed

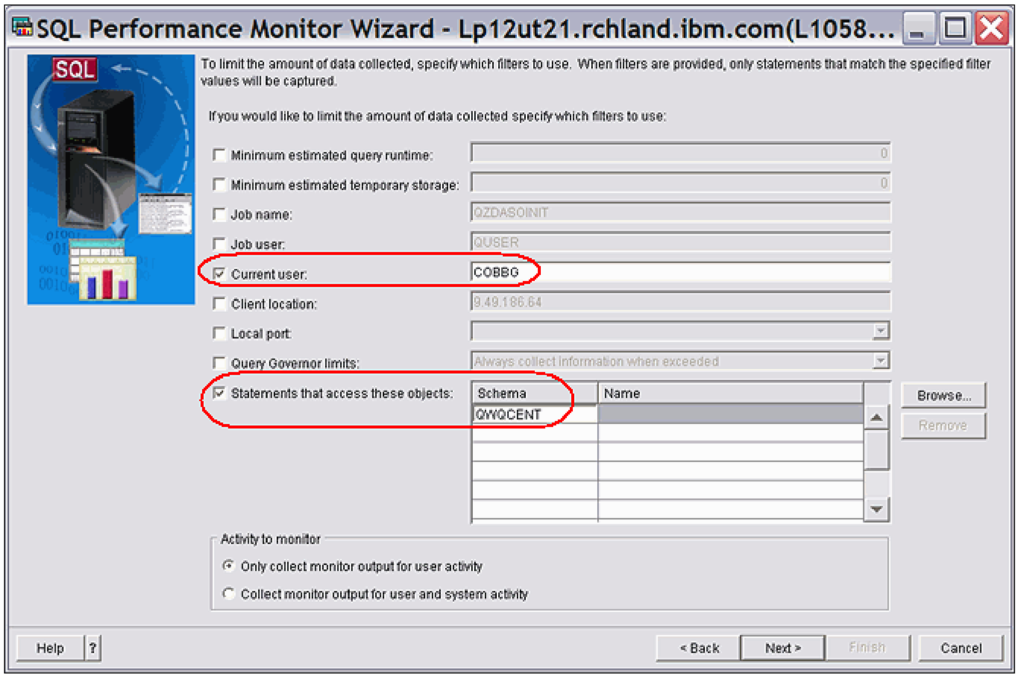
Schema for data: QGPL

Storage limit (MB): No maximum

Figure 23-11 SQL Performance Monitor Wizard window 1

The second dialog window for SQL Performance Monitor Wizard is presented. This window allows you to specify pre-filtering criteria for the monitor collection. Using pre-filtering is strongly recommended because it can greatly reduce the amount of monitor data collected. This will help keep your analysis simple. In this example, you pre-filter on the user profile running the request and the schema (library) name of the object being queried. This should keep the collection small (ideally to a single SQL statement).

4. As shown in Figure 23-12, specify the following prefilter criteria and click **Next**:
   - Check the box for **Current user** and specify your user profile name.
   - Check the box for the setting **Statements that access these objects** and under **Schema**, enter `QWQCENT`.


SQL Performance Monitor Wizard - Lp12ut21.rchland.ibm.com(L1058...

To limit the amount of data collected, specify which filters to use. When filters are provided, only statements that match the specified filter values will be captured.

If you would like to limit the amount of data collected specify which filters to use:

- ☐ Minimum estimated query runtime: 0
- ☐ Minimum estimated temporary storage: 0
- ☐ Job name: QZDASOINIT
- ☐ Job user: QUSER
- ☑ Current user: COBBG
- ☐ Client location: 9.49.186.64
- ☐ Local port:
- ☐ Query Governor limits: Always collect information when exceeded
- ☑ Statements that access these objects:

Schema	Name
QWQCENT	

Activity to monitor

- [✓ Only collect monitor output for user activity]
- Collect monitor output for user and system activity

Figure 23-12 SQL Performance Monitor Wizard window 2

5. On the third dialog window for SQL Performance Monitor Wizard, select the radio button for **All jobs** and click **Next**. An example is displayed in Figure 23-13.

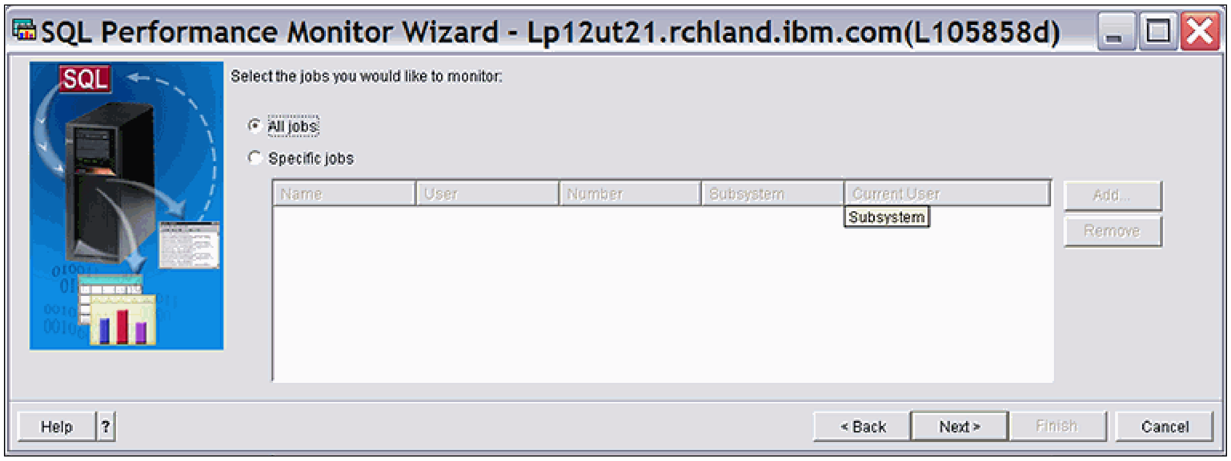
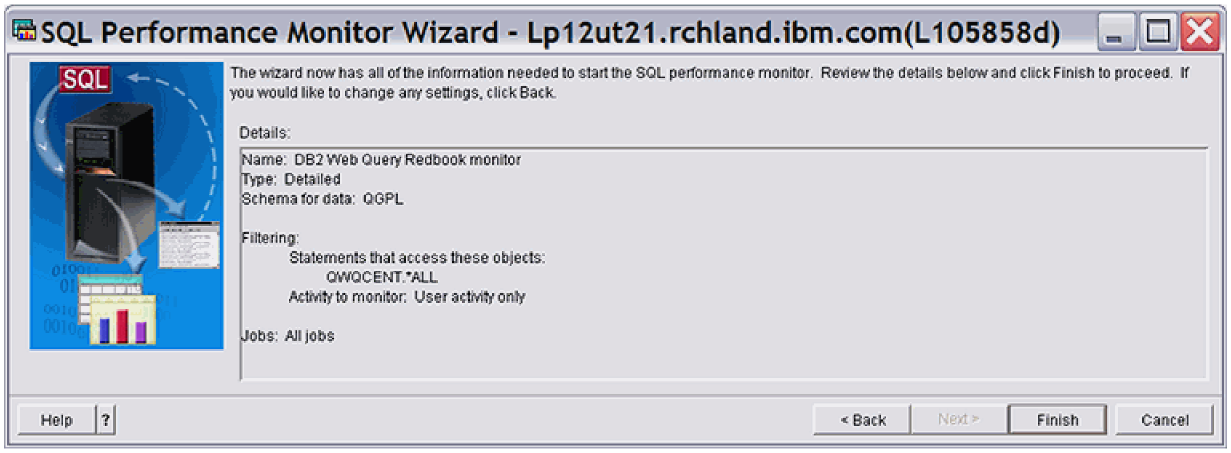

SQL Performance Monitor Wizard - Lp12ut21.rchland.ibm.com(L105858d)
SQL
Select the jobs you would like to monitor:
All jobs
Specific jobs
Name
User
Number
Subsystem
Current User
Subsystem
Add...
Remove
Help
?
< Back
Next >
Finish
Cancel

Figure 23-13 SQL Performance Monitor Wizard window 3

On the final dialog window for SQL Performance Monitor Wizard, click **Finish**.


SQL Performance Monitor Wizard - Lp12ut21.rchland.ibm.com(L105858d)
SQL
Help
?
< Back
Next >
Finish
Cancel

The wizard now has all of the information needed to start the SQL performance monitor. Review the details below and click Finish to proceed. If you would like to change any settings, click Back.

Details:

```
Name: DB2 Web Query Redbook monitor
Type: Detailed
Schema for data: QGPL

Filtering:
        Statements that access these objects:
              QWQCENT.*ALL
        Activity to monitor: User activity only

Jobs: All jobs
```

Figure 23-14 SQL Performance Monitor Wizard window 4

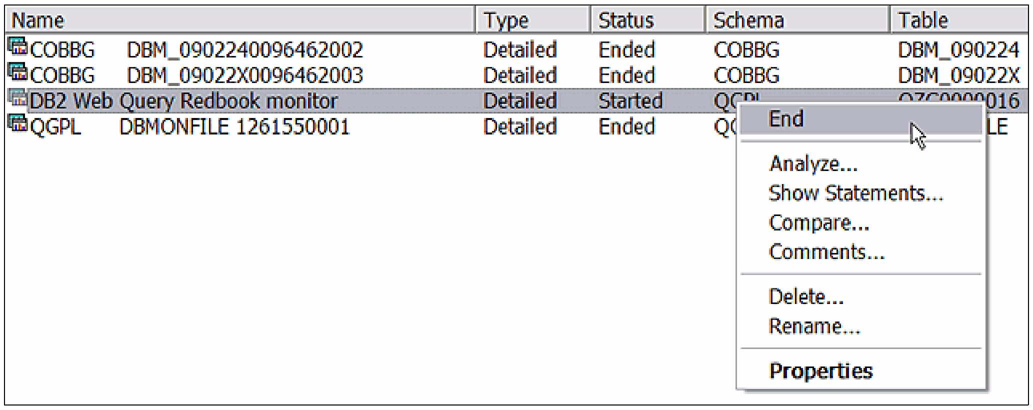
6. Return to your DB2 Web Query browser session and run the report.

   At this point the monitor data has been collected for the report run. The monitor can now be ended.

7. Return to the System i Navigator session. Under the connection, select **Databases** → **<name_of_your_database** → **SQL Performance Monitors**.

   A list of available SQL Performance monitors is displayed.

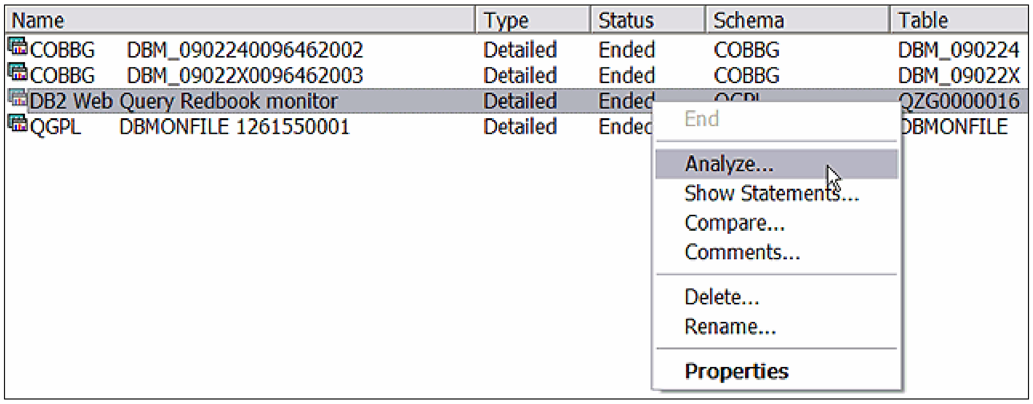
8. Find the monitor that you just started and select **End** from the right-click menu (Figure 23-15).



Name	Type	Status	Schema	Table
COBBG DBM_0902240096462002	Detailed	Ended	COBBG	DBM_090224
COBBG DBM_09022X0096462003	Detailed	Ended	COBBG	DBM_09022X
DB2 Web Query Redbook monitor	Detailed	Started	QGPL	QZG0000016
QGPL DBMONFILE 1261550001	Detailed	Ended	Q	LE

Figure 23-15 End SQL Performance Monitor

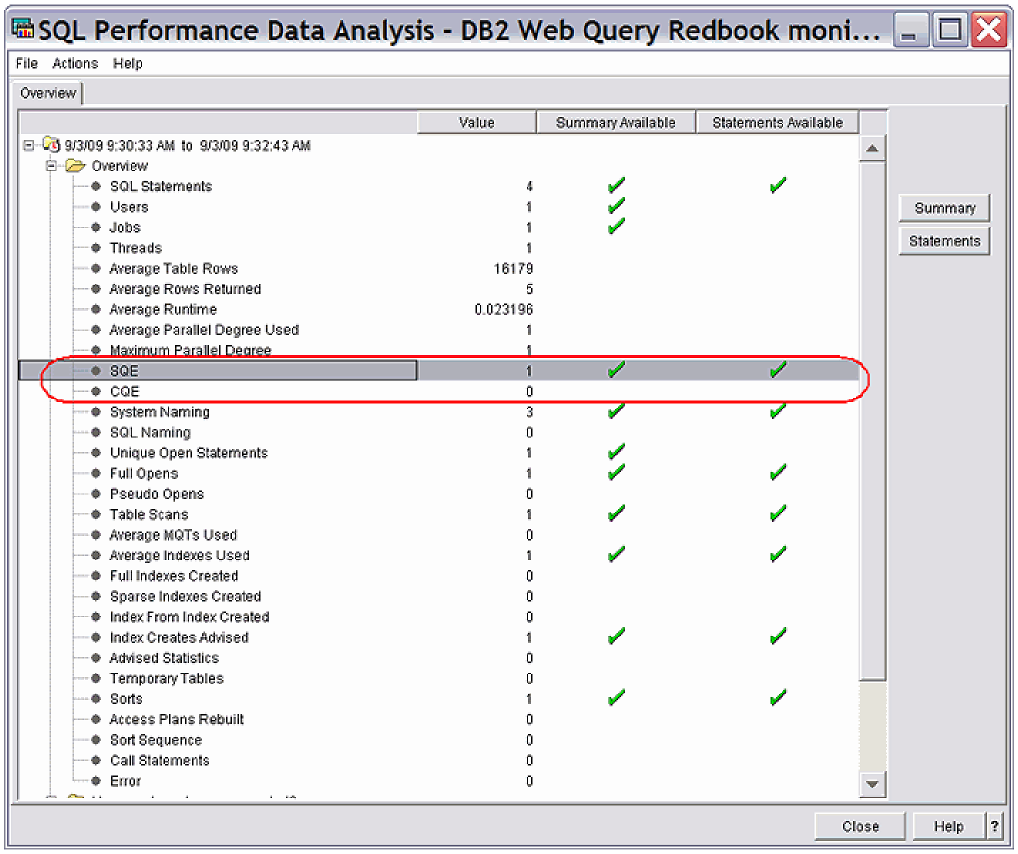
9. Now you can analyze the results. Again, right-click the monitor entry. This time select the **Analyze** option, as shown in Figure 23-16.



Name	Type	Status	Schema	Table
COBBG DBM_0902240096462002	Detailed	Ended	COBBG	DBM_090224
COBBG DBM_09022X0096462003	Detailed	Ended	COBBG	DBM_09022X
DB2 Web Query Redbook monitor	Detailed	Ended	QGPL	QZG0000016
QGPL DBMONFILE 1261550001	Detailed	Ended		DBMONFILE

Figure 23-16 Analyze SQL Performance Monitor

The SQL Performance Data Analysis dashboard window is launched. Under the Overview category, find the lines for SQE and CQE. As displayed in Figure 23-17, the dashboard shows the value of 1 for SQE and 0 for CQE. This means that your query was processed by SQE.


SQL Performance Data Analysis - DB2 Web Query Redbook moni...
File
Actions
Help
Overview
Summary
Statements
Close
Help
?

9/3/09 9:30:33 AM to 9/3/09 9:32:43 AM

Overview

	Value	Summary Available	Statements Available
SQL Statements	4	✓	✓
Users	1	✓	
Jobs	1	✓	
Threads	1		
Average Table Rows	16179		
Average Rows Returned	5		
Average Runtime	0.023196		
Average Parallel Degree Used	1		
Maximum Parallel Degree	1		
SQE	1	✓	✓
CQE	0		
System Naming	3	✓	✓
SQL Naming	0		
Unique Open Statements	1	✓	
Full Opens	1	✓	✓
Pseudo Opens	0		
Table Scans	1	✓	✓
Average MQTs Used	0		
Average Indexes Used	1	✓	✓
Full Indexes Created	0		
Sparse Indexes Created	0		
Index From Index Created	0		
Index Creates Advised	1	✓	✓
Advised Statistics	0		
Temporary Tables	0		
Sorts	1	✓	✓
Access Plans Rebuilt	0		
Sort Sequence	0		
Call Statements	0		
Error	0		

Figure 23-17 SQL Performance Data Analysis dashboard

23.4.3 Indexes

Like any other application that accesses information from the database, efficient DB2 Web Query performance heavily depends on having the correct indexes in place. When your database is in production, it might be difficult to implement recommended database design practices such as database normalization. However, at this point, you can still implement an indexing strategy that helps to optimize the performance of your query reports.

Indexes over your database tables have the following advantages:

- Provide statistics to the optimizer:

  Indexes against the queried tables give the optimizer a better opportunity to select the most efficient access method. This is because they provide relevant statistics and information, such as the average number of duplicate values and column cardinality of the tables being queried. Such useful information provided to the optimizer results in a better access plan and a better performing query.

- Improve efficiencies:

  The optimizer can choose to use the index during implementation, thus avoiding more costly implementation alternatives such as table scans or creation of temporary structures. If a table scan is performed, every row in the database table must be read. Depending on the size of the tables and the complexity of the query, this an be a lengthy process and can consume a significant share of system resources.

- Ensure uniqueness:

  A unique index on a column ensures that no duplicate values exist for that column.

Using indexes might result in the following ramifications:

- As mentioned previously, indexes can speed data retrieval. In some cases, requests might only need to use the index, never accessing the actual data (a condition referred to as *Index Only Access*).
- Even secondary indexes can be used in the case of data selection statements.
- Indexes add overhead to a database. Indexes must be maintained by the database whenever the data in the underlying table changes. Sometimes DBAs are reluctant to add indexes for reporting application efficiency if the database is one that is not dedicated to reporting purposes.

Indexing strategies is a broad topic that is covered in detail in the white paper *IBM DB2 for i indexing methods and strategies*. You can download this paper from the Web at the following address:

https://www.ibm.com/partnerworld/wps/servlet/ContentHandler/stg_ast_sys_wp_db2_i_indexing_methods_strategies

In addition, the following tips can help you get started:

- Take a proactive approach and make sure that there are indexes available over all of the selection, joining, ordering, and grouping columns of your queries.
- In a reactive mode, run your queries and use the available database feedback mechanisms to determine what indexes the optimizer wants created.

  When a query is executed, the database engine provides index advice during the optimization phase. This occurs when it determines that a useful index does not exist against the tables that are being queried. It makes a recommendation that the index be created. Afterward, this advice can be obtained from various sources, including the following sources:

  – Index Advisor
  – SQE Plan Cache
  – SQL Plan Cache snapshot
  – SQL Performance Monitor
  – Visual Explain

  The index advisories can be extracted from these sources of optimizer feedback and used to create the recommended indexes.

  Important: When an advised index is created and the query is run again, you might observe that the optimizer does not use that index during implementation. Indexes are sometimes recommended for the information that they can provide to the optimizer. This information is used to help the optimizer cost each access method. It might also help the optimizer determine that the advised index is not the optimal one to use during query execution.

For more information about SQL and database performance analysis, refer to *OnDemand SQL Performance Analysis Simplified on DB2 for i5/OS in V5R4*, SG24-7326.

23.4.4 Available hardware

Hardware is certainly a critical component when considering the factors that can impact an application's performance. Insufficient resources usually mean that users sit and wait while the work is being done. Typically, older hardware equates to fewer resources: slower processors, less memory, and overall inferior throughput. While DB2 Web Query can run on any system with V5R4 or later, newer models do offer the advantages of more processing power and better overall throughput.

The product makes heavy use of JAVA and SQL, two technologies that will consume system resources. The bottom line is that factors such as CPW ratings and amount of available memory do matter when it comes to DB2 Web Query performance. This is true of any application running on the system. If it has been a while since your last upgrade, you must evaluate your performance expectations and decide whether you want to run modern software on a server that is, for example, 12 years old.

A balanced system

While sufficient hardware resources to run your workloads is critical factor, it is also important that you have a balanced hardware configuration. What does this mean? It means that simply adding more memory or another processor to your system may not improve overall performance and create an unbalance in the environment. This is because when it comes to driving workloads, the ultimate goal is to keep the processor busy while minimizing wait times (queuing). To help illustrate this, consider the following dish-washing analogy.

Let us say that you own a restaurant and one of the workloads to be performed is washing dishes. In this *subsystem* the busser (term used for both busboy and busgirl) brings the dirty dishes and places them in the sink and the dishwasher (a person) takes the dishes out of the sink and washes them.

Each component in this process can be analogous to a component in your hardware configuration:

- The dishwasher = the processor
- The sink = memory
- The bussers = disk units/arms

The goal is to keep the dishwasher busy (as long as there is work to be done) so that the dishes are cleaned as fast as possible. Therefore, acceptable performance and a balanced system are largely dependent on the following considerations:

- The speed of the dishwasher
- The number of dishes to be washed
- Performance expectations

Let us say that the process is not going as fast as you would like and not enough dishes are being cleaned. Your performance expectations are not being met. So you take these steps:

1. You walk into the kitchen and discover that Frank, the dishwasher, is waiting for work. This means that queuing is occurring earlier in the process. You observe that there are four bussers (working frantically) and they cannot bring dishes into the sink fast enough (queuing at the disk I/O level is occurring). So you hire four more bussers.
2. This corrective action helps and Frank becomes busier but is still waiting for work. So, again, you hire four additional bussers. This has no effect on performance because now the bussers are queued up, waiting to place dishes in the sink because the sink is full (not enough memory). This is an example of an unbalanced system.
3. Showcasing your sharp business acumen, you purchase a bigger sink (more memory). This eliminates the queuing at the sink and Frank stays busy all the time. You now have a balanced system, but you are still not satisfied with the pace of which the dishes are being washed.
4. Because it worked before, you decide to trade in that sink for one that is even larger. You then observe that many more dishes can be placed in the sink, but it does not help overall performance whatsoever, This is because the dishes are piled up in a big heap in the sink and Frank cannot work any faster (too much memory). At this point the system is again out of balance.
5. Evaluating the situation, you ascertain that hiring more bussers will not help. And an even bigger sink would be a foolish purchase. You conclude that there are two choices: you can either fire Frank and hire a faster dishwasher (upgrade) or hire another dishwasher to help Frank out (add another processor). Frank is a good guy and has never called in sick once in the three years he has been employed, so you decide to help him out and hire his third cousin Gus. This new hire proves to be the right move: Frank and Gus become a formidable dishwashing duo, producing clean plates at a brisk and more than acceptable pace. The number of bussers is optimal, as is the size of the sink, thus there is no queuing. You finally have a balanced system that meets your performance expectations.

This simple analogy is meant to illuminate the fact that the amount of work to be done, performance expectations, and a balanced system must be considered when it comes to evaluating your hardware resource needs.

Workload estimator for DB2 Web Query

Determining the system resources needed to run your query workloads while maintaining a balanced system is not a trivial task. To help you with this venture, IBM has provided a Work Load Estimator (WLE) for DB2 Web Query. This tool is a sizing guide that is provided to help you estimate your hardware resource needs for DB2 Web Query workloads and is part of the IBM suite of sizings guides. It is based on a series of workload benchmarks defined and performed by the IBM STG Lab Services team.

Once you are into the DB2 Web Query WLE (Figure 23-18 on page 751), you define the workload characteristics to the tool. Information such as this should be included:

- Users:

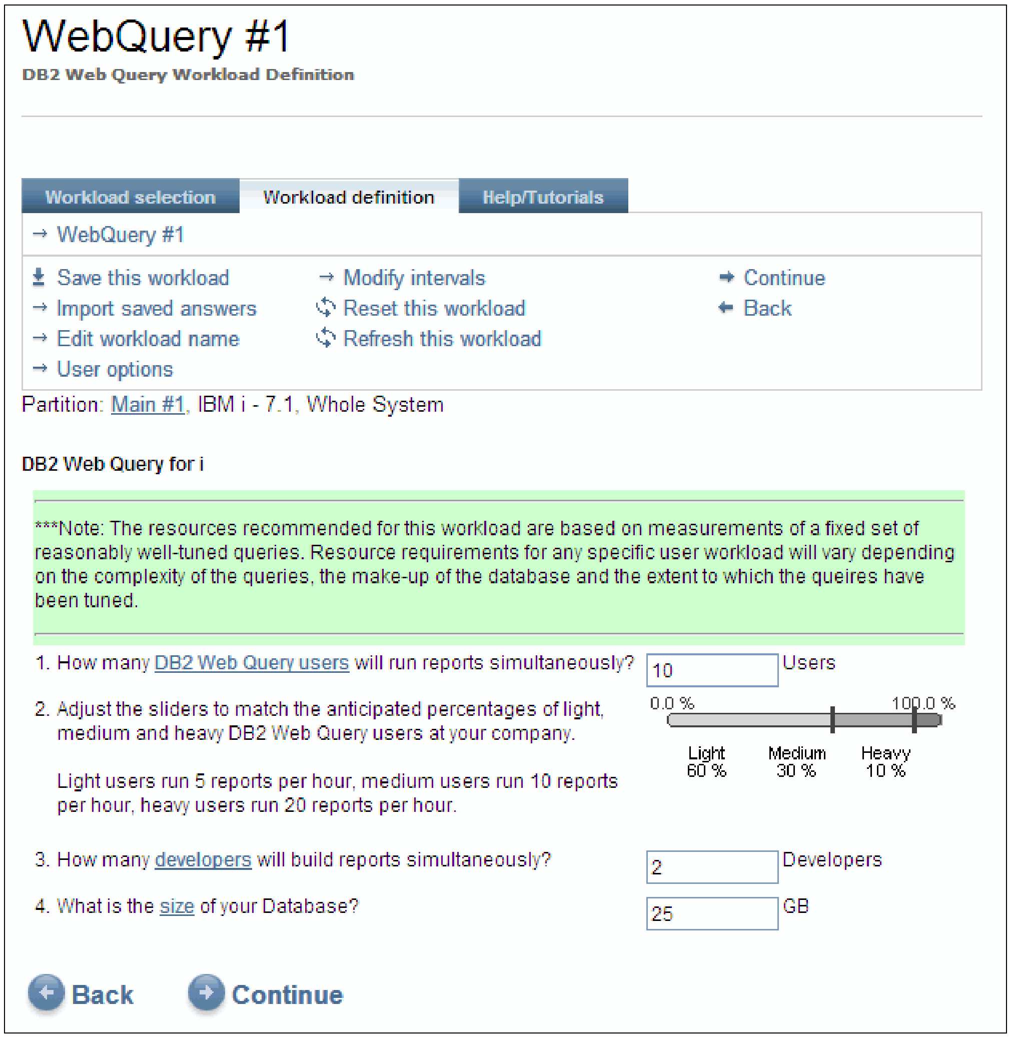
  Includes the number of concurrent users and developers and the types of users (heavy, medium, light).

- Environment:

  Attributes about the environment in which the DB2 Web Query product will be running. This includes factors such as the size of your database.

- Specifics of production environment, if applicable:

  Specify whether queries are running in a shared environment (for example, against production databases) or a dedicated environment (for example, a data warehouse).


WebQuery #1

DB2 Web Query Workload Definition

Partition: Main #1, IBM i - 7.1, Whole System

DB2 Web Query for i

***Note: The resources recommended for this workload are based on measurements of a fixed set of reasonably well-tuned queries. Resource requirements for any specific user workload will vary depending on the complexity of the queries, the make-up of the database and the extent to which the queires have been tuned.

1. How many DB2 Web Query users will run reports simultaneously? 10 Users
2. Adjust the sliders to match the anticipated percentages of light, medium and heavy DB2 Web Query users at your company. 0.0 % 100.0 % Light 60 % Medium 30 % Heavy 10 %

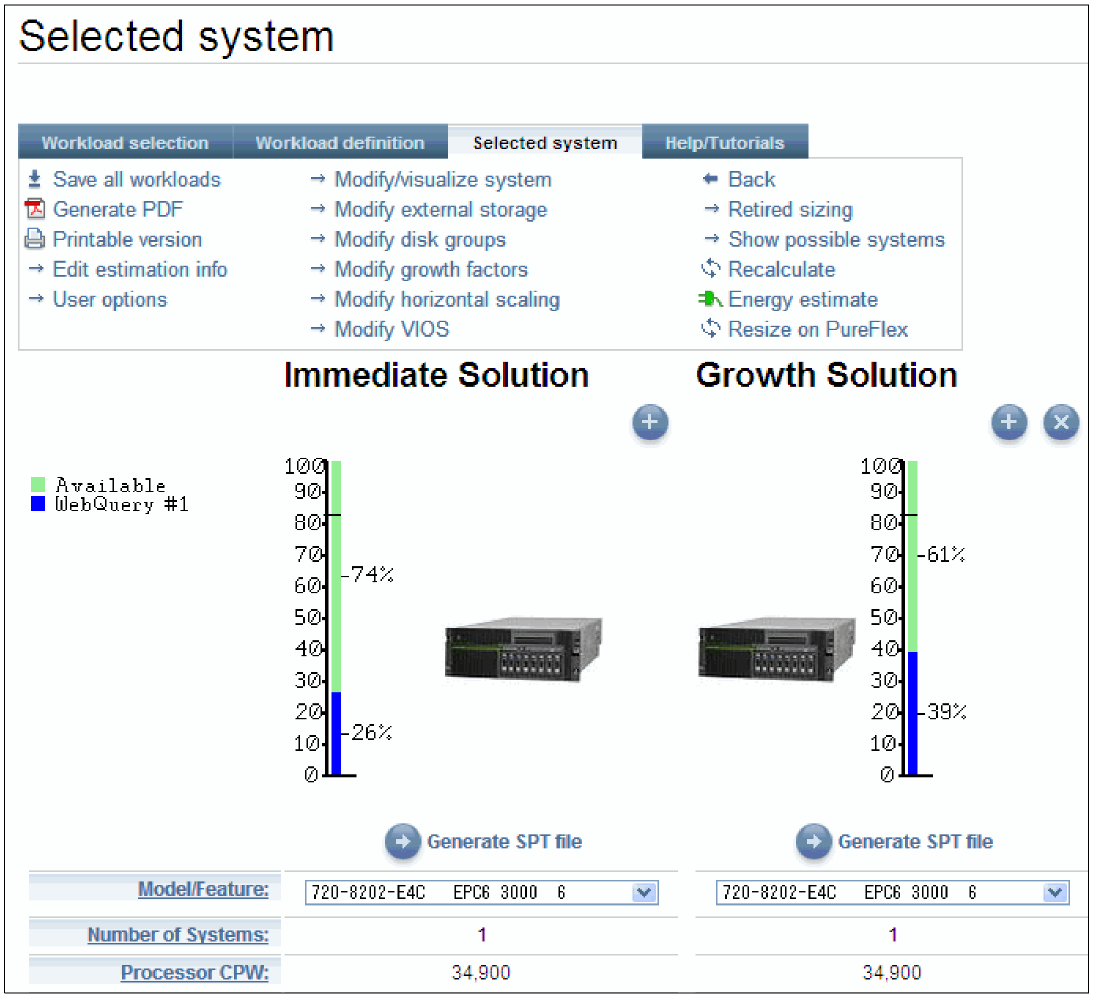
   Light users run 5 reports per hour, medium users run 10 reports per hour, heavy users run 20 reports per hour.
3. How many developers will build reports simultaneously? 2 Developers
4. What is the size of your Database? 25 GB

Figure 23-18 DB2 Web Query Workload Define

Once you have completed these steps, the sizing guide provides an estimate of the minimum hardware configuration necessary to run the defined DB2 Web Query workloads.

Note: Although DB2 Web Query does not require Power7 hardware, the DB2 Web Query WLE produces Power7 hardware recommendations only.

Figure 23-19 provides an example of the output produced by the DB2 Web Query WLE.



Selected system

Workload selection | Workload definition | Selected system | Help/Tutorials

- Save all workloads
- Generate PDF
- Printable version
- Edit estimation info
- User options
- Modify/visualize system
- Modify external storage
- Modify disk groups
- Modify growth factors
- Modify horizontal scaling
- Modify VIOS
- Back
- Retired sizing
- Show possible systems
- Recalculate
- Energy estimate
- Resize on PureFlex

	Immediate Solution	Growth Solution
Model/Feature:	720-8202-E4C EPC6 3000 6	720-8202-E4C EPC6 3000 6
Number of Systems:	1	1
Processor CPW:	34,900	34,900

Figure 23-19 Sample output of Workload Estimator for DB2 Web Query

Important:

- The use of the Workload Estimator and its results are restricted to the purpose of helping you predict a possible system solution. These results are estimates and averages based on certain assumptions and conditions, and are based on measurements and analysis with a variety of workloads (internal to IBM and by third parties), performance characterizations of systems hardware and software, and best performance practices. Ensure that realistic inputs have been provided for the high-level configuration, the workload definitions, and user options. Actual customer results may differ significantly.
- The system recommendations are based on measurements of a fixed set of reasonably well-tuned queries. Resource requirements for any specific user workload will vary depending on the complexity of the queries, the make-up of the database, and the extent to which the queries have been tuned.

23.4.5 Concurrent users

Unless they sign on to the system and do nothing, application users typically equate to more workload on the system. More units of work means that more system resources are being consumed. You may have a balanced system with satisfactory performance, but at some point (as you add more users or those existing users request more work or work that is more resource intensive) you will either experience queuing or a processor that cannot keep up.

Getting back to the dish-washing analogy, as the word spreads and your restaurant gains popularity, you suddenly have an influx of new customers. More customers means more dirty dishes that must be washed. Frank and Gus may need some help, perhaps in the form of more bussers, a larger sink, or another dishwasher. The balanced system may need an adjustment.

Much of this is obvious, but you may find it difficult to find the right balance. This again is where the DB2 Web Query WLE can prove to be helpful, because it does consider the number of concurrent users in its calculations.

23.5 Performance case study

In this section, a case study is conducted against a long-running DB2 Web Query report to delve further into performance analysis and demonstrate available optimization techniques. The objective is to determine why the particular report has an excessive runtime and provide example approaches in order to reduce the overall runtime.

In this section, we describe the following steps:

1. Identify a long-running report.
2. Perform analysis, looking for optimization disablers.
3. Determine report-tuning options.
4. Create an SQL view and synonym and change the report to use view.
5. Create a new report based on the SQL view.
6. Perform additional database analysis and tuning.

23.5.1 Identifying a long-running report

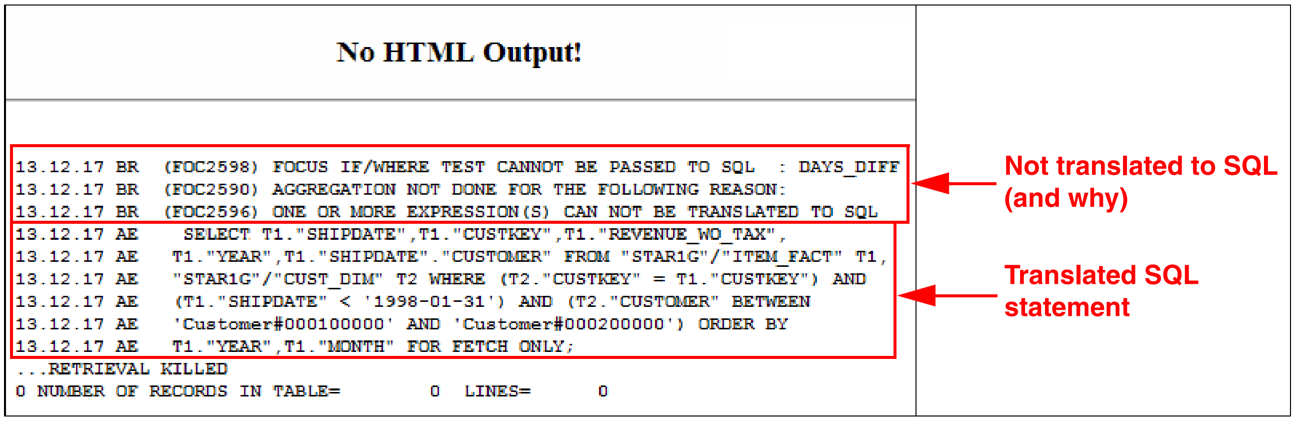
During creating reports, we may see a report that require an excessive amount of time to run to completion. However, reports that were similar in nature, meaning that they had the same general format and approximately the same number of rows returned, ran in seconds.

23.5.2 Performing analysis and looking for optimization disablers

To help determine the reasons behind the substantial runtime, DB2 Web Query provides the Run w/SQL Trace option to run the report and generate an SQL trace. The trace statements that are displayed help reveal any optimization disablers that might be present in the report.

Note: Report Assistant (included in DB2 Web Query 1.1.x) has the Run w/SQL Trace option but Info Assist does not have the Run w/SQL Trace now (as of DB2 Web Query 2.1 HotFix1). Since Info Assist will have the function later, this section provides a performance analysis with the Run w/SQL Trace.

When the report is run using the Run w/SQL Trace option, the SQL trace information is displayed as HTML output to the browser, as shown in Figure 23-20.

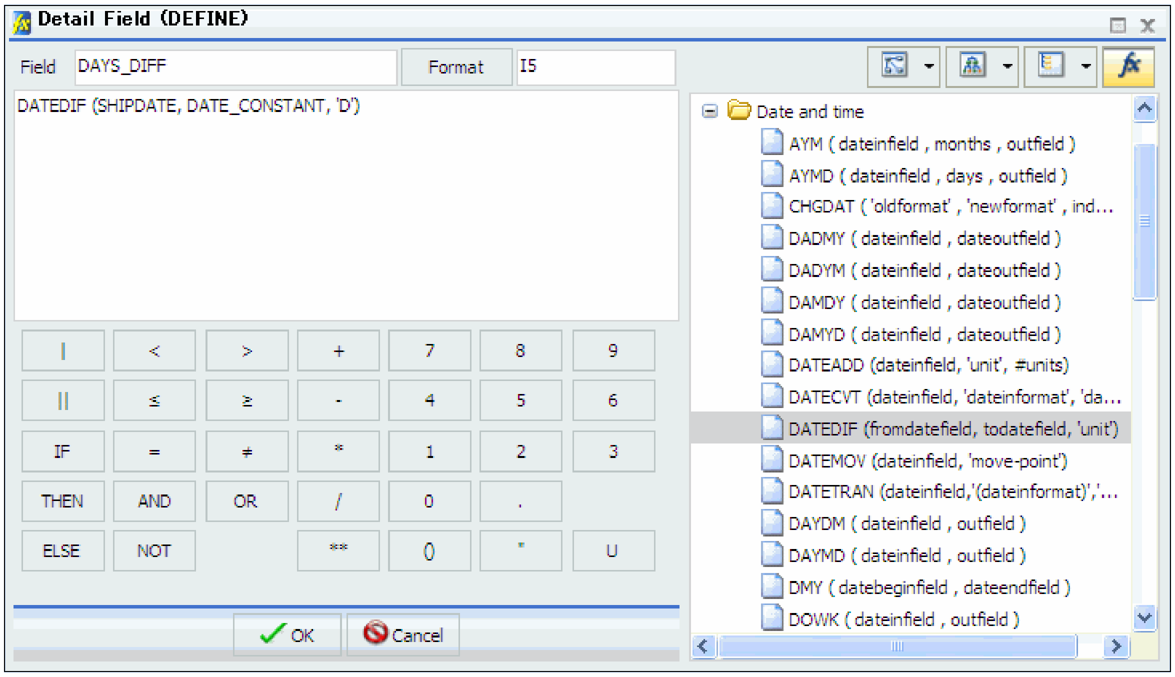


```
13.12.17 BR   (FOC2598) FOCUS IF/WHERE TEST CANNOT BE PASSED TO SQL  : DAYS_DIFF
13.12.17 BR   (FOC2590) AGGREGATION NOT DONE FOR THE FOLLOWING REASON:
13.12.17 BR   (FOC2596) ONE OR MORE EXPRESSION(S) CAN NOT BE TRANSLATED TO SQL
13.12.17 AE     SELECT T1."SHIPDATE",T1."CUSTKEY",T1."REVENUE_WO_TAX",
13.12.17 AE    T1."YEAR",T1."SHIPDATE"."CUSTOMER" FROM "STAR1G"/"ITEM_FACT" T1,
13.12.17 AE    "STAR1G"/"CUST_DIM" T2 WHERE (T2."CUSTKEY" = T1."CUSTKEY") AND
13.12.17 AE    (T1."SHIPDATE" < '1998-01-31') AND (T2."CUSTOMER" BETWEEN
13.12.17 AE    'Customer#000100000' AND 'Customer#000200000') ORDER BY
13.12.17 AE    T1."YEAR",T1."MONTH" FOR FETCH ONLY;
...RETRIEVAL KILLED
0 NUMBER OF RECORDS IN TABLE=        0  LINES=      0
```

Figure 23-20 Report trace results

This trace reveals a couple of interesting points. First, it shows that part of the report cannot be translated to SQL. In this case, translation failed for the Defined field DAYS_DIFF because of an unsuccessful IF/WHERE Test. Second, notice how the trace shows the SQL statement that is generated (and what is ultimately submitted to the database engine). Even though part of the translation failed, the Reporting Server was able to construct a meaningful SQL statement.

Using the trace, you can conclude that the problem with this report is the use of the Define field DAYS_DIFF. The line in the trace `FOCUS IF/WHERE TEST CANNOT BE PASSED TO SQL. DAYS_DIFF` indicates that DB2 Web Query attempted (and failed) to generate an equivalent search condition for the SQL statement's WHERE clause (Figure 23-21).


Detail Field (DEFINE)
Format
Date and time
AYM (datefield , months , outfield)
AYMD (datefield , days , outfield)
CHGDAT ('oldformat' , 'newformat' , ind...
DADMY (dateinfield , dateoutfield)
DADYM (dateinfield , dateoutfield)
DAMDY (dateinfield , dateoutfield)
DAMYD (dateinfield , dateoutfield)
DATEADD (dateinfield, 'unit', #units)
DATECVT (dateinfield, 'dateinformat', 'da...
DATEDIF (fromdatefield, todatefield, 'unit')
DATEMOV (dateinfield, 'move-point')
DATETRAN (dateinfield,'(dateinformat)','...
DAYDM (dateinfield , outfield)
DAYMD (dateinfield , outfield)
DMY (datebeginfield , dateendfield)
DOWK (dateinfield , outfield)
OK
Cancel

Field: DAYS_DIFF
Format: I5

```
DATEDIF (SHIPDATE, DATE_CONSTANT, 'D')
```

Figure 23-21 Define field DAYS_DIFF

Anytime that a Define field can be passed to the database engine, a more efficient report results, particularly when the Define field is part of the selection process, which it is the case in our example. If the Define field cannot be passed to the database engine, then all the rows from the result set of the generated SQL statement are returned to the Reporting Server, which now is responsible for the selection (and sorting if the report is sorted by the Define field). This kind of behavior can negatively impact performance and should be avoided if at all possible. Again, we must emphasize that the report is more efficient if the entire database access portion of the report definition can be pushed down to the database engine for processing.

23.5.3 Determining report-tuning options

Tuning this particular report basically means selecting one of the following actions:

- Eliminate the culprit Define field.
- Modify the report in an attempt to enable SQL translation for the Define field.
- Push the Define field logic down to a database using SQL views.

For this example, the Define field is required to deliver the necessary information in the report (and therefore cannot be removed). In addition, repeated efforts to fix the translation for the Define field prove to be futile. Therefore, only the third option remains, which is to push the Define field logic down to a database using SQL views.

23.5.4 Creating an SQL view and synonym

The next step is to create an SQL view that contains all the required tables, join syntax, columns, and selection, including the DAYS_DIFF field selection that DB2 Web Query was unable to directly translate to SQL. To provide the ability to calculate the number of days between the date 1998-01-31 and the value of the SHIPDATE column, a new result column is added to the view. Example 23-4 shows this result column, also named DAYS_DIFF.

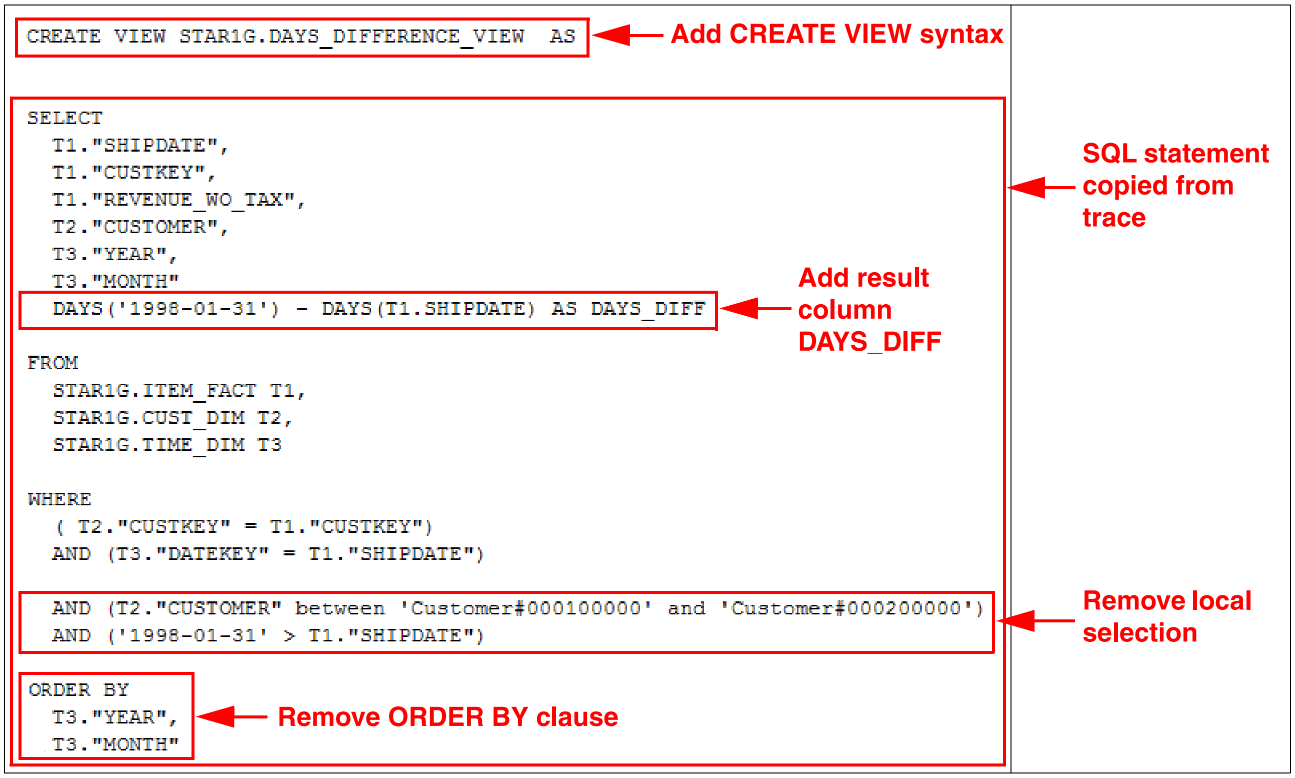
Example 23-4 DAYS_DIFF result column

```
DAYS('1998-01-31') - DAYS(T1.SHIPDATE) AS DAYS_DIFF
```

To create the new view:

1. From a System i Navigator, open a connection to the System i environment and open a Run SQL Script window.

2. Type the CREATE VIEW statement or use the SQL selection statement that is displayed in the trace as a base for this view definition and make the necessary modifications. Copy and paste the statement into the Run SQL script window and begin making the changes. For the copy and paste method, Figure 23-22 shows more details about what to add and remove from the selection statement to form the CREATE VIEW statement.



```
CREATE VIEW STAR1G.DAYS_DIFFERENCE_VIEW  AS

SELECT
  T1."SHIPDATE",
  T1."CUSTKEY",
  T1."REVENUE_WO_TAX",
  T2."CUSTOMER",
  T3."YEAR",
  T3."MONTH"
  DAYS('1998-01-31') - DAYS(T1.SHIPDATE) AS DAYS_DIFF

FROM
  STAR1G.ITEM_FACT T1,
  STAR1G.CUST_DIM T2,
  STAR1G.TIME_DIM T3

WHERE
  ( T2."CUSTKEY" = T1."CUSTKEY")
  AND (T3."DATEKEY" = T1."SHIPDATE")

  AND (T2."CUSTOMER" between 'Customer#000100000' and 'Customer#000200000')
  AND ('1998-01-31' > T1."SHIPDATE")

ORDER BY
  T3."YEAR",
  T3."MONTH"
```

Figure 23-22 Anatomy of the CREATE VIEW statement

Notice that the local selection is removed from the CREATE VIEW statement. This is added to the statement that references the view. Also observe that the ORDER BY clause is removed. This is because views cannot be ordered. Ordering is specified in the report definition.

After all the necessary modifications are made, Example 23-5 shows how the CREATE VIEW statement should look.

Example 23-5 Create SQL view

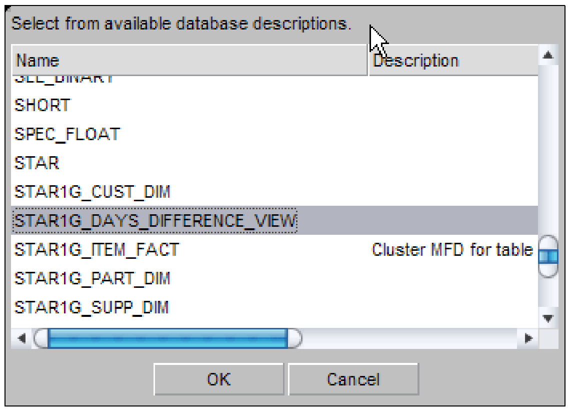
```
CREATE VIEW STAR1G.DAYS_DIFFERENCE_VIEW
   AS
   SELECT T1."SHIPDATE",T1."CUSTKEY",T1."REVENUE_WO_TAX",
      T2."CUSTOMER",T3."YEAR",T3."MONTH",
   DAYS('1998-01-31') - DAYS(T1.SHIPDATE) as DAYS_DIFF
   FROM STAR1G.ITEM_FACT T1, STAR1G.CUST_DIM T2,STAR1G.TIME_DIM T3
   WHERE (T2."CUSTKEY" = T1."CUSTKEY") AND (T3."DATEKEY" = T1."SHIPDATE")
```

3. Execute the statement to create the view.
4. From the browser, open the DB2 Web Query metadata window and create a synonym against the new view.

23.5.5 Creating a new report based on the SQL view

When the new view and its synonym are in place, a new report is created to access the contents of the view:

1. From DB2 Web Query home page, select **InfoAssist** to create a brand new report.
2. In the Select from available database descriptions window (Figure 23-23), select the new view. Click **OK**.


Select from available database descriptions.

Name	Description
SHORT	
SPEC_FLOAT	
STAR	
STAR1G_CUST_DIM	
STAR1G_DAYS_DIFFERENCE_VIEW	
STAR1G_ITEM_FACT	Cluster MFD for table
STAR1G_PART_DIM	
STAR1G_SUPP_DIM	

Figure 23-23 info Assist: selecting a new view

3. Continue creating the new report with the view, specifying the same field format and selection criteria fields (as the original report). Examples are shown in Figure 23-24 and Figure 23-25.

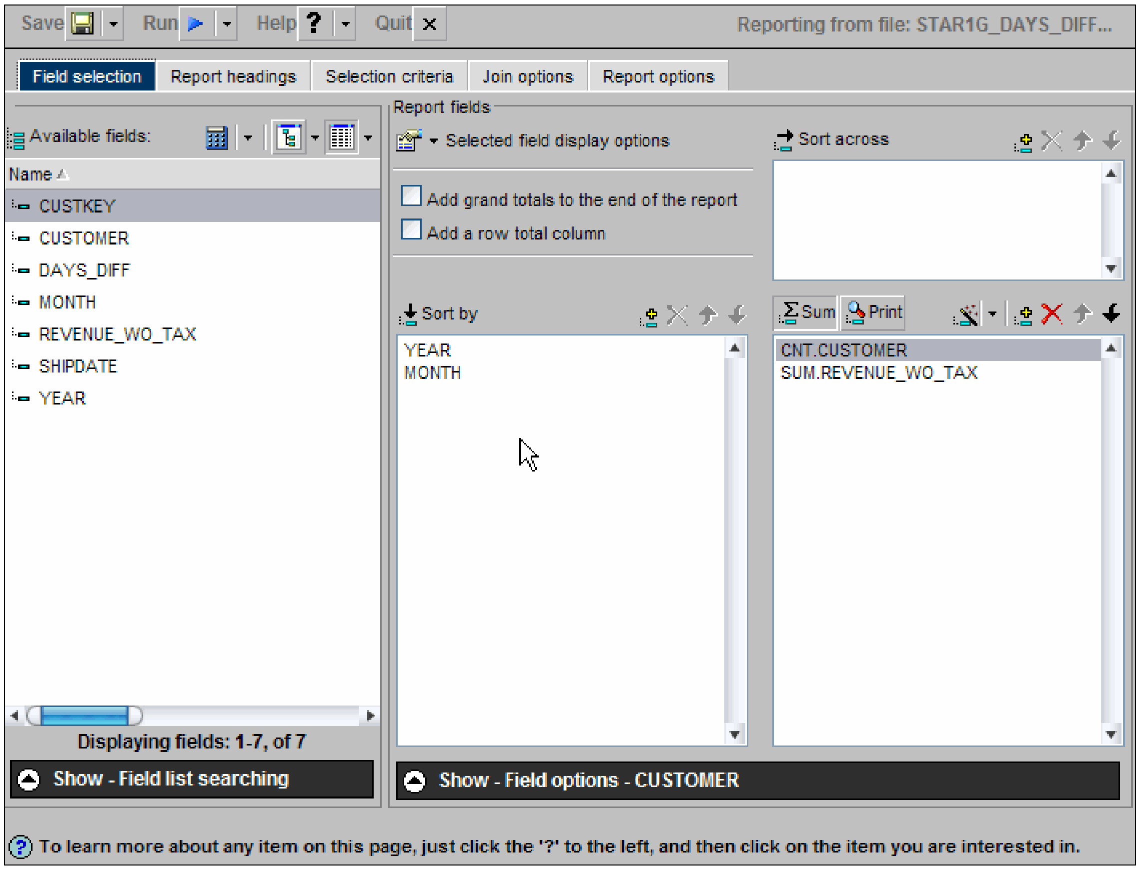
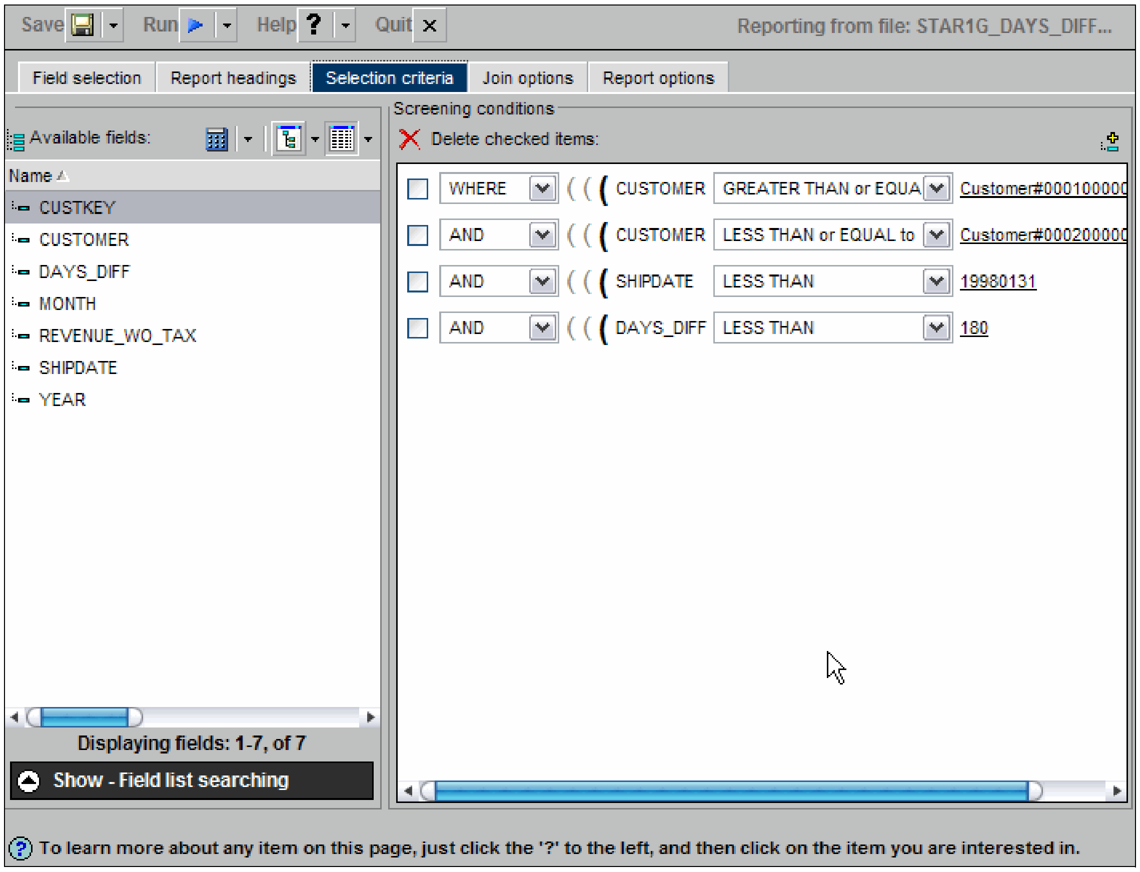

Save
Run
Help
Quit
Reporting from file: STAR1G_DAYS_DIFF...
Field selection
Report headings
Selection criteria
Join options
Report options
Available fields:
Name
CUSTKEY
CUSTOMER
DAYS_DIFF
MONTH
REVENUE_WO_TAX
SHIPDATE
YEAR
Displaying fields: 1-7, of 7
Show - Field list searching
Report fields
Selected field display options
Add grand totals to the end of the report
Add a row total column
Sort across
Sort by
YEAR
MONTH
Sum
Print
CNT.CUSTOMER
SUM.REVENUE_WO_TAX
Show - Field options - CUSTOMER
To learn more about any item on this page, just click the '?' to the left, and then click on the item you are interested in.

Figure 23-24 Report using SQL view: Field selection tab


Save
Run
Help
Quit
Reporting from file: STAR1G_DAYS_DIFF...
Field selection
Report headings
Selection criteria
Join options
Report options
Available fields:
Name
CUSTKEY
CUSTOMER
DAYS_DIFF
MONTH
REVENUE_WO_TAX
SHIPDATE
YEAR
Displaying fields: 1-7, of 7
Show - Field list searching
Screening conditions
Delete checked items:
To learn more about any item on this page, just click the '?' to the left, and then click on the item you are interested in.

WHERE	(((	CUSTOMER	GREATER THAN or EQUA	Customer#00010000
AND	(((	CUSTOMER	LESS THAN or EQUAL to	Customer#00020000
AND	(((	SHIPDATE	LESS THAN	19980131
AND	(((	DAYS_DIFF	LESS THAN	180

Figure 23-25 Report using SQL view: Selection criteria tab

4. Save the report and click **Quit**.
5. Run the new report.
6. Record performance measurements.

In the case of this example, significant improvements in performance are observed. In fact, simple benchmark measurements reveal that the report, using the view, runs over 10 times faster than the original version of the report. This is not meant to imply that you will experience the same improvements if you conduct such a tuning exercise. It is only intended to serve as an example of the kinds of efficiencies that you can obtain. Many factors can affect report performance, so your results might vary.

23.5.6 Performing additional database analysis and tuning

When your report is using the SQL view and all database processing is being handled by the database engine, your work might not yet be done. You can potentially gain further efficiencies by performing regular database analysis and tuning. You can use the following tools and technologies to assist in this effort:

- SQE Plan Cache:

  First made available in V5R2, the SQE Plan Cache is an internal, matrix-like repository that is used to store all of the statements and access plans implemented by SQE. In V5R4, an interface to this information has been made available through the System i Navigator toolset. From this interface, you can find the SQL statement generated by DB2 Web Query and begin performing your analysis.

- Visual Explain:

  Visual Explain provides a graphical representation of the optimizer implementation of a query request. The query request is broken into individual components with icons that represent each unique component. Visual Explain also includes information about the database objects that are considered and chosen by the query optimizer. Visual Explain's detailed representation of the query implementation makes it easier to understand where the greatest cost is incurred.

- Index Advisor:

  Introduced in V5R4, this feature provides an easy and quick interface to index advisories that are issued by the optimizer. If the optimizer determines that a permanent index against a reference table might be beneficial, it returns the key columns necessary to create the suggested index. The data of the system-wide Index Advisor is placed into the SYSIXADV table in the QSYS2 schema.

- Database monitor:

  The Database Performance Monitor is a set of integrated tools that is used to collect database-specific performance information for all SQL requests. It can be thought of as an SQL tracing facility, one that tracks all SQL statements, access plans used to implement the statements, resources used, and subsequent performance results. All this information is stored in a database tables, where it can be analyzed and used to identify and tune performance problem areas.

- Materialized query tables:

  As mentioned previously, an MQT is a DB2 table that contains the results of a query, along with the query's definition. Because the selection, joining, and aggregation have already been performed and the results stored in the MQT, great efficiencies can be gained if the optimizer uses this table for implementation.

- Index Only Access:

  The database optimizer can use Index Only Access if all of the columns specified in the SQL statement are represented in the index as key columns. Because all of the columns that are necessary to satisfy the request are present in the index, the database engine does not have to perform random access to the table to retrieve this data. The elimination of this additional I/O operation can result in significant improvements in query response times.

Database performance analysis and tuning are rather broad topics that are briefly discussed in 23.4, "DB2 for i optimization" on page 738. For a more extensive discussion about this subject refer to *OnDemand SQL Performance Analysis Simplified on DB2 for i5/OS in V5R4*, SG24-7326.

For this exercise, feedback from the optimizer (obtained by locating the statement in the SQE plan cache and launching Visual Explain) suggests the creation of the indexes shown in Example 23-6.

Example 23-6 Indexes created for the case study

```
CREATE INDEX STAR1G.CUST_DIM_CUSTKEY_CUSTOMER
   ON STAR1G.CUST_DIM ( CUSTKEY ASC , CUSTOMER ASC )
   PAGESIZE( 64 ) ;

CREATE ENCODED VECTOR INDEX STAR1G.ITEM_FACT_CUSTKEY_EVI
   ON STAR1G.ITEM_FACT ( CUSTKEY ASC )
   WITH 65355 DISTINCT VALUES ;
```

```
CREATE INDEX STAR1G.ITEM_FACT_CUSTKEY_SHIPDATE
   ON STAR1G.ITEM_FACT ( CUSTKEY ASC , SHIPDATE ASC )
   PAGESIZE( 64 ) ;

CREATE ENCODED VECTOR INDEX STAR1G.ITEM_FACT_SHIPDATE_EVI
   ON STAR1G.ITEM_FACT ( SHIPDATE ASC )
   WITH 65355 DISTINCT VALUES ;

CREATE INDEX STAR1G.TIME_DIM_YEAR_MONTH
   ON STAR1G.TIME_DIM ( "YEAR" ASC , "MONTH" ASC )
   PAGESIZE( 64 ) ;
```

These indexes are created and the report is run again. Simple benchmark testing reveals modest improvements to the runtime after these indexes are created.

23.6 Performance benchmark

A performance benchmark was conducted to help determine DB2 Web Query runtime performance expectations relative to the equivalent Query/400 objects and SQL statements.

23.6.1 Objectives

The benchmark had the following objectives:

- Measure the overhead and resource usage of using the DB2 Web Query product to execute existing Query/400 reports.
- Measure the overhead and resource usage of running DB2 Web Query reports versus equivalent SQL statements.

23.6.2 Scenarios

The benchmark test bucket included six scenarios, each with a set of eight different queries or statements. Each query was run with 10 different sets of host variable values to measure variances in selectivity and cardinality. We measured the following scenarios of queries:

- Query/400 *QRYDFN objects (using the IBM i RUNQRY command)
- DB2 Web Query running the *QRYDFN objects
- Direct SQL statements
- DB2 Web Query reports using the DB2 CLI adapter
- Direct SQL statements with all full open and re-optimization
- DB2 Web Query reports using the DB2 CLI adapter with all full open and re-optimization

For each measurement, we captured the following data:

- Total execution time for a given number of loops of the executed queries and reports
- Performance monitor data
- Task-profile trace, also known as TRPOF

  This is a sample-based trace that queries the processor, at user-defined intervals, to gather data on what is currently running on the processor (or processors).

23.6.3 Database and system configuration

For all of the benchmark measurements, a sample database with approximately 1 GB of data was used. The measurements were done on a Model 515 System i environment with one or two processors, 7.5 GB of memory, and six disk arms. After running sets of measurements on the system, it was determined that one processor was a better match for the amount of memory and number of disk arms on the system. All measurements reported were done with a one-processor configuration.

23.6.4 Metrics

The measurement data was summarized into four key metrics:

- Minimum average response time per query
- Maximum throughput
- CPU usage
- Memory usage

Minimum average response time per query

The metric for minimum average response time shows the average response time per query for a single user. The results of this metric are shown in Figure 23-26. The average response time (sec/query) is computed as the inverse of the measured throughput (queries/sec) for a single user running back-to-back queries with no think time.

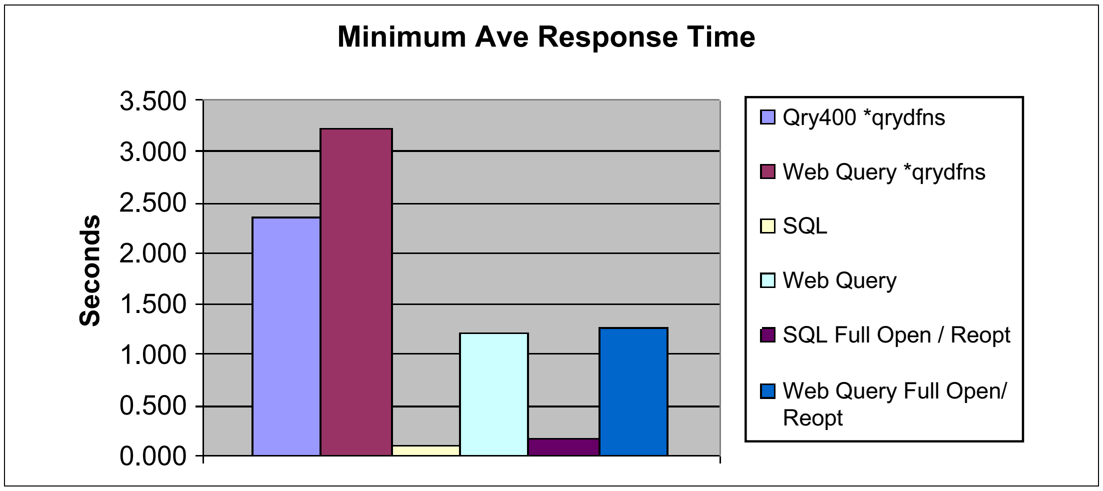


Figure 23-26 Minimum average response time chart

Figure 23-26 shows that there is a moderate response time increase for *QRYDFN objects that are run through DB2 Web Query compared to *QRYDFN run through Query/400. Some response time increase is expected when running from a remote client. As expected, the direct SQL statements have the fastest response time. The response time of the DB2 Web Query reports, although longer than the direct SQL statements, is significantly faster than the response time of the *QRYDFN objects.

This is primarily due to the efficiencies of the SQL generated by the DB2 Web Query running through SQE. Because it is an SQL CLI application that runs in server mode, each server job is recycled after every DB2 Web Query request. This means that all queries are ad hoc in nature and do incur full opens. However, the DB2 Web Query response time is only slightly increased due to this. This is the case due to server jobs being recycled after each DB2 Web Query report. The extra overhead of more full opens is incurred in the base DB2 Web Query response time numbers as well.

Maximum throughput

The objective of measuring maximum throughput is to show the maximum number of queries per second that ran before reaching a system bottleneck, such as processor or disk utilization. See Figure 23-27. This limit is measured by increasing the number of users who are running queries until no additional throughput is gained.

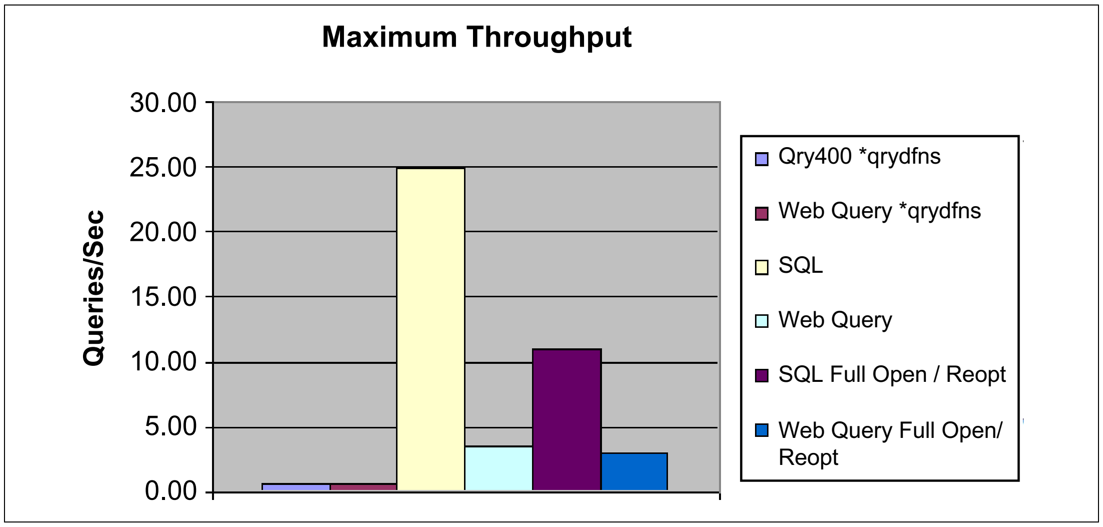


Figure 23-27 Maximum throughput chart

The maximum throughput for *QRYDFN objects that run through the Web Query interface is almost equivalent to the maximum throughput of running the *QRYDFN objects through Query/400. As expected, the executed SQL statements have the greatest throughput. DB2 Web Query introduces additional overhead compared to SQL.

However, the maximum throughput for the DB2 Web Query reports is much greater than the throughput of the DB2 Web Query *QRYDFN runs. This is due to the efficiencies of the SQL generated by the Web Query product running through SQE.

With all queries incurring full open and optimization, more like ad hoc queries, the Web Query maximum throughput is only reduced slightly. This is the case due to server jobs being recycled after each Web Query report. Therefore, the extra overhead of more full opens occurs in the base Web Query throughput numbers as well.

Resource usage: CPU

The chart of CPU resource usage in Figure 23-28 shows the amount of CPU resource used per query on average. The CPU utilization is measured when the system was running at maximum throughput. The CPU usage is then calculated as the CPU utilization divided by the queries per second at the maximum throughput point.

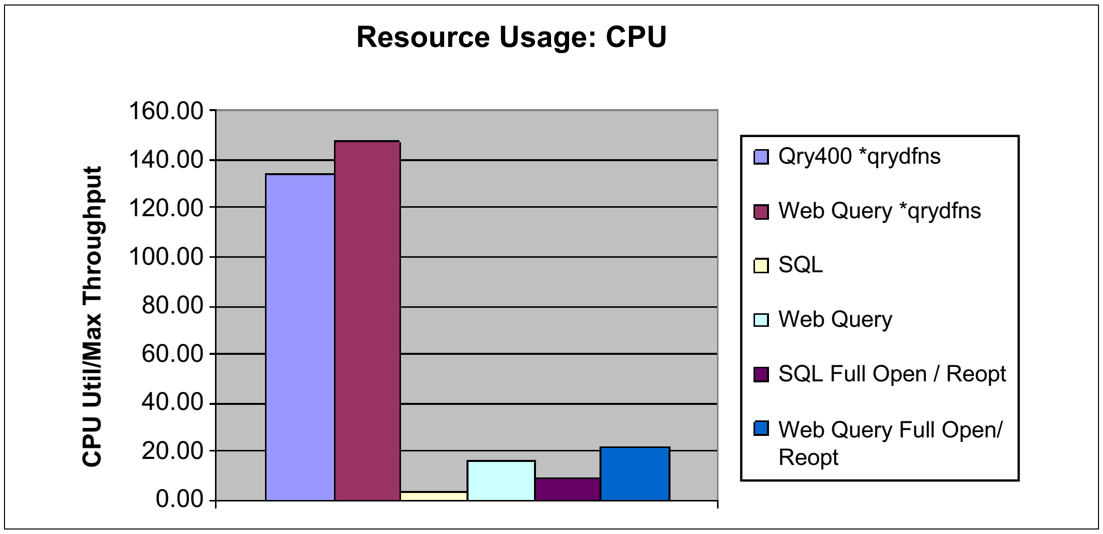


Figure 23-28 CPU usage chart

Figure 23-28 shows a small increase in CPU resource used when *QRYDFN objects are run through the Web Query interface compared to the CPU used by *QRYDFN objects run through Query/400. The chart also shows the dramatic reduction in CPU used by Web Query reports and SQL compared to the *QRYDFN objects. This is due to the efficiencies of SQL running in SQE. The Web Query reports use more CPU than the equivalent plain SQL.

Resource usage: memory

The chart of memory resource usage shows the minimum amount of base pool memory needed to be able to maintain the corresponding maximum throughput rates shown in the maximum throughput chart (Figure 23-29).

The memory requirement is determined by collecting a memory curve. Throughput for a fixed number of users is measured at various base pool sizes. The smallest memory size measured with the high level of throughput is the memory requirement point. DB2 Web Query running *QRYDFN objects compared to *QRYDFN objects run through Query/400 require more memory. This is expected due to the additional server jobs used by DB2 Web Query. As expected, the plain SQL queries require the least amount of memory. DB2 Web Query reports, running at a higher throughput rate, require about the same amount of memory as the *QRYDFN objects run through Query/400.

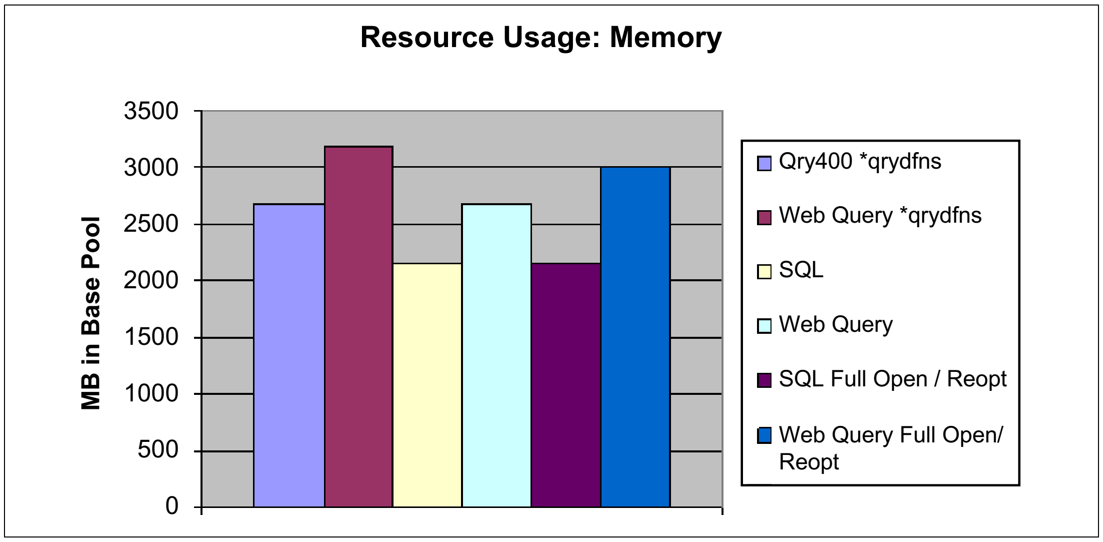
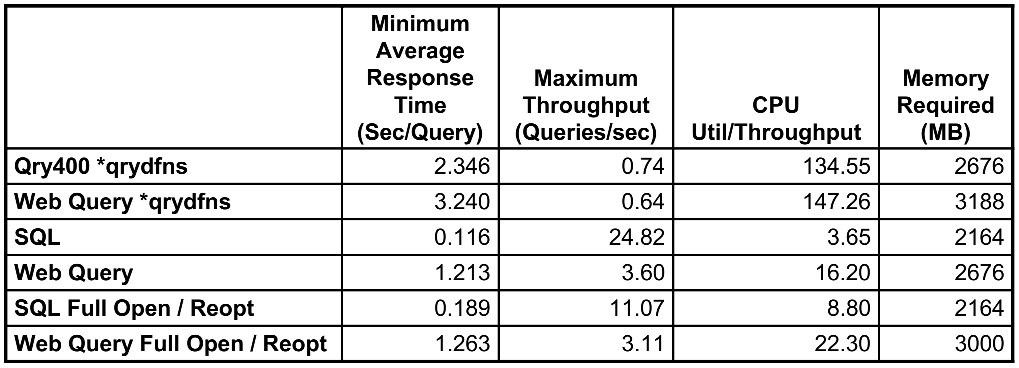


Figure 23-29 Memory usage chart

Figure 23-30 lists the data used in the graphs in this section.



	Minimum Average Response Time (Sec/Query)	Maximum Throughput (Queries/sec)	CPU Util/Throughput	Memory Required (MB)
Qry400 *qrydfns	2.346	0.74	134.55	2676
Web Query *qrydfns	3.240	0.64	147.26	3188
SQL	0.116	24.82	3.65	2164
Web Query	1.213	3.60	16.20	2676
SQL Full Open / Reopt	0.189	11.07	8.80	2164
Web Query Full Open / Reopt	1.263	3.11	22.30	3000

Figure 23-30 Benchmark results table

23.6.5 Conclusions

We used the results of the benchmark to derive the following conclusions:

- Comparisons of the DB2 Web Query *QRYDFN versus the Query/400 *QRYDFN show an increased response time. This result is expected when running to a client application as is the case with DB2 Web Query. The server CPU used is 9% more, and more memory is required.
- Comparisons of DB2 Web Query (using DB2 CLI adapter) reports to Query/400 *QRYDFN are positive. Even with running to a client, the DB2 Web Query reports run faster, use much less CPU, and require about the same amount of memory. This can primarily be attributed to the DB2 CLI adapter generating SQL and using the new SQL Query Engine (SQE).
- Comparisons of Web Query to SQL show that DB2 Web Query has longer response times, which again is expected when running to a client, and uses more resources. The CPU used by Web Query is two to four times the CPU used when running SQL, depending on whether the full open re-optimization was forced every time. Note that Web Query recycles the server jobs for every query.
- If you are using DB2 Web Query to modernize Query/400 queries (by creating new reports), there is little additional overhead.
- If you are creating new reports, always keep in mind that SQL is being generated. Therefore, an understanding of SQL performance basics is vital.

24

Migration considerations

This chapter describes the migration considerations when you update Web Query for IBM i from release 1.1.x. We first introduce the new concepts and terms available in Web Query 2.1.0, then describe migrating from 1.1.x.

24.1 New concepts and terms in Web Query 2.1.0

Effective with this release, the Report Assistant, Graph Assistant, Power Painter, Web development tools and the Developer Workbench Graph Assistant tool have been replaced with Web-based Rich Internet Application facilities. This aligns with the on-going strategy to consolidate tool sets and to standardize on a common look-and-feel across all report development products.

Info Assist Basic is a modern, Web-based facility that provides comparable functionality as found in Report Assistant, as well as over 80 chart types. It will include functionality equivalent to that found within Report Assistant and Graph Assistant.

24.1.1 Business Intelligence Portal

The Business Intelligence Portal (BI Portal) is new in Web Query 2.1.0 and is the driving force behind the new user interface. It is the successor to the Business Intelligence Dashboard (BI Dashboard). It does everything that BI Dashboard does, and more.

The BI Portal is about being able to build complete, modern Web sites. The end user experiences the drag-and-drop features that are available in popular online portals. This is a key point, as end users do not need to learn anything new. This results in no training and a high usage rate.

24.1.2 New Web Query Administration User ID

QWQADMIN is the new Web Query Administrator user ID that has the ability to manage users and configuration settings for a Web Query installation. The QWQADMIN user ID has the following attributes:

- Does not take up a named user license.
- Can add and remove users to Web Query.
- Can add or remove other Web Query administrators.
- Can grant users authorization to specific roles by adding them to groups.
- Can associate group profiles to specific folder run group(s).
- Can launch the Report Broker and Administration Consoles to configure the Web Query installation.
- Cannot manage folders, procedures, schedules or metadata.
- Has ownership of all the metadata.

24.1.3 Web Query folders

For Web Query 2.1.0, almost all content is stored in the Web Query Client Repository (DB2 Tables) as shown in Figure 24-1. The only exception is Web Query metadata. Synonyms are still stored in the IFS in application directories.

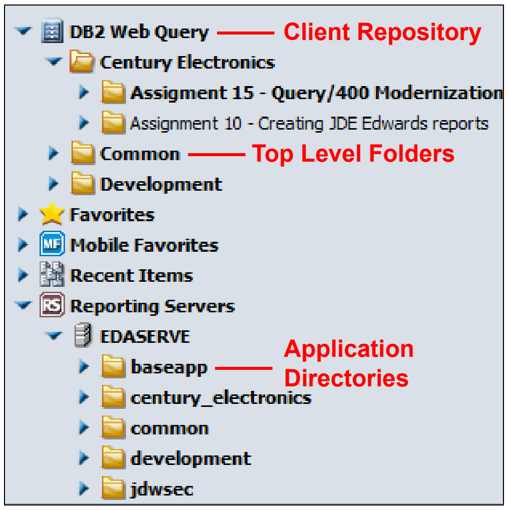


Figure 24-1 Folders structure

Web Query content consists of procedures, HTML files, Stylesheets, images, Report Broker schedules, and distribution lists. This content is stored in one or more folders.

Top level folders

Top level folders reside at the top of the Web Query Client Repository and are used to segregate applications. For example, you may want to have a top level folder for Sales and another folder for Human Resources (HR) where users can be assigned to one or more folders. A user can have a different set of roles for each folder. For example, a user can be assigned to only run reports in HR but takes on a developer/dba role in Sales.

Each top level folder is created with its own set of Web Query Groups that define the authorization rules for the folder. For details, see "Web Query Groups" on page 770.

The Common top level folder exists for all Web Query installations. The purpose of this folder is to contain content that can be run by all users. If you prefer not to display the Common top level folder, there is an option to hide it.

Hiding the Common top level folder

Note: Customers that have migrated from Web Query 1.1.x will have their Common Domain contents migrated to the Common top level folder.

Here is how to hide the common top level folder:

1. Login to Web Query using the QWQADMIN user ID.
2. Right-click the Common top level folder and click **Hide**.

Application directories and metadata management

Application directories are where synonyms are created. They are IFS directories that map to the following path:

```
/qibm/UserData/qwebqry/apps
```

In Web Query 1.1.x, the baseapp application directory is the default location for newly created synonyms. Synonyms in the baseapp directory are accessible from all Domains. If Developer Workbench was licensed, it could be used to create new application directories which could be linked to a Domain's application directory path.

In release 2.1.0, the baseapp application directory fulfills the same role in 1.1.x. However, a new application directory is created for every top level folder and is automatically linked to the folder as the 1st directory of the application directory path.

The application directory path is searched whenever a synonym is required to do the following operations:

- Develop a new procedure
- Edit an existing procedure
- Run an existing procedure

The automatic creation of this application directory and making it first in the application directory path allows you to segregate synonyms without using Developer Workbench. Synonyms that only pertain to one top level folder or application should be created in that folder's application directory. Synonyms that need to be shared across all applications should be copied or created in the baseapp directory.

Note: If a synonym with the same name exists in both application directories of the application directory path, the first one found in the path will be used.

Web Query Groups

Web Query Groups are pre-defined to represent a specific set of functions or role. Global Groups define a role at the Web Query product level and apply across folders, whereas folder Groups define a role at the top level folder level.

The two global Groups are defined in Table 24-1.

Table 24-1 Global groups

Group name	Role description
WebQueryAdministrator	Can perform all functions in Web Query and can access all folders.
DevWorkBench	Can connect to Web Query using Developer Workbench.

The six folder Groups are defined in Table 24-2.

Table 24-2 Folder groups

Group name	Role description
Folder-run	Can run procedures in the respective folder.
Folder-analyst	Folder-run role plus ability to develop and run procedures in private folders.
Folder-dev	Folder-analyst role plus ability to develop, run and publish procedures in published folders within the respective folder.
Folder-dba	Can manage metadata in the respective folder's application folder.
Folder-sched	Can manage schedules and distribution lists in the respective folder.
Folder-admin	Can manage users in the respective folder. **Note:** A Folder-admin cannot acquire or release a developer or group profile license.

Each top level folder that is created will automatically have the six folder based groups created in the Web Query repository.

For example, if you add a top-level folder named Sales, the following six groups are automatically created:

- Sales-run. Can run reports in the Sales folder.
- Sales-analyst. Can develop and run reports in private folders within the Sales folder.
- Sales-developer. Can develop, run, and publish reports in a published folder within the Sales folder.
- Sales-dba. Can manage metadata in the Sales folder application directory.
- Sales-sched. Can manage schedules and distribution lists in the Sales folder.
- Sales-admin. Can add a user to or remove a user from the Sales folder group.

Users are added to one or more groups to provide the functionality they require to perform their job. This is done using the Security Center.

Note: The group permissions are additive, not progressive. That is, if you want a user to have schedule and admin rights, you have to add them to both.

24.2 Migrating from Release 1.1.x

Web Query Release 2.1.0 introduces a number of new concepts and terms that are documented in 24.1, "New concepts and terms in Web Query 2.1.0" on page 768. It is important that you read this section very carefully.

Users, group profiles, and content can be migrated from Web Query Release 1.1.x 5733QU2 to Release 2.1.0 5733WQX.

24.2.1 Migrating Web Query users, Group Profile license information, and metadata

Web Query Release 2.1.0 introduces a new security model. A key component of this security is the Group, which is best described as a set of operations that formulate a role. Table 24-3 maps Release 1.1.x authorizations to the new Groups defined in Release 2.1.0.

Table 24-3 Group map

Release 1.1.x Profiles	Release 2.1.0 Groups
MUdomain profile	Folder-run
MDdomain profile	Folder-dev, Folder-dba
MRADMIN	WebQueryAdministrator
MRSCHEDULE	Folder-sched

The post installation exit program migrates both metadata and profile licenses during the product installation:

- When the base product is installed, the metadata is copied to the new directory structure.
- When option 4 Web Query Developer Users is installed, the named user licenses are migrated from QU2 base to WQX option 4.
- When option 6 Web Query Runtime Enablement Groups is installed, the licensed group profiles are migrated from QU2 base to option 6.

For example, when option 4 Web Query Developer Users is installed, the QU2 administrators are migrated as WebQueryAdministrator group users, the QU2 domain developers are migrated as Folder-dba group users, and the QU2 report schedulers are migrated as Folder-sched group users.

24.2.2 Migrating Web Query content

The content includes Domains, Reports, HTML files, Stylesheets, and images. Report Broker content can be migrated as well. This includes schedules and distribution lists.

When content migration is executed at first startup (STRWEBQRY), the Web Query 2.1.0, Reporting Server must be active (that is, successfully started) for the migration to occur. Content migration is a one-time operation on first successful startup after installation. If after the migration, you add or change reports in QU2, those items are not automatically migrated to WQX.

Note: Content migration may take several minutes. A status message is displayed.

In Web Query Release 2.1.0, the term Domain is no longer used. The new terminology is *top level folder*. Domains will map directly to a top level folder. A Domain's folders map to sub folders. Procedures, HTML files, bitmaps, and stylesheets are still relative terms.

Table 24-4 maps Release 1.1.x objects to Release 2.1.0 objects.

Table 24-4 Object map

Release 1.1.x	Release 2.1.0 Object
Domain	Top Level folder
Domain folders	Sub folder

With Release 2.1.0, the best practice for users is to use the new top level folder and app folder association, as opposed to using the baseapp approach. Using this best practice, an app folder with the same name as a newly created top level folder will be created and linked to that top level folder. Creating and managing metadata in the linked app folder is the best practice.

24.3 Other considerations

In this section, we discuss a few other considerations to keep in mind.

Command proxies

During Web Query 2.1.0 installation, the command proxies in QSYS are switched to the new product. So STRWEBQRY, ENDWEBQRY, and the other Web Query commands will be directed to the new release unless they are qualified QWEBQRY77 or QWEBQRY76.

Security model

According to the new security model of Web Query Release 2.1.0, after migration, the top level folder directory, subdirectories, and the metadata files are owned by QWQADMIN. A unique authorization list is created at time of folder creation to secure the folder and its metadata files. Users are added to the authorization list and their authorities correspond to their permission group assignments in the Security Center. QWQADMIN have all access and other users will have only read access. Public will have no access.

Coexistence between 1.1.x and 2.1.0

While WQX base is in 70-day trial, both products can run concurrently and function independently. This allows time to test (convenience and flexibility) before switching production to the new release.   After WQX is keyed, either product can be active, but not both. If QU2 is already active when WQX is started, WQX will fail to start. If WQX is active before QU2, it will use the 11331 port; QU2 can be started, but users will be unable to login. During migration, content is copied from QU2 to WQX, but the QU2 content (reports, schedules, metadata) remains intact.

NLS setting

In Web Query 1.1.x, for non-English clients, users had to configure the client NLS settings for their codepage in the Web Query administrator console. In Web Query 2.1.x, the client is migrated to Unicode for compatibility with any Language setting on the login page.

Items not migrated

After Web Query 2.1.0 is installed, some items are not migrated: configurations, custom settings, connections, additional adapters (Microsoft SQL Server and JD Edward adapters).

25

IBM i Business Intelligence solution

DB2 Web Query for i provides analytical (business intelligence) capabilities for end users. One of the critical success factors of any business intelligence application is delivering that information to those end users (analysts) to meet their requirements related to timing.

Some analysts require real time access to information in a self-service model (no dependency on someone else to run reports for them). Others may find that having reports run in batch mode and delivered to them overnight meets their needs. Still others may view complex analysis jobs as needing to be done as efficiently as possible but not expecting that to be instantaneous.

In other words, setting expectations on performance of the BI environment and implementing a solution that meets those expectations can make or break a BI project.

In some cases, the best way to achieve those performance requirements may be to isolate and optimize the analytics workload from other production workloads.

In this chapter, we discuss these various considerations.

25.1 Operational data stores, data marts, and data warehouses

Separating the analytics workload could involve creation of an isolated, optimized system/database for improving performance while not negatively impacting production systems.

An operational data store (ODS) is a replicated image of your production database. It may not be an exact replica, but is close to it. Some minimal data transformations may alter the data upon movement into the ODS, such as elimination of unimportant fields/columns, or transforming codes into descriptive names.

Data marts and data warehouses take that isolated reporting repository beyond a mirrored image of the database, adding much more data transformations of the data to support analytics. For example, creation of a surrogate key to manage the consolidation of two databases with customer information but non-unique customer IDs is a common data warehouse data transformation.

For IBM i customers, an optimized environment for analytics while leveraging your skills and administrative functions and policies needs serious consideration in meeting the performance expectations of your analysts!

25.2 Introducing the new IBM i for Business Intelligence solution

IBM i for Business Intelligence is a packaged solution that is easy to order and easy to implement, a perfect choice for improving your capability to analyze data -turning it into information that can help transform your business.

You can demonstrate immediate results with Operational Reporting and transition into a robust Data Warehouse, based on your business demands.

This solution combines the strengths of Power Systems, IBM i, DB2 for i, DB2 Web Query for i, and data transportation software to deliver an integrated platform that houses the extracted and transported data sourced from your production systems.

IBM i for Business Intelligence comes with what you need to start-up your reporting environment and it is ready when you are as an expandable, growth platform to extend your data analytic capabilities, transforming sourced data by leveraging additional ETL tooling and services to build a Data Warehouse.

This new solution is offered as three configurations (small, medium, and large) that include software and required licensing, as well as getting started services designed to get you up and running fast. There are no long wait times before you can show new business information to your executives.

25.2.1 DB2 Web Query for i Standard Edition

Included in the IBM i for BI solution is the Standard Edition of DB2 Web Query. This product provides a very robust set of functions to support the ad-hoc, self-service reporting requirements, to the batch oriented automated report execution and distribution.

Include data from any Microsoft SQL Server database into your applications with the included data adapter for SQL Server. Integrate reports into customized web based applications with the ability to invoke DB2 Web Query functions programmatically.

Analytical functions in Standard Edition include mobility support, spreadsheet integration, and onLine analytical processing.

Improving performance and simplify management

IBM has spent over a decade building features into DB2 for i for improved query performance and simplified management, including many functions automated by DB2 for i when processing queries. The DB2 for i SQL Query Engine (SQE) can improve performance over five times from previous technology.

Take advantage of advanced indexing and statistic technologies, along with the ability to create and store aggregate (summary) data for best performance. These technologies are especially important in processing the types of requests typical in a reporting environment where analysts want to view summarized data and then drill into it. The On Demand Performance Center simplifies the ability for an administrator to understand how DB2 is processing queries and provides advice on steps to take to improve runtime performance.

Reducing I/T support efforts

The IBM i for BI solution also includes installation services such that you can get up and running with your ODS in a matter of days.

By leveraging the operational and administrative functions that you already have built around your current IBM i on Power Systems environment allows you to minimize costs and risk, while still providing an appliance-like optimized solution for DB2 Web Query analytics.

Unlike true “appliances,” the multi-core, partition capable systems available in the three-sized versions of the IBM i for BI solution can be expandable for other purposes.

You can implement a combined ODS and disaster recovery system, or take advantage of POWER7 resources to run other applications, including IBM AIX® or Linux operating system based applications. Get the advantages of an appliance without the lock-in of only using it for a single purpose.

26

Adapter for Microsoft SQL Server

The DB2 Web Query Adapter for Microsoft SQL Server extends your reporting environment by incorporating access to data stored in one or more Microsoft SQL Server databases. Current versions supported are 2000, 2005, and 2008.

Although the majority of your data is likely in DB2 on i, access to data stored in Microsoft SQL Server databases may also be key to creating desired reports. By installing DB2 Web Query Standard, you can seamlessly incorporate SQL Server data in your reports in real time without the need of a complex replication process. The Microsoft SQL Server adapter allows your organization to use a single query and report writing tool, DB2 Web Query, to access all of the necessary data for your reports.

In earlier chapters, you learned how you can extend the value of the information stored in DB2 for i by doing online analytical processing, Active Reports, dashboards, and automatic report distribution. With this adapter, you can do the same with information stored in your Microsoft SQL Server databases.

This chapter will take you through the steps necessary to establish a connection to your SQL Servers. But, before we get started, you must install the option.

26.1 Installation

To begin, you must install Web Query Standard Edition. Also make sure that you have the latest PTF group installed for DB2 Web Query.

Due to Microsoft's licensing agreements, we are unable to ship required code as part of the Web Query package. Therefore, to complete the installation, you must install a version of Microsoft's JDBC driver for SQL Server on your system.

To determine which version to download, we briefly discuss versions. Microsoft currently offers four levels of its JDBC driver. Version 1.1 is designed for SQL Server 2000, Version 1.2 for SQL Server 2005, and Version 2.0 for SQL Server 2008. At the time of writing, Web Query only supports the 1.1, 1.2, and 2.0 versions of the JDBC driver.

When configuring Web Query, we can only configure it for one level of the driver, either 1.1, 1.2, or 2.0. If you have only SQL Server 2000 servers, we recommend using Version 1.1. If you have only SQL Server 2005 servers, you must use Version 1.2. If you have SQL Server 2008, you need to use Version 2.0. Version 2.0 is also backwards compatible to SQL Server 2000 and SQL Server 2005. Therefore, if you are in doubt as to which version of driver to choose, we recommend the 2.0 version, as that will work with all levels of SQL Server.

As we are discussing configuration settings, it is important to note that the settings are tied to the driver level and not the level of SQL Server. Let us assume that you are using the 1.2 version to connect to both SQL Server 2000 and 2005 servers. Even when you are defining the connection to the SQL Server 2000 server, you follow the settings for the 2005 server since those correspond to the 1.2 version of the driver that you are using.

So now that you know which version of the driver you want to download, use your favorite search engine or `Microsoft.com` to find links to the 1.1, 1.2, or 2.0 version of the JDBC driver. Searching for “Microsoft SQL Server JDBC Driver” usually returns the correct page at the top of the results. If prompted, download the UNIX version of the driver, not the Windows version.

This will download a tar/gz package containing the required Java Archives (JAR) files. Use your favorite decompression utility, such as the open source 7-Zip, to extract the required files. For Version 1.1 of the driver, the files are `msbase.jar`, `mssqlserver.jar`, and `msutil.jar`. For Version 1.2 of the driver, there is just one: `sqljdbc.jar`. For Version 2.0 of the driver, the files are `sqljdbc.jar` and `sqljdbc4.jar`.

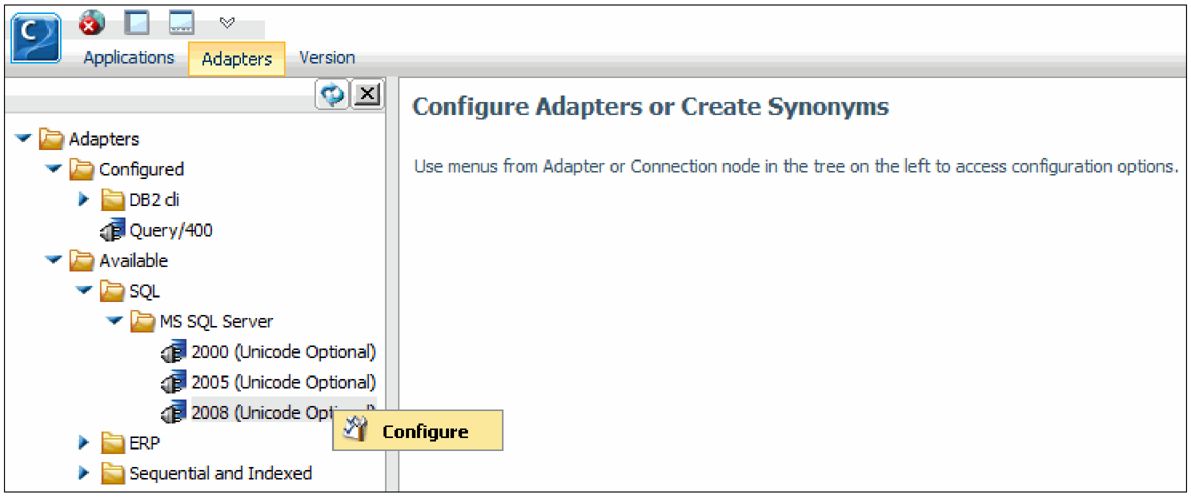
Once you have the JAR files, FTP them in binary mode or use a mapped drive to put them onto the system. The JAR files must be placed into the Java Extensions directory located at `/QIBM/UserData/Java400/ext`. If you choose version 2.0, only copy the sqljdbc4.jar to the IFS. By placing them in this directory, we do not have to worry about setting the proper classpath. This directory is always included.

26.2 Establishing a connection to your Microsoft SQL Server

To configure the adapter within DB2 Web Query, you must provide both connection and authentication information. In this example we show how to configure the SQL adapter for Microsoft SQL Server 2008 using the JDBC driver Version 2.0 (`sqljdbc4.jar`) To configure the adapter from DB2 Web Query, follow these steps:

1. Log on to Web Query as a developer or an administrator.
2. Right-click the **Century Electronics** top folder, choose **Metadata**, and then select **New**.

3. A new pop-up window will appear. From this window, expand **Available**, expand **SQL**, and the expand **MS SQL Server**. You will see three possible MS SQL Server versions (2000 for Version 1.1 of the driver, 2005 for Version 1.2 of the driver, or 2008 for Version 2.0) represented. Again, choose the version that matches the level of driver that you installed, which may not be the same as the level of SQL Server to which you are connecting. For this example, right-click **2008 (Unicode Optional)** and click **Configure**. See Figure 26-1.

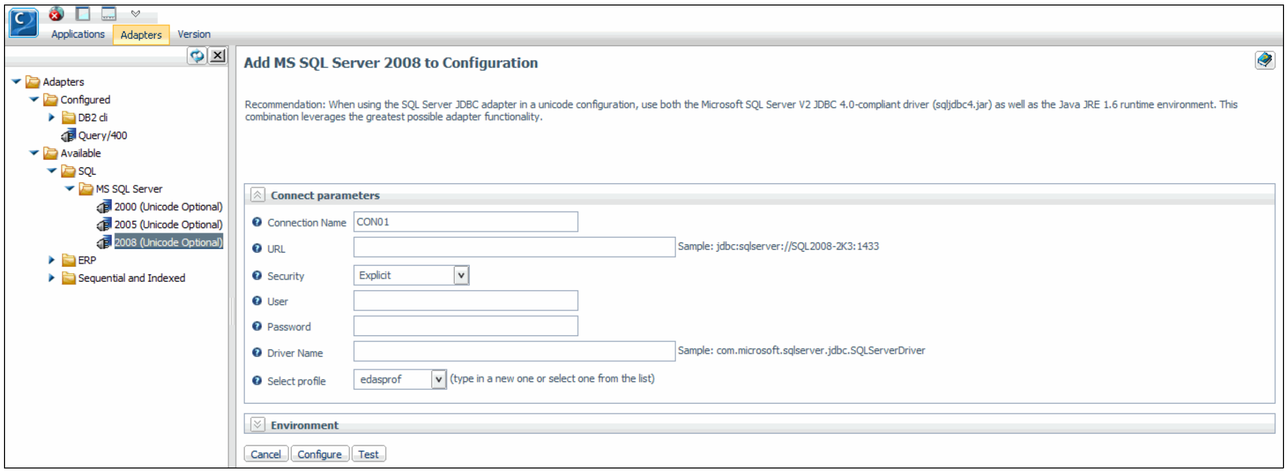


Configure Adapters or Create Synonyms

Use menus from Adapter or Connection node in the tree on the left to access configuration options.

Figure 26-1   Configure New SQL Adapter

4. The Add MS SQL Server 2008 to Configuration panel is presented. See Figure 26-2.



Add MS SQL Server 2008 to Configuration

Recommendation: When using the SQL Server JDBC adapter in a unicode configuration, use both the Microsoft SQL Server V2 JDBC 4.0-compliant driver (sqljdbc4.jar) as well as the Java JRE 1.6 runtime environment. This combination leverages the greatest possible adapter functionality.

Connect parameters

Connection Name	CON01	
URL		Sample: jdbc:sqlserver://SQL2008-2K3:1433
Security	Explicit	
User		
Password		
Driver Name		Sample: com.microsoft.sqlserver.jdbc.SQLServerDriver
Select profile	edasprof	(type in a new one or select one from the list)

Environment

Figure 26-2   Add MS SQL Server Configuration

5. In the configuration panel, name your adapter as desired. In this example, the connection name is called SQLTEST. This can be any name that you want.

6. Enter the URL location for your SQL Server data source. This consists of the driver prefix, the separator (://), followed by the host name (location) of the SQL Server, the separator (:), and the port, in the format as follows:

   ```
   prefix://hostname:port
   ```

If you are using the 1.1 version of the driver (2000), the prefix is jdbc:microsoft:sqlserver. If you are using the 1.2 version (2005), the prefix is jdbc:sqlserver. If you are using the 2.0 version (2008), the prefix is jdbc:sqlserver. The host name for your system can be either the DNS name (sqlserver.mycompany.com) or dotted IP address (1.2.3.4). If you use a DNS name, make sure that IBM i can resolve it by doing a simple PING from the command line to the SQL Server's DNS name.

The final piece of this URL is the port number. The default port for the SQL Server is 1433. However, not all SQL Servers run on that port. To find your SQL Server's port number, proceed as follows. First open up the Windows Task Manager. Add the PID (Process ID) column by clicking **View** → **Select Columns** → **Check the PID option** → **OK**. Find the image name `sqlservr.exe` and record its PID. Then open a command prompt and run **`netstat -a -n -o`** and look for the matching PID number. Under the Local Address column will be 0.0.0.0:xyz. The value xyz after the colon is the port number.

Assume that the DNS name of our SQL Server is sqltest.rchland.ibm.com, the port is 1433, and the version of the driver is 2.0. The URL would then be:

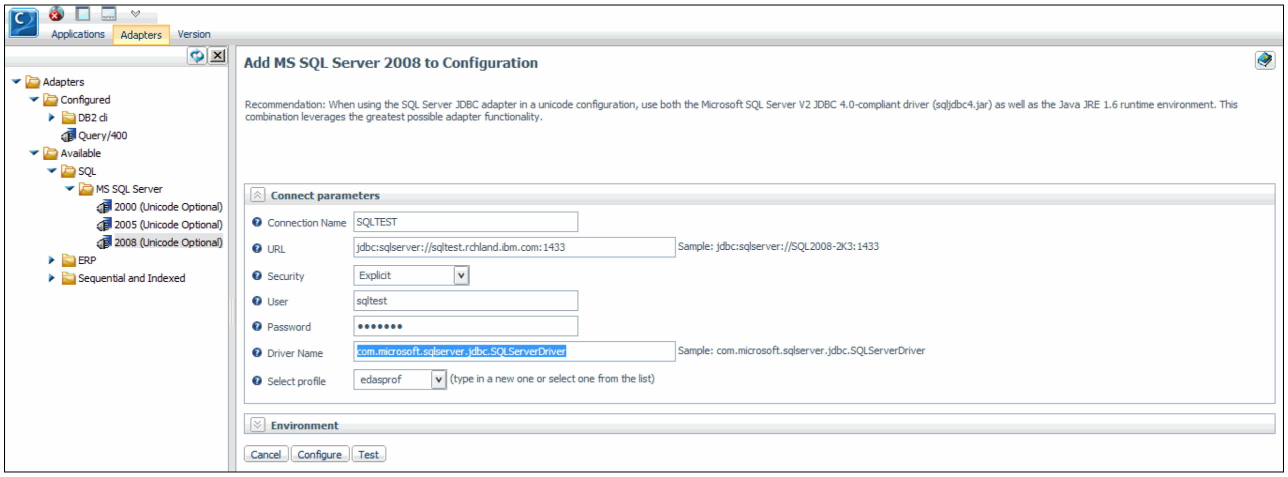
```
jdbc:sqlserver://sqltest.rchland.ibm.com:1433
```

If these were the same but using the 1.1 version of the driver, the URL would then be:

```
jdbc:microsoft:sqlserver://sqltest.rchland.ibm.com:1443
```

7. Select for Security the option **Explicit** and provide an SQL Server user and password necessary for authentication. All connections to the SQL Server will use this profile. Therefore, it is important that this profile is granted access to all tables to be accessed in the SQL Server. In this example, a user of sqltest and corresponding password was provided. This will be unique to your location.
8. The final step is to enter the driver name. This tool is varied with the version of the driver. If you are using the 1.1 driver, enter `com.microsoft.jdbc.sqlserver.SQLServerDriver` for the driver name. If you are using version 1.2 of the driver, enter `com.microsoft.sqlserver.jdbc.SQLServerDriver` for the driver name. If you are using version 2.0 of the driver use `com.microsoft.sqlserver.jdbc.SQLServerDriver`
9. Do not be concerned that the CLASSPATH is NOT SET, provided that you posted the JDBC driver JAR file in the above directory location. Web Query will find the necessary driver.

In our example, the final configuration panel is as shown in Figure 26-3.



Add MS SQL Server 2008 to Configuration

Recommendation: When using the SQL Server JDBC adapter in a unicode configuration, use both the Microsoft SQL Server V2 JDBC 4.0-compliant driver (sqljdbc4.jar) as well as the Java JRE 1.6 runtime environment. This combination leverages the greatest possible adapter functionality.

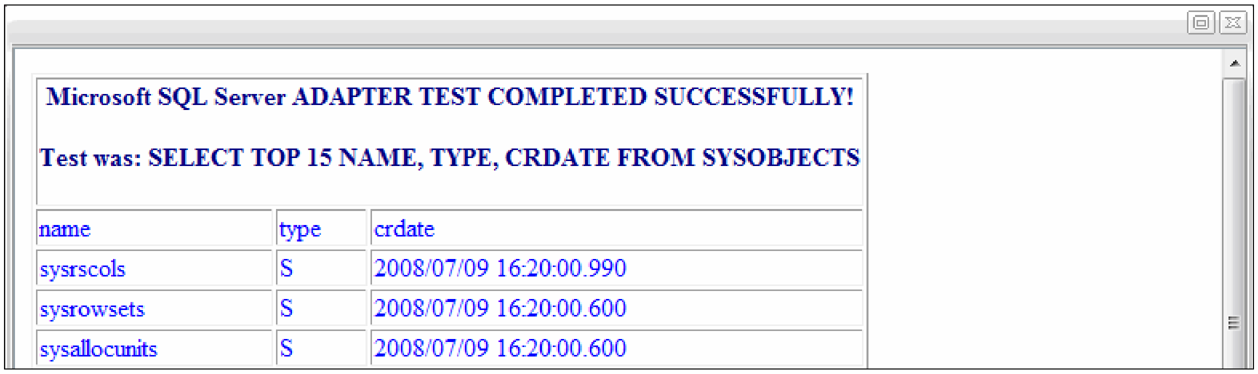
Connect parameters

Connection Name: SQLTEST
URL: jdbc:sqlserver://sqltest.rchland.ibm.com:1433 Sample: jdbc:sqlserver://SQL2008-2K3:1433
Security: Explicit
User: sqltest
Password: •••••••
Driver Name: com.microsoft.sqlserver.jdbc.SQLServerDriver Sample: com.microsoft.sqlserver.jdbc.SQLServerDriver
Select profile: edasprof (type in a new one or select one from the list)

Environment

Figure 26-3 Add MS SQL Server

10. If you want to make sure that your configuration is correct, click the **test** button. If your configuration is valid, a new web page will open containing a result set from your SQL Server as shown in Figure 26-4.



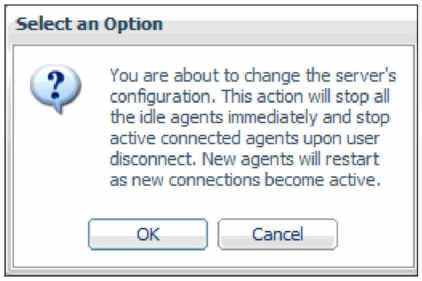
Microsoft SQL Server ADAPTER TEST COMPLETED SUCCESSFULLY!

Test was: SELECT TOP 15 NAME, TYPE, CRDATE FROM SYSOBJECTS

name	type	crdate
sysrscols	S	2008/07/09 16:20:00.990
sysrowsets	S	2008/07/09 16:20:00.600
sysallocunits	S	2008/07/09 16:20:00.600

Figure 26-4 Validating SQL Settings

11. Click **Configure** once the appropriate information is provided.You will receive a message stating that you about to change Server's Configuration (Figure 26-5). Click **OK**.



Select an Option

You are about to change the server's configuration. This action will stop all the idle agents immediately and stop active connected agents upon user disconnect. New agents will restart as new connections become active.

Figure 26-5 Message Server Configuration

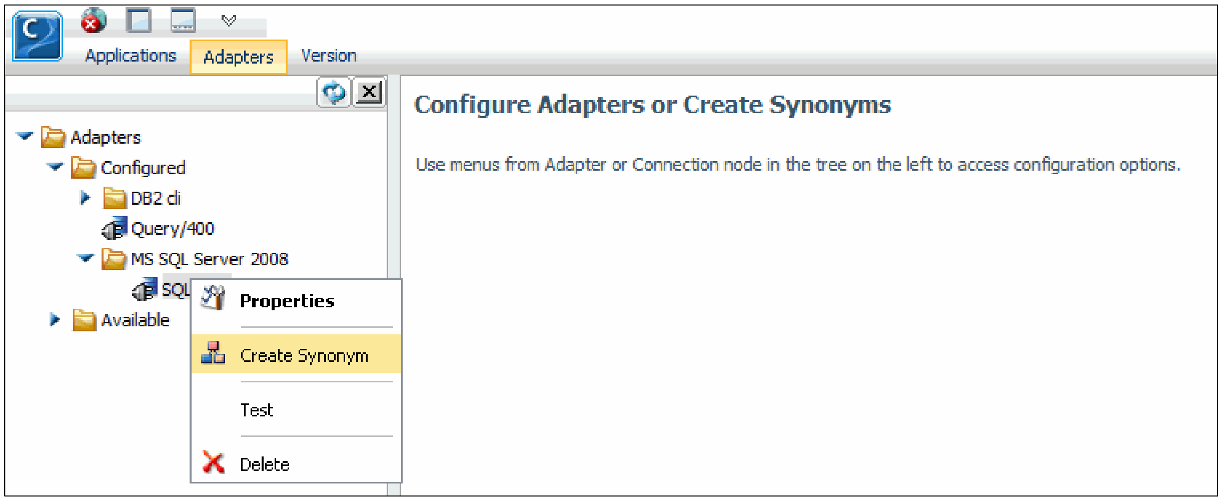
12. Once you have configured the new adapter, you should restart the DB2 Web Query servers before moving on to create metadata. Use the ENDWEBQRY and STRWEBQRY commands.

26.3 Creating metadata

Just as described in Chapter 3, "Defining metadata" on page 21, once you have established a connection to a data source, you can create various synonyms over the data (tables, views, stored procedures) at the data source that you want to query.

1. Again, right-click the **Century Electronics** top folder, choose **Metadata**, then select **New**.

2. In the Adapter pane, you should now see your MS SQL Server as a data source. Right-click the newly created connection and select **Create Synonym,** as shown in Figure 26-6.

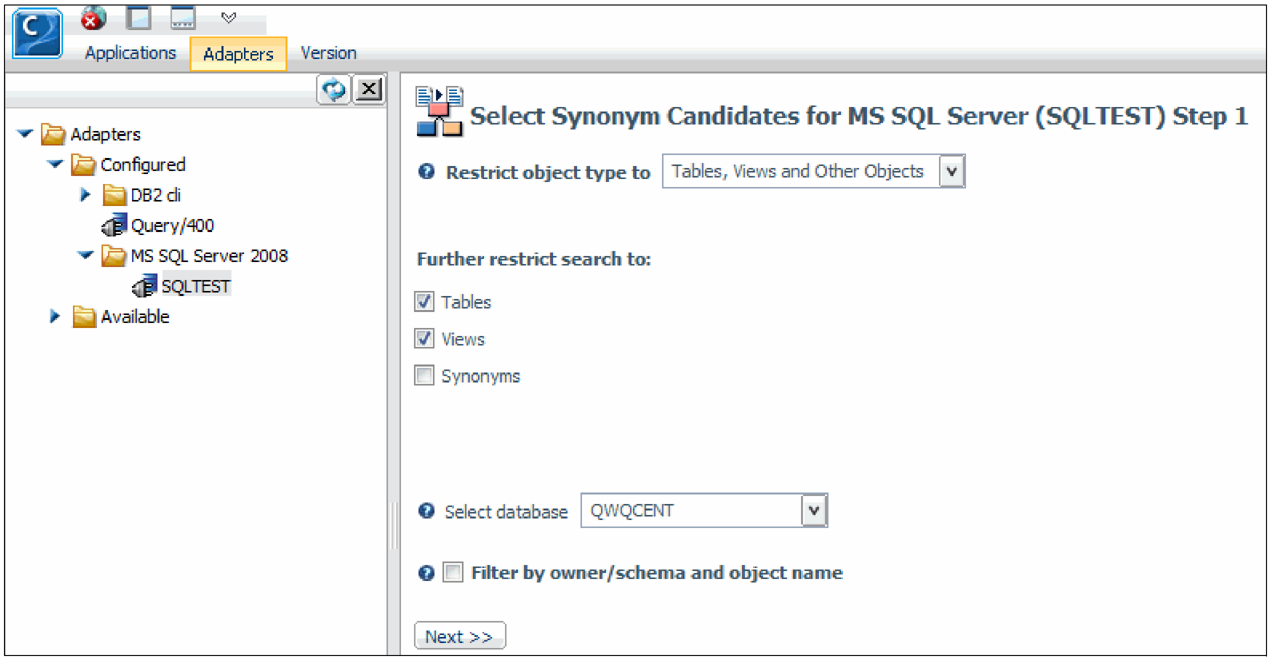


Configure Adapters or Create Synonyms

Use menus from Adapter or Connection node in the tree on the left to access configuration options.

Figure 26-6 Creating synonyms for SQL Server

3. Defining this synonym is a two-step process. You will first see a panel similar to Figure 26-7.



Select Synonym Candidates for MS SQL Server (SQLTEST) Step 1

Restrict object type to: Tables, Views and Other Objects

Further restrict search to:

- [✓] Tables
- [✓] Views
- [ ] Synonyms

Select database: QWQCENT

[ ] Filter by owner/schema and object name

Figure 26-7 Identify the target database

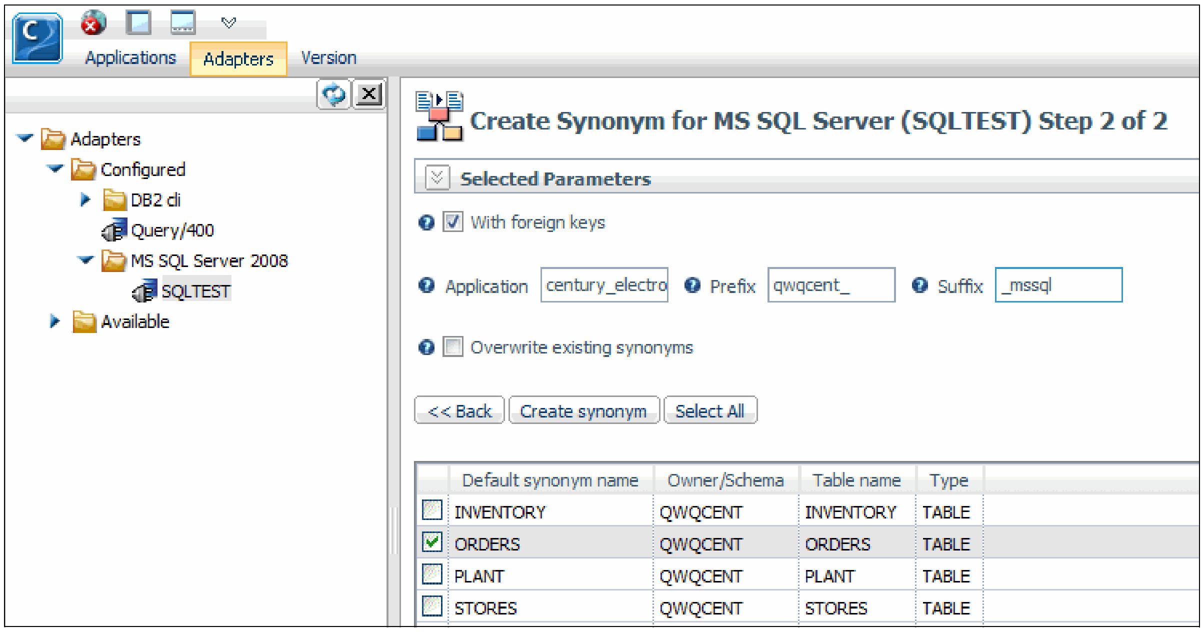
First identify the target database on the SQL Server. Remember, SQL Server is different from IBM i in that is has a concept of multiple databases. Within each of those databases are the schemas that contain the tables. In this example, we have imported the QWQCENT sample database so that it appears in the Select database list. In this example, only table objects are requested from the selected database.

4. Click **Next** to advance to the second step.

5. In the second step, select the tables that you are interested in querying.

   In this example, the selection is the Orders table and is specifying the **With foreign keys** option.

   Provide the prefix or suffix if desired and click **Create Synonym** once you have selected the tables of interest. As a best practice, we recommend using the schema name followed by an underscore as the prefix and a suffix such as _mssql. This way, your Order table appears as `qwqcent_Orders_mssql`. This groups all items from the HumanResources schema together and provides you with a reminder that this data source is from a SQL Server, as shown in Figure 26-8.



Create Synonym for MS SQL Server (SQLTEST) Step 2 of 2

Selected Parameters

[✓] With foreign keys

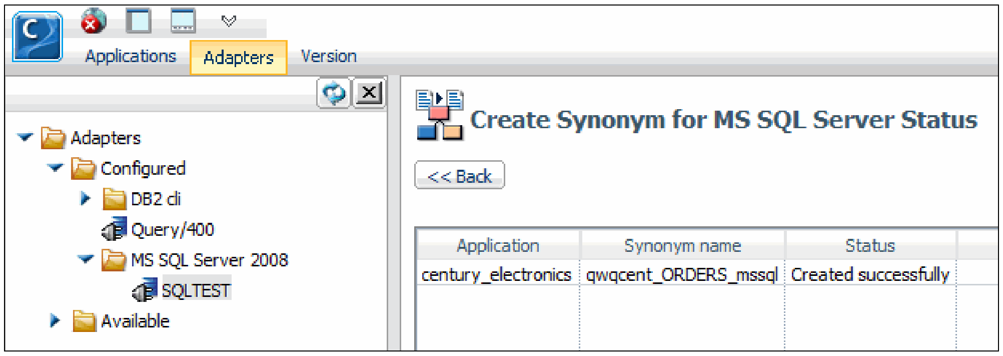
Application: century_electro Prefix: qwqcent_ Suffix: _mssql

[ ] Overwrite existing synonyms

	Default synonym name	Owner/Schema	Table name	Type
☐	INVENTORY	QWQCENT	INVENTORY	TABLE
✓	ORDERS	QWQCENT	ORDERS	TABLE
☐	PLANT	QWQCENT	PLANT	TABLE
☐	STORES	QWQCENT	STORES	TABLE

Figure 26-8 Selected table from the imported qwqcent into SQL

6. Click **Create Synonym**.
7. You will see a summary completion panel similar to Figure 26-9. Close that window.



Create Synonym for MS SQL Server Status

Application	Synonym name	Status
century_electronics	qwqcent_ORDERS_mssql	Created successfully

Figure 26-9 Create Synonym Completion Message

8. You now have a synonym that is specific to tables located in a database on an MS SQL Server. You can now use this new synonym to build reports using Info Assist, or Developer Workbench, exactly the same way that you do with local data synonyms.

26.4 Additional SQL Server adapter information

For additional information about the DB2 Web Query SQL Server adapter, a documentation PDF file is located under the documents link on the DB2 Web Query Getting Started page, located at this website:

`http://www.ibm.com/systems/i/software/db2/webquery/gettingstarted.html`

Here is some of the information available in this document:

- How to customize your SQL Server environment
- How to call SQL Server stored procedures
- Information about SQL Server data types

Part 4

Appendixes

In this section, we have included the following appendixes for your convenience:

- Appendix A, “Date and time functionality” on page 789
- Appendix B, “DB2 Web Query system variables” on page 839
- Appendix C, “Change management considerations” on page 841

A

Date and time functionality

Date and time components are very critical in querying and reporting. DB2 Web Query provides a variety of ways to provide these key elements in your reporting environment. This appendix contains many of the details needed for including date and time elements in your reports and charts.

Date and time system variables

Table A-1 lists the date and time variables that are available in DB2 Web Query. These variables can be specified in the headings and footers at the report and page level, as well as used in Define fields.

Table A-1 Date and time variables

System variable	Description	Format or value	Example (for October 07, 2012)
&DATE	Returns the current date	MM/DD/YY	10/07/2012
&DMY	Returns the current date	DDMMYY	071012
&DMYY	Returns the current (four-digit year) date	DDMMCCYY	07102012
&MDY	Returns the current date. Useful for numerical comparisons	MMDDYY	100712
&MDYY	Returns the current (four-digit year) date	MMDDCCYY	10072012
&TOD	Returns the current time that the query was executed	HH.MM.SS	15.50.07
&YMD	Returns the current date	YYMMDD	121007
&YYMD	Returns the current (four-digit year) date	CCYYMMDD	20121007
&DATEWtr	Returns the full name of the day of the week	Name of day of week	Wednesday
&DATEMtrDYY	Returns the name of the month followed by the day and the four digit year	Name of month DD, YYYY	October 7, 2012
&DATEWtr, &DATEMtrDYY	Returns the full name of the day of the week, followed by the name of the month, followed by the day and the four-digit year	Name of day of week, Name of month DD, YYYY	Wednesday, October 7, 2012

Date format

The various date formats enable you to define a field as a date and work with it as a date. Using the date format, you can perform the following tasks:

- Define date components, such as year, quarter, month, day, and day of week, and extract them easily from the date fields.
- Sort reports into date sequence, regardless of how the date appears. For example, January sorts before April even though, without date smarts, April alphabetically comes before January.
- Do arithmetic with dates and compare the dates without resorting to special date-handling functions.

Date format display options

The date format does not specify type or length. Instead, it specifies date component options (D, W, M, Q, Y, and YY) and display options. These options are shown in Table A-2.

Note: Use of these format options may result in queries that do not translate the date formatting to SQL. This can result in a query that does not perform optimally. If you experience this, you should explore other techniques to convert your date formats. These alternative methods are discussed later in this chapter.

Table A-2 Date format options

Display option	Meaning	Effect
D	Day	Displays a value from 1 to 31 for the day.
M	Month	Displays a value from 1 to 12 for the month.
Y	Year	Displays a 2-digit year.
YY	Four-digit year	Displays a 2-digit year.
T	Translate month	When used with M in a date (MT or TM), the 3-letter abbreviation for the month in uppercase is displayed.
t	Translate month	When used with M in a date (Mt or tM), the 3-letter abbreviation for the month is displayed, capitalizing only the first letter of the month or day.
TR	Translate month or day	TR is like T, but displays the full name in uppercase.
tr	Translate month or day	tr is like t, but displays the full name in mixed case.
Q	Quarter	Displays the quarter Q1–Q4.
W	Day of week	On its own, W displays the number of the day of the week (1–7, Mon=1). Used in combination with other date options, W displays a 3-letter abbreviation of the day of the week in uppercase.
w	Day of week	Functions as uppercase W (described previously), except that the first letter is uppercase and the following letters are lowercase.
WR	Day of week	Functions the same as uppercase W (described above), except that the entire day name is displayed instead of an abbreviation.
wr	Day of week	Functions the same as lowercase w (described above), except that the entire day name is displayed instead of an abbreviation.
JUL	Julian format	Displays date in Julian format.
YYJUL	Julian format	Displays a Julian format date in the format YYYYDDD. The 7-digit format displays the 4-digit year and the number of days counting from January 1. For example, January 3, 2001 in Julian format is 2001003.

Table A-3 shows samples of output for various date formatting options.

Table A-3 Sample output for date formatting options

Translation	Display
MT	JAN
Mt	Jan
MTR	JANUARY
Mtr	January
WR	MONDAY
wr	Monday
Q	Q1
YQ	07Q1

Controlling the date separator

You can control the date separators when the date is displayed. In basic date format, such as YMD and MDYY, the date components are displayed separated by a slash character (/). The same is true for the year-month format. The year-quarter format is displayed with the year and quarter separated by a blank (for example, 12 Q3 or Q3 2012). The single component formats display just the single number or name.

The separating character can be changed to a period, a dash, or a blank, or can be eliminated entirely. Table A-4 shows the FORMAT specifications that can be used to change the separating character.

Table A-4 Date separators

Format	Display
YMD	12/09/22
Y.M.D	12.09.22
Y-M	12-09
YBMBD	12 09 22 (The letter B signifies blank spaces.)
Y|M|D	120922 (The concatenation symbol | eliminates the separation character.)

Using date fields

Table A-5 shows valid examples of specifying dates.

Table A-5 Examples of specifying the dates

Situation	Natural date literal
In WHERE screening	WHERE MYDATE IS 'SEP 22 2012'
In arithmetic expressions	MYDATE - '2012 SEP 22'
In computational date comparisons	IF MYDATE GT '22 SEP 2012'

Date fields in arithmetic expressions

The general rule for manipulating date fields in arithmetic expressions is that date fields in the same expression must specify the same date components. The date components can be specified in any order and display options are ignored. Valid date components are Y or YY, Q, M, W, and D. For example, NEWQUARTER and THISQUARTER both have FORMAT specifications of Q and the value of THISQUARTER is 2. In this case, consider the following statement:

```
NEWQUARTER = THISQUARTER + 3
```

This statement gives NEWQUARTER a value of 1 (that is, the remainder of 5 divided by 4).

The following example calculates the number of days elapsed since January 1, 1999:

```
YEARTODATE = ORDERDATE - 'JAN 1 1999' ;
```

Converting date fields

At this point, you may be wondering: “Can DB2 Web Query handle my legacy date fields?”

That is a good question. Dates and times are vital business dimensions that are required in most business reports. After all, how often do you create a report against data in your database without filtering, sorting, or aggregating the data against a date-related or time-related field? For most reports, the date and time dimension is an integral piece of information.

When it comes to storing date values in their database files, many IBM i shops use legacy date data types. Legacy dates are typically defined as numeric or alphanumeric fields that contain numbers or character strings that represent the date. An example of this is a field defined as zoned decimal (8,0), which contains the value 06152012 to represent the date June15, 2012. However, to DB2 Web Query, this field is nothing more than a decimal field. However, to DB2 Web Query, this field is nothing more than a decimal field and it is not recognized as a date data.

But getting back to the original question, the answer is that, yes, DB2 Web Query can handle your legacy date fields. This section describes how to implement two different techniques for converting legacy date formats to fields that DB2 Web Query recognizes as true date fields (also referred to as *smart dates*).

When the tool recognizes fields as dates, it can provide additional reporting features:

- Advanced date and time manipulation, calculations, and analysis
- Report selection parameters that can be specified by invoking JavaScript calendar widgets for a more user-friendly experience
- Date decomposition to break the date into separate fields that represent the year, quarter, month, and day

All of these capabilities enable the report developer to deliver a report that is easy to use and to provide the report formatting and information that is required.

This section focuses on date conversions. Two types of conversions are possible:

- Format conversion
- Date component conversion

In *format conversion*, the value of a date format field can be assigned to an alphanumeric or integer field that uses date display options. The reverse conversion is also possible.

In *date component conversion*, a field whose format specifies one set of date components can be assigned to another field by specifying different date components. For example, the value of REPORTDATE (DMY) can be assigned to SALESDATE (Y). In this case the year is extracted from REPORTDATE. If REPORTDATE is Apr 27 99, SALESDATE is 99.

Changing the usage

Before you begin with any actual date conversion efforts, you first must determine whether date conversions in your company's DB2 Web Query environment are even necessary. You may be ahead of the curve and have already implemented true date and time fields in your DB2 for i production database. If this is indeed the case, and you simply want to display the dates in your reports differently from how they are stored in the database, all that may be necessary is changing the Usage attribute in the file's DB2 Web Query synonym. For example, perhaps you are storing the date columns in YYMD format and simply want those dates to appear in DMYY format on your reports.

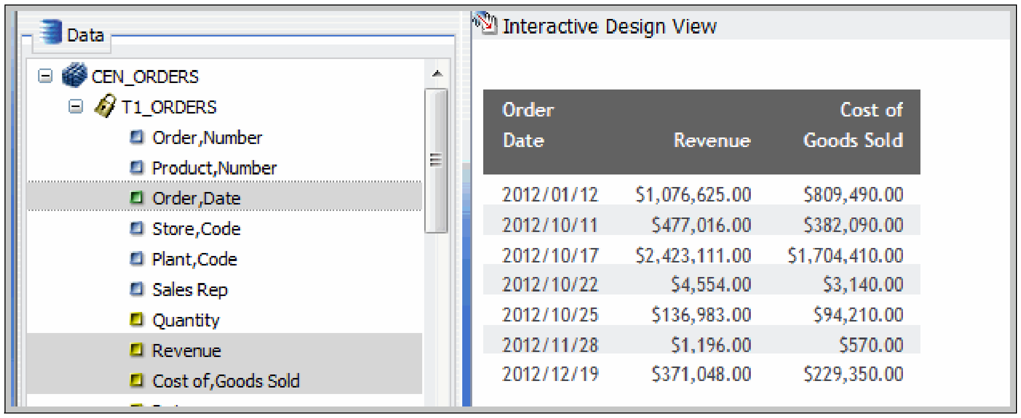
Note: YYMD is the *short name* used to refer to the actual YYYY/MM/DD format. Similarly, MDYY refers to the full MM/DD/YYYY format.

In this exercise, you will convert the order date field from the YYMD format (as it is stored in the database) so that it is displayed in MDYY format on a DB2 Web Query report:

1. From your browser session, log into DB2 Web Query as a developer.
2. From your Century Electronics folder, create a new report.
3. From the list of synonyms, select **cen_orders** as the report’s data source.

4. In the Info Assist tool, from list of available fields, select **ORDERDATE, LINETOTAL,** and **COSTOFGOODSOLD** and drag them into the Interactive Design View pane.

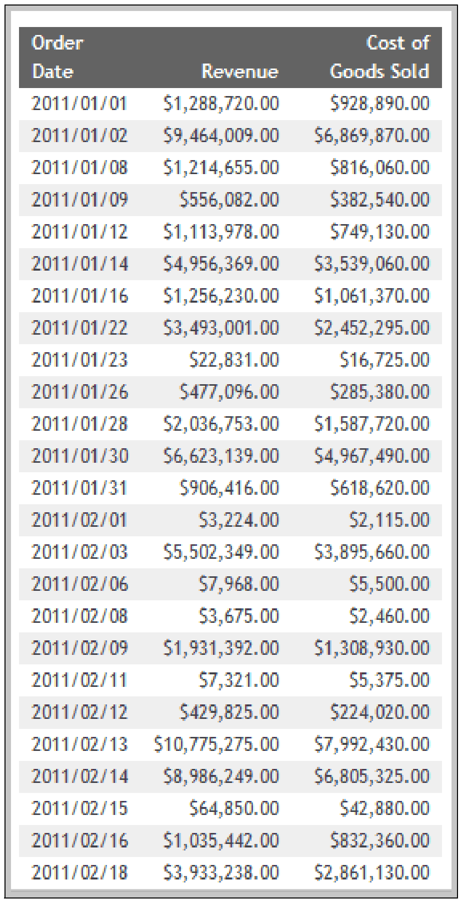
   When you have finished, the report should look like the example provided in Figure A-1.



Order Date	Revenue	Cost of Goods Sold
2012/01/12	$1,076,625.00	$809,490.00
2012/10/11	$477,016.00	$382,090.00
2012/10/17	$2,423,111.00	$1,704,410.00
2012/10/22	$4,554.00	$3,140.00
2012/10/25	$136,983.00	$94,210.00
2012/11/28	$1,196.00	$570.00
2012/12/19	$371,048.00	$229,350.00

Figure A-1 Revenue by Order Date report

5. Save the report as `Revenue by Order Date`.
6. Run the report. In this report (Figure A-2) you see revenue and cost of goods sold ordered and grouped by order date.



Order Date	Revenue	Cost of Goods Sold
2011/01/01	$1,288,720.00	$928,890.00
2011/01/02	$9,464,009.00	$6,869,870.00
2011/01/08	$1,214,655.00	$816,060.00
2011/01/09	$556,082.00	$382,540.00
2011/01/12	$1,113,978.00	$749,130.00
2011/01/14	$4,956,369.00	$3,539,060.00
2011/01/16	$1,256,230.00	$1,061,370.00
2011/01/22	$3,493,001.00	$2,452,295.00
2011/01/23	$22,831.00	$16,725.00
2011/01/26	$477,096.00	$285,380.00
2011/01/28	$2,036,753.00	$1,587,720.00
2011/01/30	$6,623,139.00	$4,967,490.00
2011/01/31	$906,416.00	$618,620.00
2011/02/01	$3,224.00	$2,115.00
2011/02/03	$5,502,349.00	$3,895,660.00
2011/02/06	$7,968.00	$5,500.00
2011/02/08	$3,675.00	$2,460.00
2011/02/09	$1,931,392.00	$1,308,930.00
2011/02/11	$7,321.00	$5,375.00
2011/02/12	$429,825.00	$224,020.00
2011/02/13	$10,775,275.00	$7,992,430.00
2011/02/14	$8,986,249.00	$6,805,325.00
2011/02/15	$64,850.00	$42,880.00
2011/02/16	$1,035,442.00	$832,360.00
2011/02/18	$3,933,238.00	$2,861,130.00

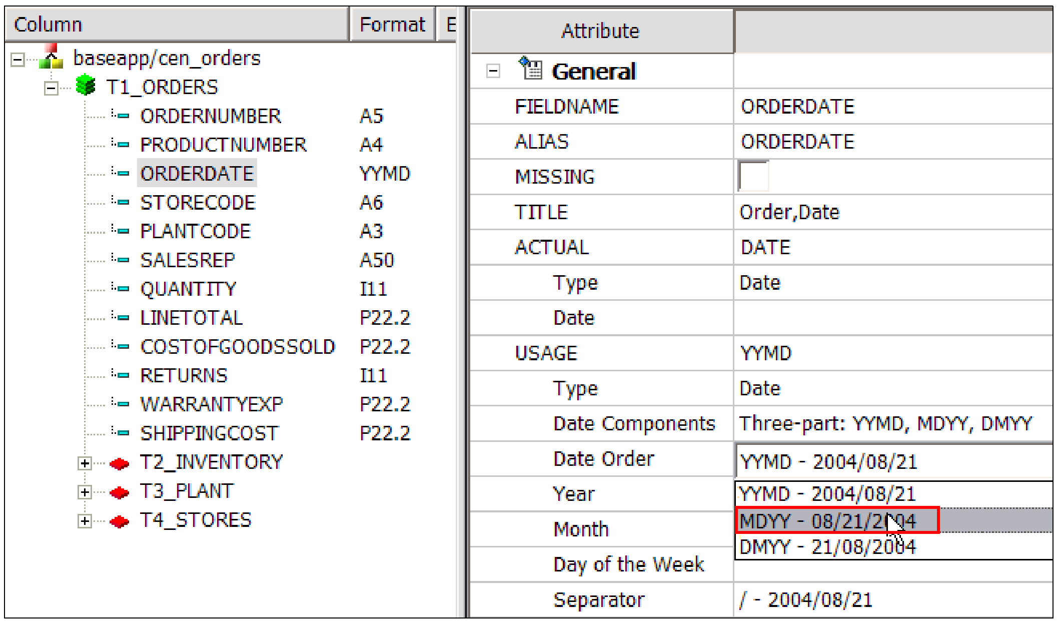
Figure A-2 Results of Revenue by Order Date report

This is a useful report, but suppose that your users really want the order date shown in MDYY format. Because the order date field in your database is defined as YYMD format, you must take additional steps to override this attribute when displaying the date field in a DB2 Web Query report. These steps involve changing the usage value of the order date field as defined in the synonym of the file this report is based on.

To update the synonym you will use DB2 Web Query Developer Workbench:

7. Open DB2 Web Query Developer Workbench and open a connection to your environment.
8. Expand **Data Servers** → **EDASERVE** → **Applications** → **baseapp**.
9. Find and right-click the **cen_orders.mas** synonym file and select **Edit in Synonym Editor** from right-click menu.

   The synonym editor is opened.
10. Select the **ORDERDATE** field. Notice that this is a *true* DATE field (see the ACTUAL attribute) and its USAGE value is YYMD (which is how it is defined in the database). As shown in Figure A-3, override this attribute by selecting **MDYY** from the Date Order setting under USAGE.



Column	Format
baseapp/cen_orders	
T1_ORDERS	
ORDERNUMBER	A5
PRODUCTNUMBER	A4
ORDERDATE	YYMD
STORECODE	A6
PLANTCODE	A3
SALESREP	A50
QUANTITY	I11
LINETOTAL	P22.2
COSTOFGOODSSOLD	P22.2
RETURNS	I11
WARRANTYEXP	P22.2
SHIPPINGCOST	P22.2
T2_INVENTORY	
T3_PLANT	
T4_STORES	

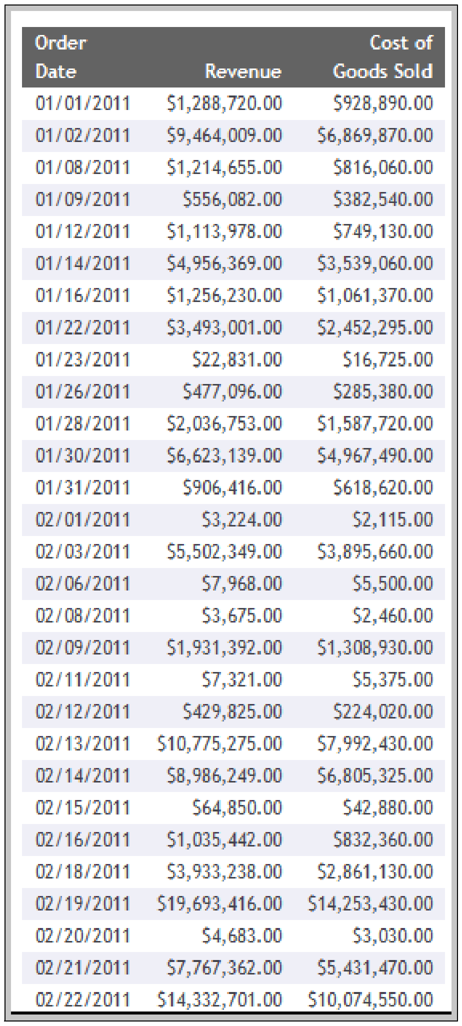
Attribute	
General	
FIELDNAME	ORDERDATE
ALIAS	ORDERDATE
MISSING	
TITLE	Order,Date
ACTUAL	DATE
Type	Date
Date	
USAGE	YYMD
Type	Date
Date Components	Three-part: YYMD, MDYY, DMYY
Date Order	YYMD - 2004/08/21
Year	YYMD - 2004/08/21
Month	MDYY - 08/21/2004
Day of the Week	DMYY - 21/08/2004
Separator	/ - 2004/08/21

Figure A-3 Change date order

11. The USAGE field changes to MDYY. Click the **Save** icon to save this change.

12. Return to your DB2 Web Query browser session and click the **Revenue by Order Date** report again to run it.

Notice that the Order Date column is now presented in MDYY format (Figure A-4).



Order Date	Revenue	Cost of Goods Sold
01/01/2011	$1,288,720.00	$928,890.00
01/02/2011	$9,464,009.00	$6,869,870.00
01/08/2011	$1,214,655.00	$816,060.00
01/09/2011	$556,082.00	$382,540.00
01/12/2011	$1,113,978.00	$749,130.00
01/14/2011	$4,956,369.00	$3,539,060.00
01/16/2011	$1,256,230.00	$1,061,370.00
01/22/2011	$3,493,001.00	$2,452,295.00
01/23/2011	$22,831.00	$16,725.00
01/26/2011	$477,096.00	$285,380.00
01/28/2011	$2,036,753.00	$1,587,720.00
01/30/2011	$6,623,139.00	$4,967,490.00
01/31/2011	$906,416.00	$618,620.00
02/01/2011	$3,224.00	$2,115.00
02/03/2011	$5,502,349.00	$3,895,660.00
02/06/2011	$7,968.00	$5,500.00
02/08/2011	$3,675.00	$2,460.00
02/09/2011	$1,931,392.00	$1,308,930.00
02/11/2011	$7,321.00	$5,375.00
02/12/2011	$429,825.00	$224,020.00
02/13/2011	$10,775,275.00	$7,992,430.00
02/14/2011	$8,986,249.00	$6,805,325.00
02/15/2011	$64,850.00	$42,880.00
02/16/2011	$1,035,442.00	$832,360.00
02/18/2011	$3,933,238.00	$2,861,130.00
02/19/2011	$19,693,416.00	$14,253,430.00
02/20/2011	$4,683.00	$3,030.00
02/21/2011	$7,767,362.00	$5,431,470.00
02/22/2011	$14,332,701.00	$10,074,550.00

Figure A-4 Results of revenue by order date after changing usage

Using DB2 Web Query functions to convert to dates

Changing the usage field may be enough if your date fields are defined as true date fields in your database and all you want to do is change the month-day-year order of how the date is displayed. However, as mentioned earlier, if you are like many IBM i shops, you probably have date fields defined as a data type other than date or time stamp (because these data types were not supported by the RPG compiler until V3R1 of the operating system). Consequently, you may have date fields defined as packed decimal (8,0), which contain the value 04102008 to represent the date April 10, 2008.

In this exercise, the file used has packed decimal (8,0) date fields with this MDYY format. You use the DB2 Web Query built-in functions (BIFs) to convert these fields to virtual fields that are defined as a true date. This true date field is used as the basis for other virtual columns in other formats such as year and day of the week.

Attention: You could also create these virtual fields in the synonym (rather than in the report). In fact, this is the recommended approach, but for the purpose of this exercise, the conversions are performed in the report definition.

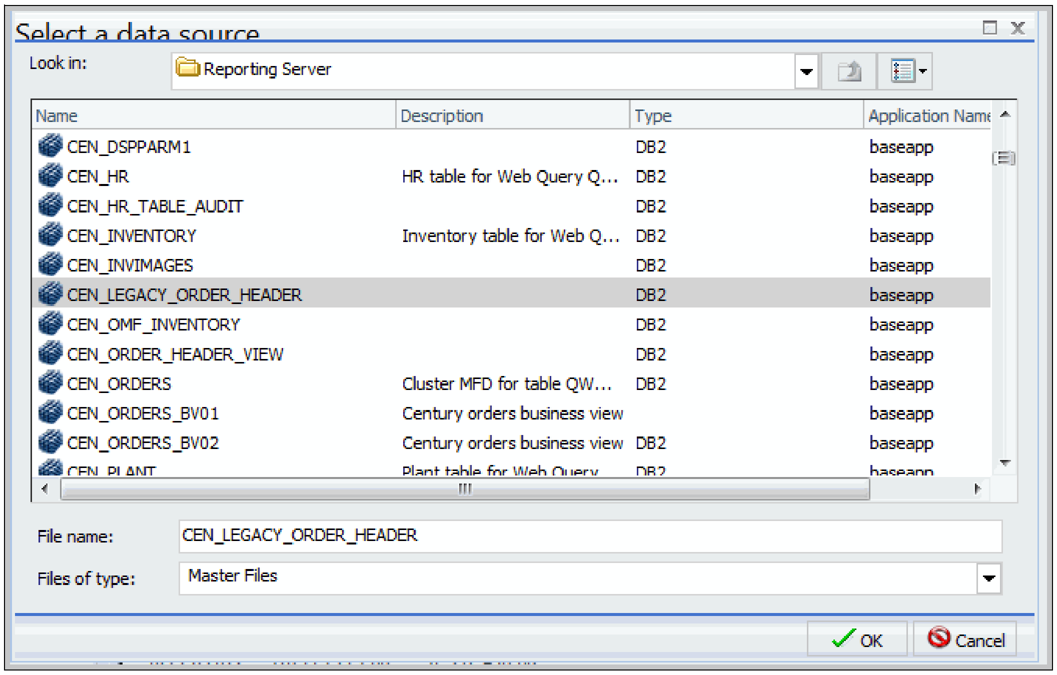
You will create a report that uses these virtual fields is in a two-dimensional format:

- Sorted/grouped vertically by the year of the order
- Sorted/grouped across by the name of the day of the week

The measure shown for each of these groupings will be the aggregated order amount.

To perform date conversions using the DB2 Web Query BIFs, follow these steps:

1. Open DB2 Web Query in a browser session and log in.
2. Create a DB2 Web Query synonym over the LEGACY_ORDER_HEADER table. Give it a prefix of `cen_`.
3. Create a new report in the Century Electronics folder
4. From the list of displayed synonyms, select **CEN_LEGACY_ORDER_HEADER** as the data source, as shown in Figure A-5.



Name	Description	Type	Application Name
CEN_DSPPARM1		DB2	baseapp
CEN_HR	HR table for Web Query Q...	DB2	baseapp
CEN_HR_TABLE_AUDIT		DB2	baseapp
CEN_INVENTORY	Inventory table for Web Q...	DB2	baseapp
CEN_INVIMAGES		DB2	baseapp
CEN_LEGACY_ORDER_HEADER		DB2	baseapp
CEN_OMF_INVENTORY		DB2	baseapp
CEN_ORDER_HEADER_VIEW		DB2	baseapp
CEN_ORDERS	Cluster MFD for table QW...	DB2	baseapp
CEN_ORDERS_BV01	Century orders business view		baseapp
CEN_ORDERS_BV02	Century orders business view	DB2	baseapp
CEN_PLANT	Plant table for Web Query	DB2	baseapp

File name: CEN_LEGACY_ORDER_HEADER

Files of type: Master Files

Figure A-5 Select synonym

e. In Info Assist, create a new Define field by selecting the **Data** and selecting the **Detail (Define)** icon, as shown in Figure A-6.

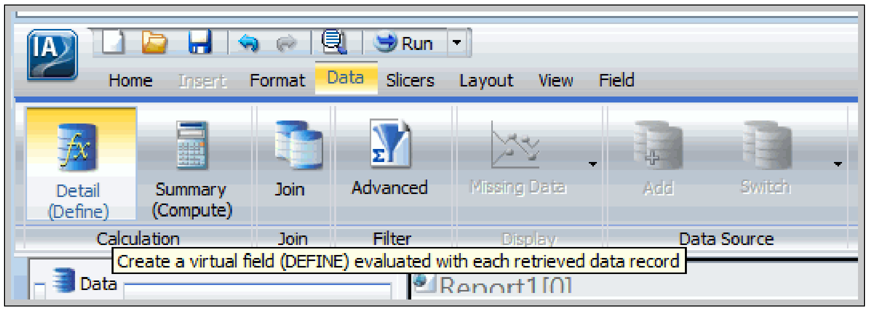


Figure A-6 Create new Define field

To convert the packed decimal fields to dates, use the DATECVT function. This function converts the field value of any standard date format or legacy date format into a new date, in either the desired standard date format or the legacy date format. These are the parameters for this function:

– date is the input legacy field to be converted.

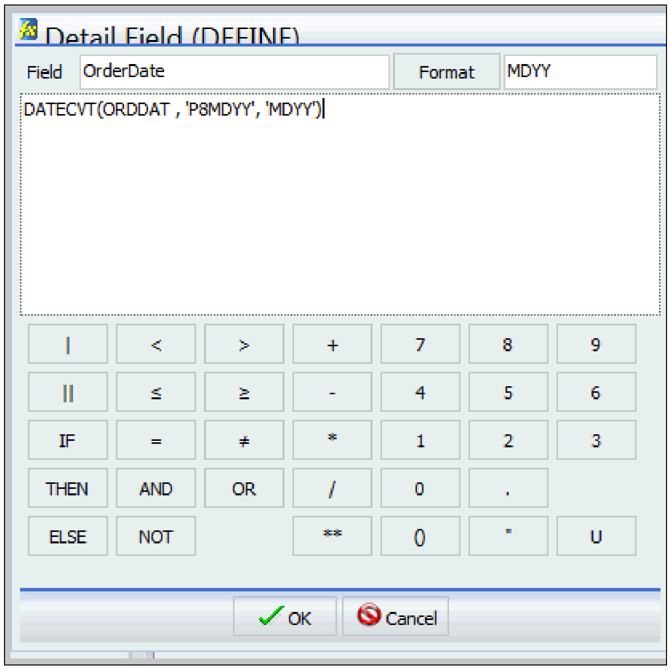
– in_format is the format of the input legacy date (for example, P8MDYY, I8MDYY, I6YMD, and A8MDYY).

– output_format is the output date format (for example, YYMD, YQ, M, DMY, and JUL).

5. From the Define Field Creator window, specify the following items, then click **OK**:

– Field: OrderDate (This is the new virtual column name.)
– Format: MDYY (This is the output date format of the new virtual column.)
– Expression: DATECVT(ORDDAT, 'P8MDYY', 'MDYY')

An example is provided in Figure A-7.



Field: OrderDate
Format: MDYY

```
DATECVT(ORDDAT , 'P8MDYY', 'MDYY')
```

Figure A-7 Define field for ORDERDATE

Note: ORDDAT is a Packed Decimal (8,0) field. If you inspect the DB2 Web Query synonym for the this field, you will notice that it is defined as P9 (rather than P8). This is normal. The extra digit is used to store the decimal point. For the purposes of date conversion, always ignore extra digits when specifying the value of the input format parameter. In this case the value should be P8MDYY.

6. Create another Define field to display just the year of the OrderDate column. Specify the following attributes, then click **OK**:
   - Field: OrderYear
   - Format: YY
   - Expression: OrderDate

   An example is provided in Figure A-8.

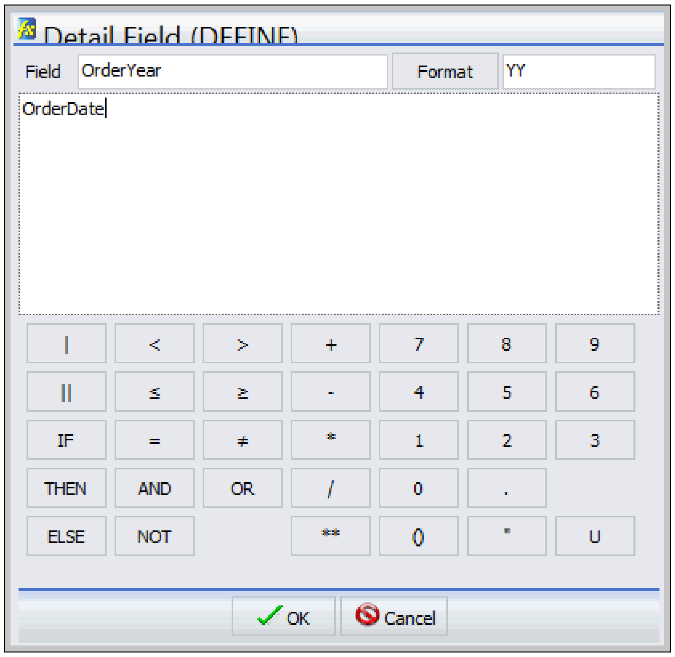


Figure A-8 Define field for ORDERYEAR

7. Create another Define field to display the day of the week (MON, TUE, and so on) of the OrderDate column. Specify the following attributes, then click **OK**:

   – Field: OrderDayOfWeek
   – Format: WT
   – Expression: OrderDate

   An example is provided in Figure A-9.

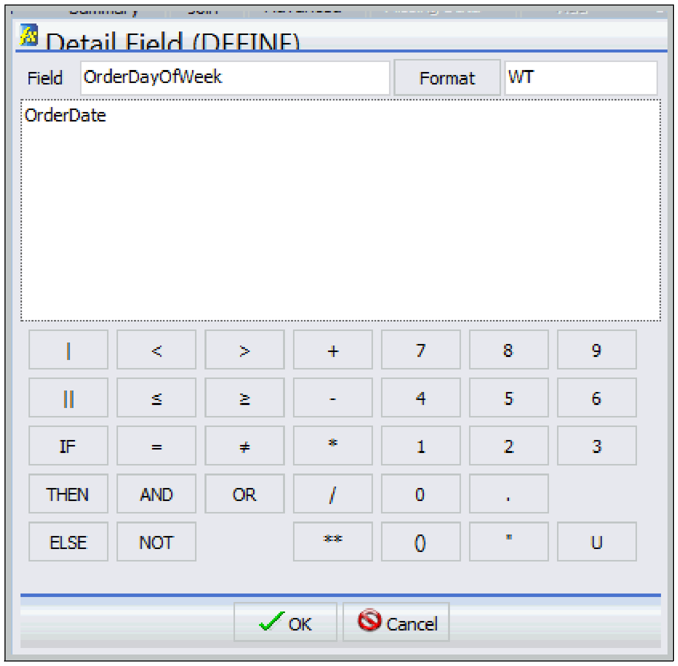


Figure A-9 Define field for ORDERDAYOFWEEK

Attention: A full list of date display formats can be found in “Date format display options” on page 791.

8. Finish the report by performing the following steps:

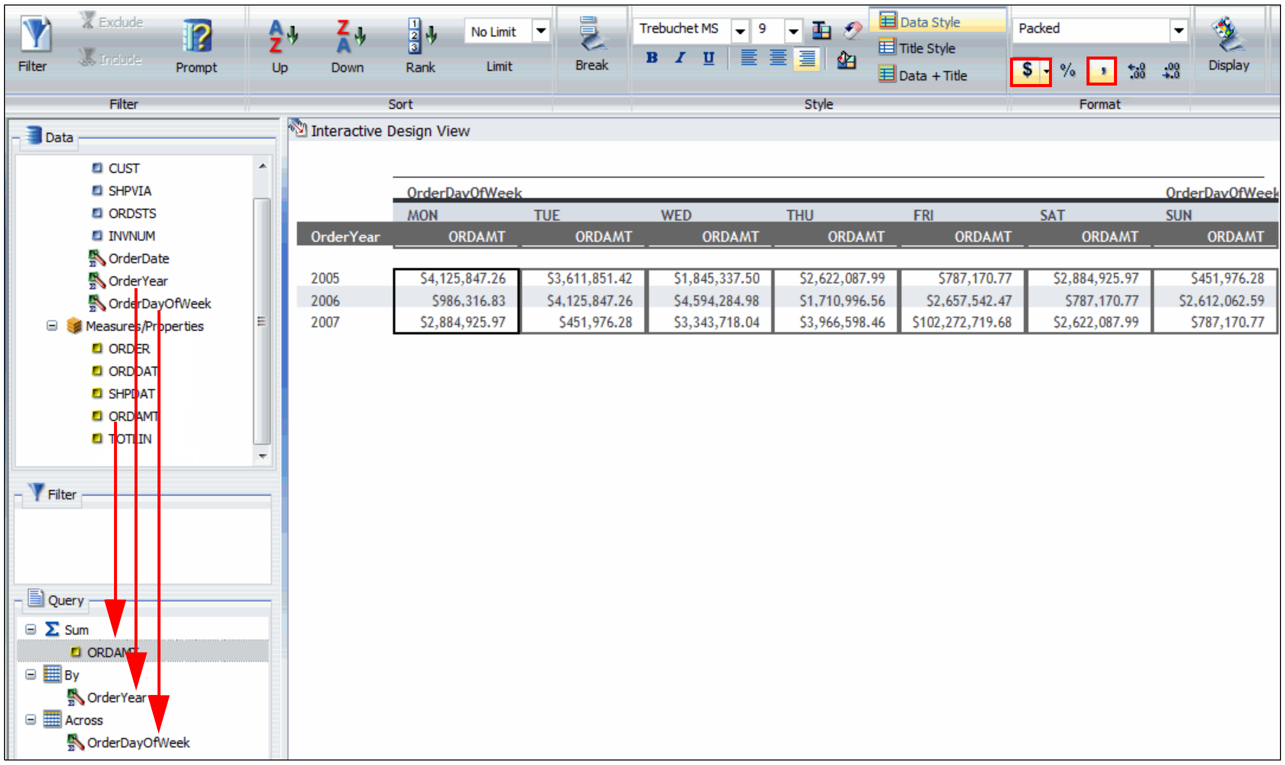
   a. Drag the new **OrderYear** field into the Sort-By panel.

   b. Drag the **OrderDayOfWeek** field into the Sort across panel.

   c. Drag the **ORDAMT** field into the Sum pane and select it. Click the Show field options icon for this field.

   d. Edit the format of the ORDAMT field. Specify comma inclusion and floating currency.

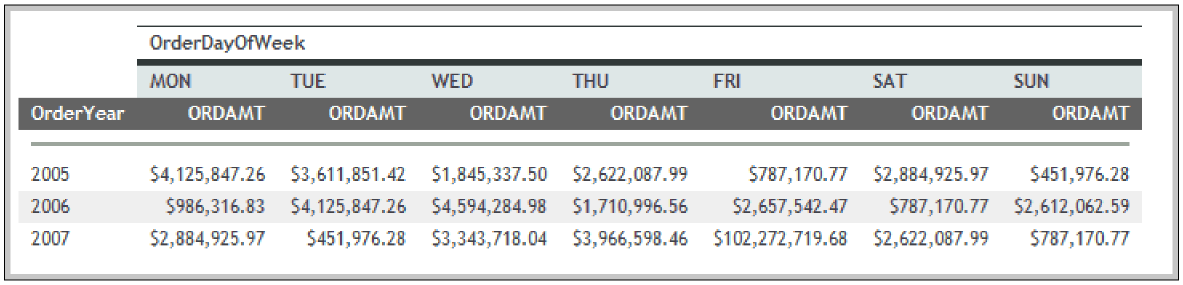
   e. When finished, the report definition should look like the example provided in Figure A-10.



	OrderDayOfWeek						
	MON	TUE	WED	THU	FRI	SAT	SUN
OrderYear	ORDAMT	ORDAMT	ORDAMT	ORDAMT	ORDAMT	ORDAMT	ORDAMT
2005	$4,125,847.26	$3,611,851.42	$1,845,337.50	$2,622,087.99	$787,170.77	$2,884,925.97	$451,976.28
2006	$986,316.83	$4,125,847.26	$4,594,284.98	$1,710,996.56	$2,657,542.47	$787,170.77	$2,612,062.59
2007	$2,884,925.97	$451,976.28	$3,343,718.04	$3,966,598.46	$102,272,719.68	$2,622,087.99	$787,170.77

Figure A-10 Report using Define fields for date conversion

9. Run the report. It should look like the example displayed in Figure A-11.



	OrderDayOfWeek						
	MON	TUE	WED	THU	FRI	SAT	SUN
OrderYear	ORDAMT	ORDAMT	ORDAMT	ORDAMT	ORDAMT	ORDAMT	ORDAMT
2005	$4,125,847.26	$3,611,851.42	$1,845,337.50	$2,622,087.99	$787,170.77	$2,884,925.97	$451,976.28
2006	$986,316.83	$4,125,847.26	$4,594,284.98	$1,710,996.56	$2,657,542.47	$787,170.77	$2,612,062.59
2007	$2,884,925.97	$451,976.28	$3,343,718.04	$3,966,598.46	$102,272,719.68	$2,622,087.99	$787,170.77

Figure A-11 Results of report that uses Define fields for date conversion

10. Save your report as `Date conversion using Web Query functions`.
11. CloseInfo Assist.

Using a date conversion table to convert to dates

Also referred to as a calendar table, a date conversion table is simply a DB2 for i table that contains one row for each individual day within a specified date range. Each row is made up of columns that represent the same date value in various ways (for example, the day of the week). Your reporting and business intelligence requirements will dictate how many date value representations you add to your conversion table.

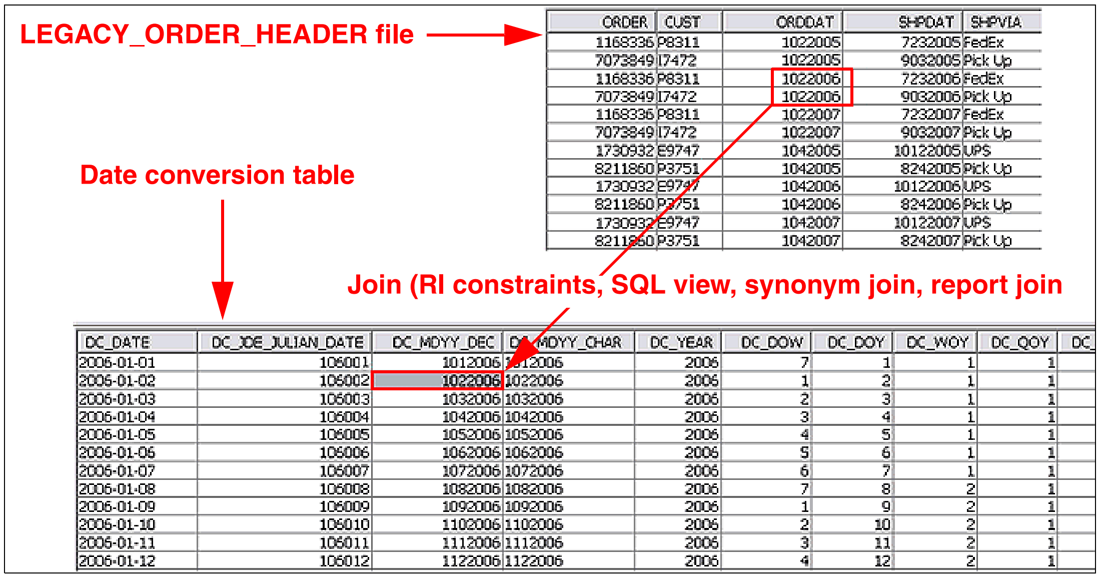
For example, a date conversion table could include the following columns:

- Julian date
- Date (a *true* DB2 date field)
- Fiscal year
- Fiscal quarter
- Day of the week (Monday, Tuesday, and so on)
- Month of the year (January, February, and so on)
- Season (spring, summer, autumn, winter)
- Same day (of the week) last year
- Week ending date
- Week of the year
- Super Bowl Sunday flag (Y or N)
- Day before a holiday flag (Y or N)
- Day after a holiday flag (Y or N)
- Full moon flag (Y or N)
- And so on

In order for this technique to work, the date conversion table must have a column that represents the date in the same format as the legacy file. You then define an inner join from the legacy file to the date conversion table using the legacy date fields as the join columns. This can be done with any of the following techniques:

- Implementing Referential Integrity (setting up Primary and Foreign Keys)
- Creating an SQL view with syntax to join your legacy files to the date conversion table
- Defining the join in the DB2 Web Query synonym
- Defining the join in each DB2 Web Query reports/graphs

When the report is run, DB2 Web Query uses the chosen join definition method to generate the SQL syntax necessary to join the legacy file and the date conversion table. For each row returned in the legacy file, the matching row (for that date) of the date conversion table is also returned, providing the report with all the various columns representing that particular date. The result is a very efficient date conversion implementation and a faster-running report. Figure A-12 illustrates how the join to the date conversion table works.



ORDER	CUST	ORDDAT	SHPDAT	SHPVIA
1168336	P8311	1022005	7232005	FedEx
7073849	I7472	1022005	9032005	Pick Up
1168336	P8311	1022006	7232006	FedEx
7073849	I7472	1022006	9032006	Pick Up
1168336	P8311	1022007	7232007	FedEx
7073849	I7472	1022007	9032007	Pick Up
1730932	E9747	1042005	10122005	UPS
8211860	P3751	1042005	8242005	Pick Up
1730932	E9747	1042006	10122006	UPS
8211860	P3751	1042006	8242006	Pick Up
1730932	E9747	1042007	10122007	UPS
8211860	P3751	1042007	8242007	Pick Up

DC_DATE	DC_JDE_JULIAN_DATE	DC_MDYY_DEC	DC_MDYY_CHAR	DC_YEAR	DC_DOW	DC_DOY	DC_WOY	DC_QOY	DC_
2006-01-01	106001	1012006	1012006	2006	7	1	1	1	
2006-01-02	106002	1022006	1022006	2006	1	2	1	1	
2006-01-03	106003	1032006	1032006	2006	2	3	1	1	
2006-01-04	106004	1042006	1042006	2006	3	4	1	1	
2006-01-05	106005	1052006	1052006	2006	4	5	1	1	
2006-01-06	106006	1062006	1062006	2006	5	6	1	1	
2006-01-07	106007	1072006	1072006	2006	6	7	1	1	
2006-01-08	106008	1082006	1082006	2006	7	8	2	1	
2006-01-09	106009	1092006	1092006	2006	1	9	2	1	
2006-01-10	106010	1102006	1102006	2006	2	10	2	1	
2006-01-11	106011	1112006	1112006	2006	3	11	2	1	
2006-01-12	106012	1122006	1122006	2006	4	12	2	1	

Figure A-12 Joining files to date conversion table

With this technique, each date format (column) in the date conversion table is available to the DB2 Web Query development tools and therefore easily can be included in any report. This gives the report developer the ability to effortlessly do some interesting customized analysis. For example:

- What are the profit margins on days before and after holidays?
- How many bags of corn chips are sold the week before the Super Bowl?
- What is the rate of product returns on the day after Christmas as compared to any other day of the year?
- Are more galoshes sold in the spring or the fall?
- How many boxes of diapers are sold on days when there is a full moon?

The usefulness of a date conversion table goes well beyond DB2 Web Query. It can be implemented in any application or tool that has access to DB2 for i. If you have RPG programs, you can join to this table (using either embedded SQL or native Record Level Access operations such as CHAIN) to perform quick and easy date conversions. It can even be used in your Query/400 reports.

Implementing the date conversion method is relatively simple and can be performed in the following four steps:

1. Create the date conversion table.
2. Populate the date conversion table.
3. Add join segment to the synonym.
4. Create reports.

Creating the date conversion table

A pre-populated date conversion table named DATE_CONV is included in the QWQCENT library. Starting in version 2.1, this library is shipped in a save file format when you install the DB2 Web Query product 5733WQX. The name of the save file is QWQCENT and it resides in library QWEBQRY. The QWQCENT library may be updated occasionally to provide more objects, columns, stored procedure, and other examples. So you may want to occasionally check to see if a new version has been provided and if so, restore it. At the time this book was published, DATE_CONV contained the columns shown in Table A-6.

Table A-6 Date conversion table

Column Name	Data Length	Length	Description
DC_DATE	DATE	4	Date (date format)
DC_JDE_JULIAN_DATE	DECIMAL	6	JDE Julian Date (CYYDDD decimal)
DC_MDYY_DEC	DECIMAL	8	Date (MMDDYYYY packed decimal)
DC_MDYY_ZONED	NUMERIC	8	Date (MMDDYYYY zoned decimal)
DC_MDYY_CHAR	CHAR	8	Date (MMDDYYYY character)
DC_YYMD_DEC	DECIMAL	8	Date (YYYYMMDD packed decimal)
DC_YYMD_ZONED	NUMERIC	8	Date (YYYYMMDD zoned decimal)
DC_YYMD_CHAR	CHAR	8	Date (YYYYMMDD character)
DC_MDY_DEC	DECIMAL	6	Date (MMDDYY packed decimal)

DC_MDY_ZONED	NUMERIC	6	Date (MMDDYY zoned decimal)
DC_MDY_CHAR	CHAR	6	Date (MMDDYY character)
DC_YMD_DEC	DECIMAL	6	Date (YYMMDD packed decimal)
DC_YMD_ZONED	NUMERIC	6	Date (YYMMDD zoned decimal)
DC_YMD_CHAR	CHAR	6	Date (YYMMDD character)
DC_CC_CHAR	CHAR	2	Century (2 characters)
DC_YY_CHAR	CHAR	2	Year (2 characters)
DC_MM_CHAR	CHAR	2	Month (2 characters)
DC_DD_CHAR	CHAR	2	Day (2 characters)
DC_YEAR	INTEGER	4	Year (4 digits)
DC_DOW	INTEGER	4	Day of week (1-7)
DC_DOW_ISO	INTEGER	4	Day of week (1-7)
DC_DOY	INTEGER	4	Day of year (1-366)
DC_WOY	INTEGER	4	Week of year (1-52)
DC_WOY_ISO	INTEGER	4	Week of year (1-53)
DC_QOY	INTEGER	4	Quarter of year (1-4)
DC_CC	NUMERIC	2	Century (2 digits)
DC_YY	NUMERIC	2	Year (2 digits)
DC_MM	NUMERIC	2	Month (2 digits)
DC_DD	NUMERIC	2	Day (2 digits)
DC_CCYYMM	NUMERIC	6	Century, Year, Month CCYYMM (6 digits)
DC_DAY_NAME	CHAR	9	Day Name (Monday, and so on)
DC_QUARTER_NAME	CHAR	6	Quarter name (2008Q1)
DC_WEEKEND	CHAR	1	Weekend Flag (Y or N)
DC_HOLIDAY	CHAR	1	Holiday (Y or N)
DC_DAY_BEFORE_HOLIDAY	CHAR	1	Day Before Holiday (Y or N)
DC_DAY_AFTER_HOLIDAY	CHAR	1	Day AfterHoliday (Y or N)
DC_FULL_MOON	CHAR	1	Full Moon (Y or N)
DC_SEASON	CHAR	6	Season (Spring, Summer, Autumn, Winter)
DC_FISCAL_YEAR	INTEGER	4	Fiscal year (4 digits)
DC_FISCAL_QUARTER	INTEGER	4	Fiscal quarter (1-4)
DC_MONTH_NAME	CHAR	9	Month name (January, etc)

DC_MONTH_ABRV	CHAR	3	Month abbreviation (Jan, Feb, and so on)
DC_JULIAN	NUMERIC	7	Date in Julian format
DC_CYYMMDD	DECIMAL	7	CYYMMDD packed C = 0 for 1900 & C = 1 for 2000
DC_EXCEL_DATE	INTEGER	4	Date in Excel format
DC_WEEK_STARTING_DATE	DATE	4	Week starting date (the prior Saturday)
DC_WEEK_ENDING_DATE	DATE	4	Week ending date (the next Friday)
DC_SAME_DAY_LAST_YEAR	DATE	4	Same day last year
DC_CURRENT_DAY	CHAR	1	Current Day (Y/N)
DC_CURRENT_WEEK	CHAR	1	Current Week (Y/N)
DC_CURRENT_MONTH	CHAR	1	Current Month (Y/N)
DC_CURRENT_QUARTER	CHAR	1	Current Quarter (Y/N)
DC_CURRENT_YEAR	CHAR	1	Current Year (Y/N)
DC_CURRENT_YTD	CHAR	1	Current Year to Date (Y/N)
DC_CURRENT_DAY_LAST_YEAR	CHAR	1	Current Day Last Year (Y/N)
DC_CURRENT_WEEK_LAST_YEAR	CHAR	1	Current Week Last Year (Y/N)
DC_CURRENT_MONTH_LAST_YEAR	CHAR	1	Current Month Last Year (Y/N)
DC_CURRENT_QUARTER_LAST_YEAR	CHAR	1	Current Quarter Last Year (Y/N)
DC_CURRENT_YEAR_LAST_YEAR	CHAR	1	Current Year Last Year (Y/N)
DC_CURRENT_YTD_LAST_YEAR	CHAR	1	Current Year To Date Last Year (Y/N)
DC_PREVIOUS_MONTH	CHAR	1	Previous Month (Y/N)
DC_PREVIOUS_FISCAL_YEAR	CHAR	1	Previous fiscal year (Y/N)
DC_CURRENT_FISCAL_YEAR	CHAR	1	Current fiscal year (Y/N)
DC_PREVIOUS_FISCAL_YTD	CHAR	1	Previous fiscal year to date (Y/N)
DC_CURRENT_FISCAL_YTD	CHAR	1	Current fiscal year to date (Y/N)
DC_NTH_DAY_OF_WEEK_OF_MONTH	INTEGER	4	Nth Day of the Week of the month

Populating and maintaining the date conversion table

As mentioned previously, a pre-populated date conversion table is included in the QWQCENT library. Therefore, unless there are columns that you want to add or remove, you can simply use this version of the table. If you are interested in creating a customized version, an SQL stored procedure used to populate this table is also provided in the QWQCENT library. The name of this stored procedure is LOAD_DATE_CONVERSION_TABLE and it creates one row for each day on and between January 1, 1900 and December 31, 2030. You can see the SQL source code of this stored procedure by using the Generate SQL option from System i Navigator as shown in Figure A-13.

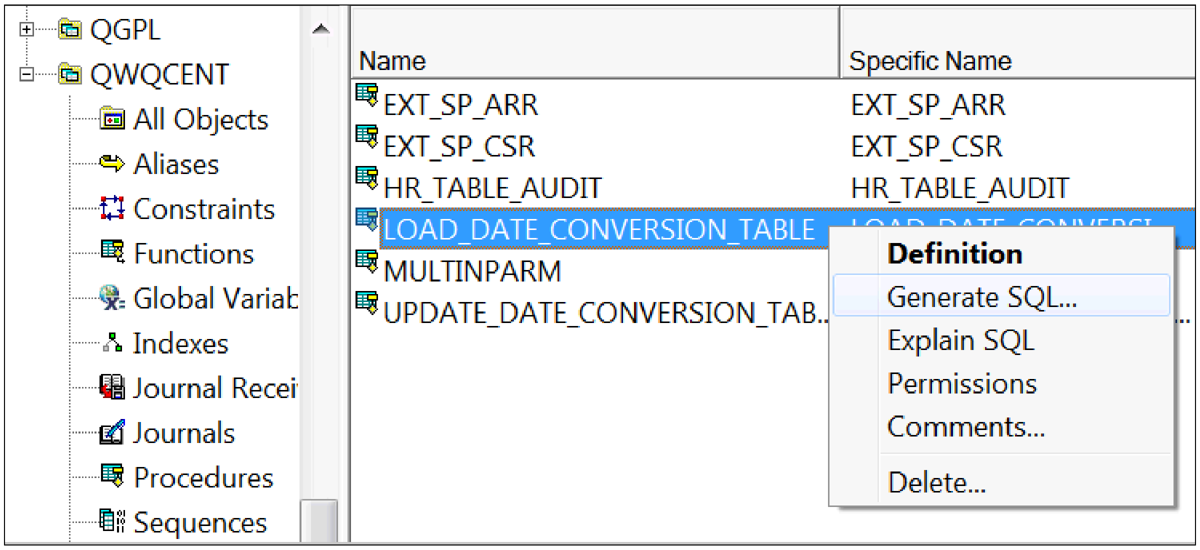


Figure A-13 Generate SQL for LOAD_DATE_CONVERSION_TABLE procedure

Notice that several of the columns in the date conversion table are *current* flags that contain a value of ‘Y’ or ‘N’ depending on whether the condition for that flag is met. For example, the column DC_CURRENT_YEAR contains a ‘Y’ if that row represents a date whose year is equal to the current year. So if the current date is June 20, 2012, then the value for the DC_CURRENT_YEAR column would be ‘Y’ for all of the rows that represent dates for the year 2012. Because these are dynamic values, you must add a process to keep these columns in the date conversion table updated. Again a stored procedure that does this is provided in the QWQCENT library and it is named UPDATE_DATE_CONVERSION_TABLE. You can also see the source for this procedure by using the Generate SQL option as described above.

To keep these current flags accurate, all you must do is make sure that the UPDATE_DATE_CONVERSION_TABLE stored procedure runs every day. To do this, create a job scheduler entry to call this stored procedure daily. This can be accomplished by taking the following steps:

1. Create a source file member to store an SQL statement script:

   ```
   ADDPFM FILE(QGPL/QTXTSRC) MBR(UPDDATCONV)
   ```

2. Edit this source file member and add the following SQL statement to call the stored procedure:

   ```
   CALL QWQCENT/UPDATE_DATE_CONVERSION_TABLE
   ```

3. Now add a job scheduler job to run the SQL script in the source file member:

   ```
   ADDJOBJS JOB(UPDDATCONV) SCDCDE(*DAILY) TIME(0010) CMD(RUNSQLSTM
   SRCFILE(QGPL/QTXTSRC) SRCMBR(UPDDATCONV) COMMIT(*NONE) NAMING(*SYS))
   ```

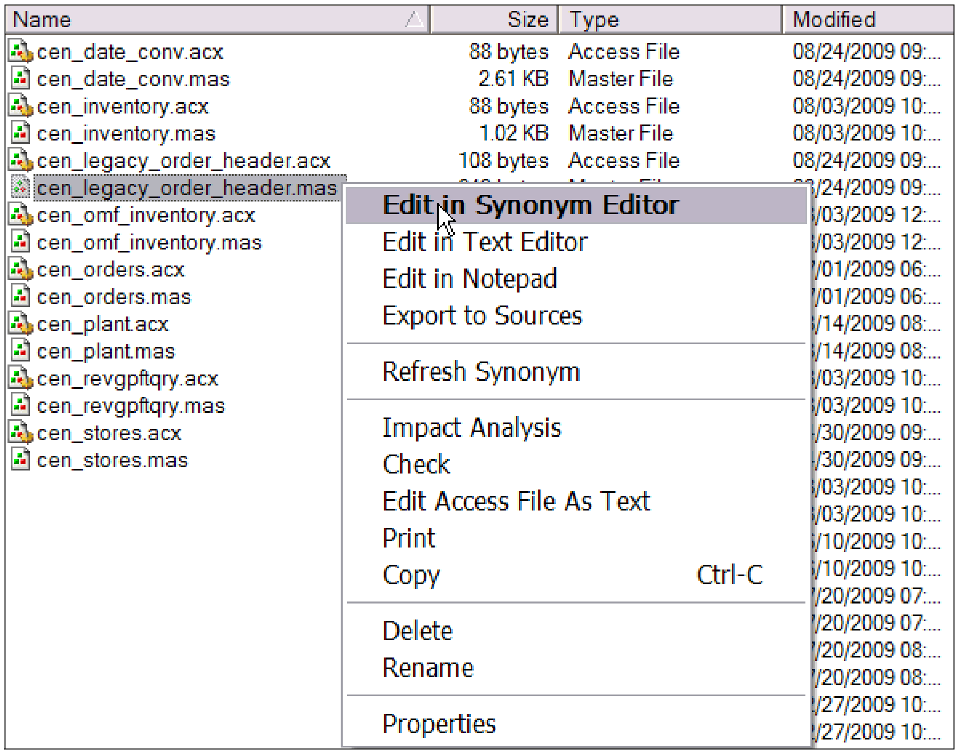
This job scheduler job runs daily at 1 a.m. and calls the RUNSQLSTM command to run the script (in the specified source file member) through SQL.

Adding join segments to the synonym

As described earlier, there are several ways that the date conversion table can be joined to the legacy file. This section describes how to implement the join in the DB2 Web Query synonym. In general, it is advantageous to define the joins in the synonym (rather than individual reports and graphs) for the simple reason that this piece of logic is centralized in one location. Each report and graph that uses the synonym will be able to use the join segment (and not have to redefine it).

For this example, you will use the table in library QWQCENT named LEGACY_ORDER_HEADER (the 10-character system name is ORDHDR):

1. Create a DB2 Web Query synonym over the DATE_CONV table. Give it a a prefix of `cen_`.
2. Open DB2 Web Query Developer Workbench and open a connection to your environment.
3. Find the `cen_legacy_order_header.mas` file under **Data Servers** → **EDASERVE** → **Applications** → **baseapp**.
4. Right-click this file and select **Edit in Synonym Editor**, as shown in Figure A-14.



Name	Size	Type	Modified
cen_date_conv.acx	88 bytes	Access File	08/24/2009 09:...
cen_date_conv.mas	2.61 KB	Master File	08/24/2009 09:...
cen_inventory.acx	88 bytes	Access File	08/03/2009 10:...
cen_inventory.mas	1.02 KB	Master File	08/03/2009 10:...
cen_legacy_order_header.acx	108 bytes	Access File	08/24/2009 09:...
cen_legacy_order_header.mas			/24/2009 09:...
cen_omf_inventory.acx			/03/2009 12:...
cen_omf_inventory.mas			/03/2009 12:...
cen_orders.acx			/01/2009 06:...
cen_orders.mas			/01/2009 06:...
cen_plant.acx			/14/2009 08:...
cen_plant.mas			/14/2009 08:...
cen_revgpftqry.acx			/03/2009 10:...
cen_revgpftqry.mas			/03/2009 10:...
cen_stores.acx			/30/2009 09:...
cen_stores.mas			/30/2009 09:...
			/03/2009 10:...
			/03/2009 10:...
			/10/2009 10:...
			/10/2009 10:...
			/20/2009 07:...
			/20/2009 07:...
			/20/2009 08:...
			/20/2009 08:...
			/27/2009 10:...
			/27/2009 10:...

Figure A-14 Edit cen_legacy_order_header

At this point, add a join segment to the date conversion table.

5. Right-click the **CEN_LEGAGY_ORDER_HEADER** segment name and select **Insert** → **Copy of Existing Synonym**, as shown in Figure A-15.

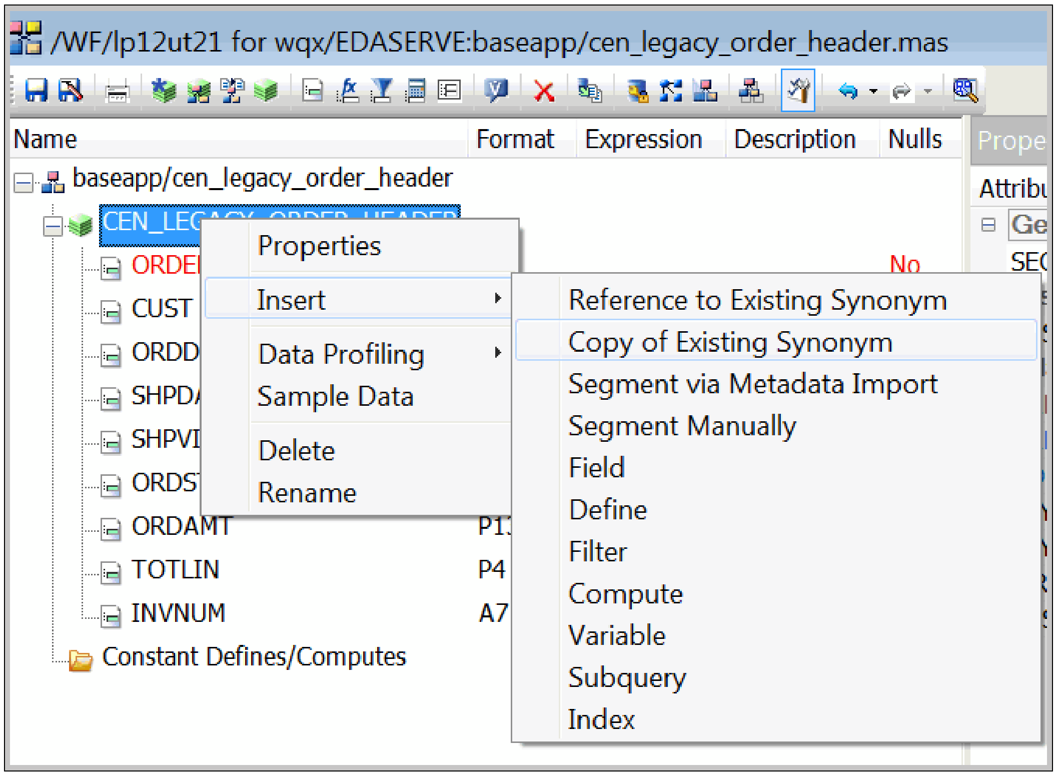
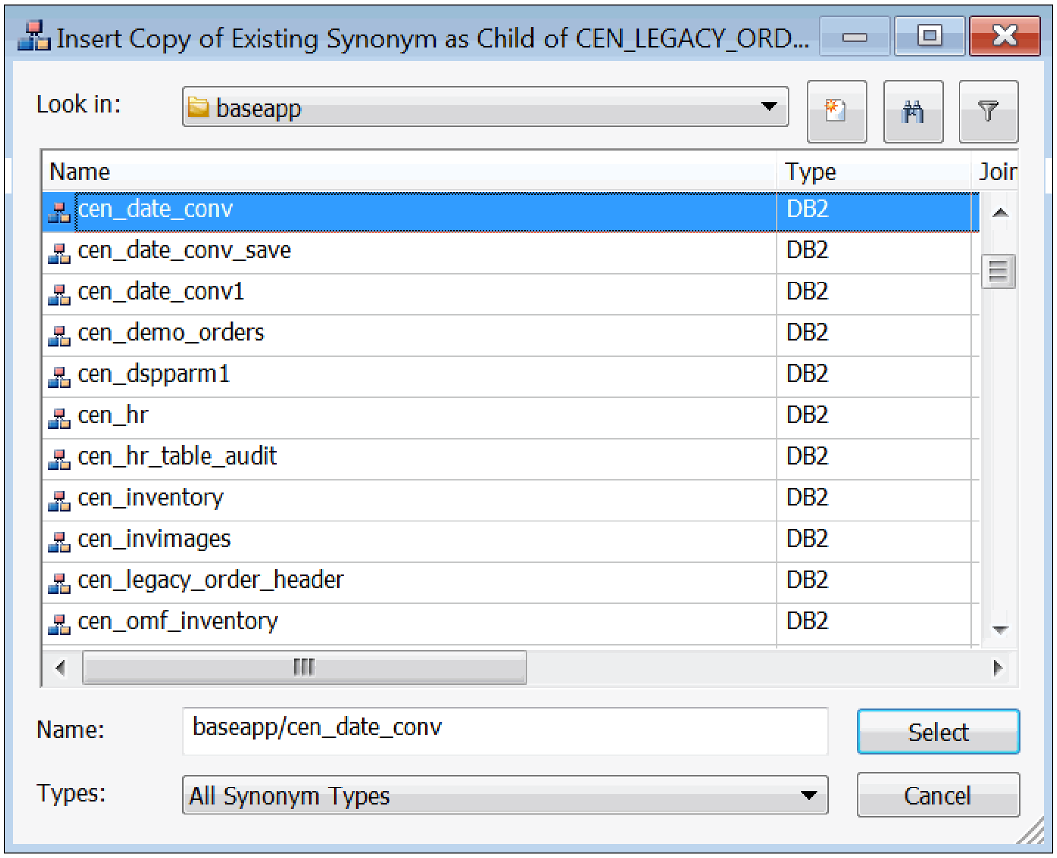


Figure A-15 Add join segment to legacy_orders_table

A list of synonyms in the BASEAPP folder is presented.

6. As demonstrated in Figure A-16, find and select the **cen_date_conv** synonym and click **Select**.


Insert Copy of Existing Synonym as Child of CEN_LEGACY_ORD...
Look in:
baseapp
Select
Cancel
Types:
All Synonym Types

Name	Type	Joir
cen_date_conv	DB2	
cen_date_conv_save	DB2	
cen_date_conv1	DB2	
cen_demo_orders	DB2	
cen_dspparm1	DB2	
cen_hr	DB2	
cen_hr_table_audit	DB2	
cen_inventory	DB2	
cen_invimages	DB2	
cen_legacy_order_header	DB2	
cen_omf_inventory	DB2	

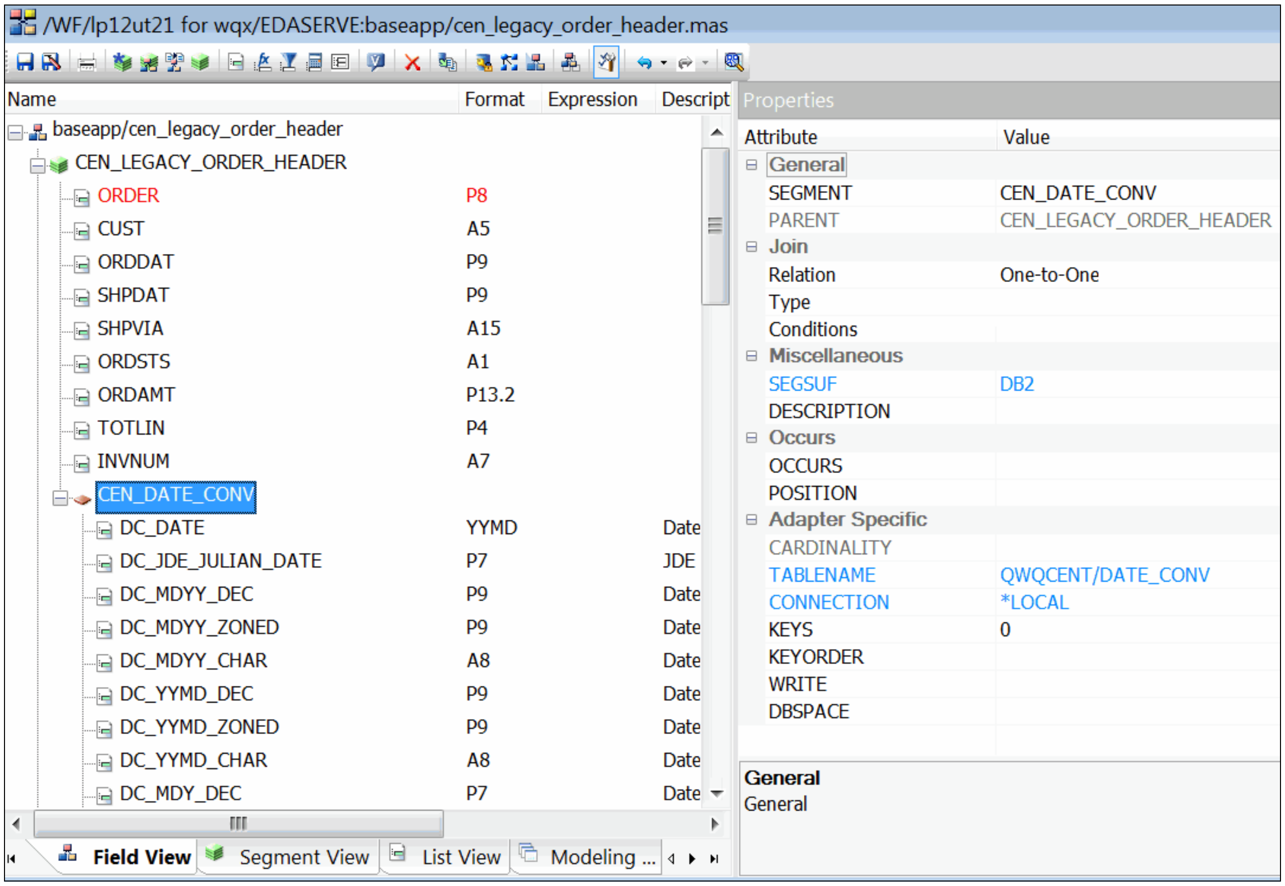
Name: baseapp/cen_date_conv

Figure A-16 Select cen_date_conv synonym

The new segment is added to the synonym. Next you must change the segment type from Multiple to Unique.

7. Select the new **CEN_DATE_CONV** join segment and locate the Type attribute under SEGTYPE.

8. This is a drop-down list. Click it to and change the type to **One to One**, as shown in Figure A-17. This instructs DB2 Web Query to rely on SQL syntax to create the join and when accessing the joined columns (allowing you to sort and select fields across multiple join segments).


/WF/lp12ut21 for wqx/EDASERVE:baseapp/cen_legacy_order_header.mas

Name	Format	Expression	Descript
baseapp/cen_legacy_order_header			
CEN_LEGACY_ORDER_HEADER			
ORDER	P8		
CUST	A5		
ORDDAT	P9		
SHPDAT	P9		
SHPVIA	A15		
ORDSTS	A1		
ORDAMT	P13.2		
TOTLIN	P4		
INVNUM	A7		
CEN_DATE_CONV			
DC_DATE	YYMD		Date
DC_JDE_JULIAN_DATE	P7		JDE
DC_MDYY_DEC	P9		Date
DC_MDYY_ZONED	P9		Date
DC_MDYY_CHAR	A8		Date
DC_YYMD_DEC	P9		Date
DC_YYMD_ZONED	P9		Date
DC_YYMD_CHAR	A8		Date
DC_MDY_DEC	P7		Date

Properties

Attribute	Value
General	
SEGMENT	CEN_DATE_CONV
PARENT	CEN_LEGACY_ORDER_HEADER
Join	
Relation	One-to-One
Type	
Conditions	
Miscellaneous	
SEGSUF	DB2
DESCRIPTION	
Occurs	
OCCURS	
POSITION	
Adapter Specific	
CARDINALITY	
TABLENAME	QWQCENT/DATE_CONV
CONNECTION	*LOCAL
KEYS	0
KEYORDER	
WRITE	
DBSPACE	

General
General

Figure A-17 Change segment type to One to One

9. Define the fields that will be used to join the tables together. As shown in Figure A-18, right-click again the **CEN_DATE_CONV** segment and select **Join Properties**.

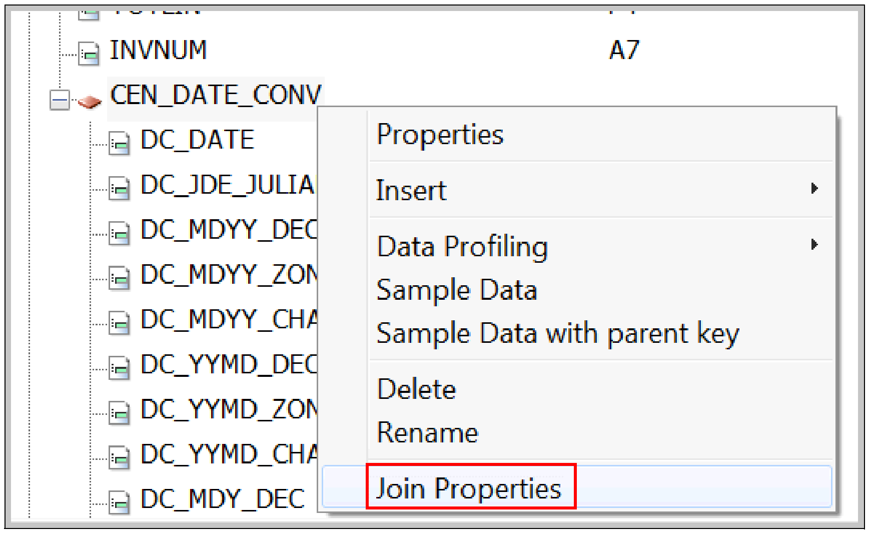


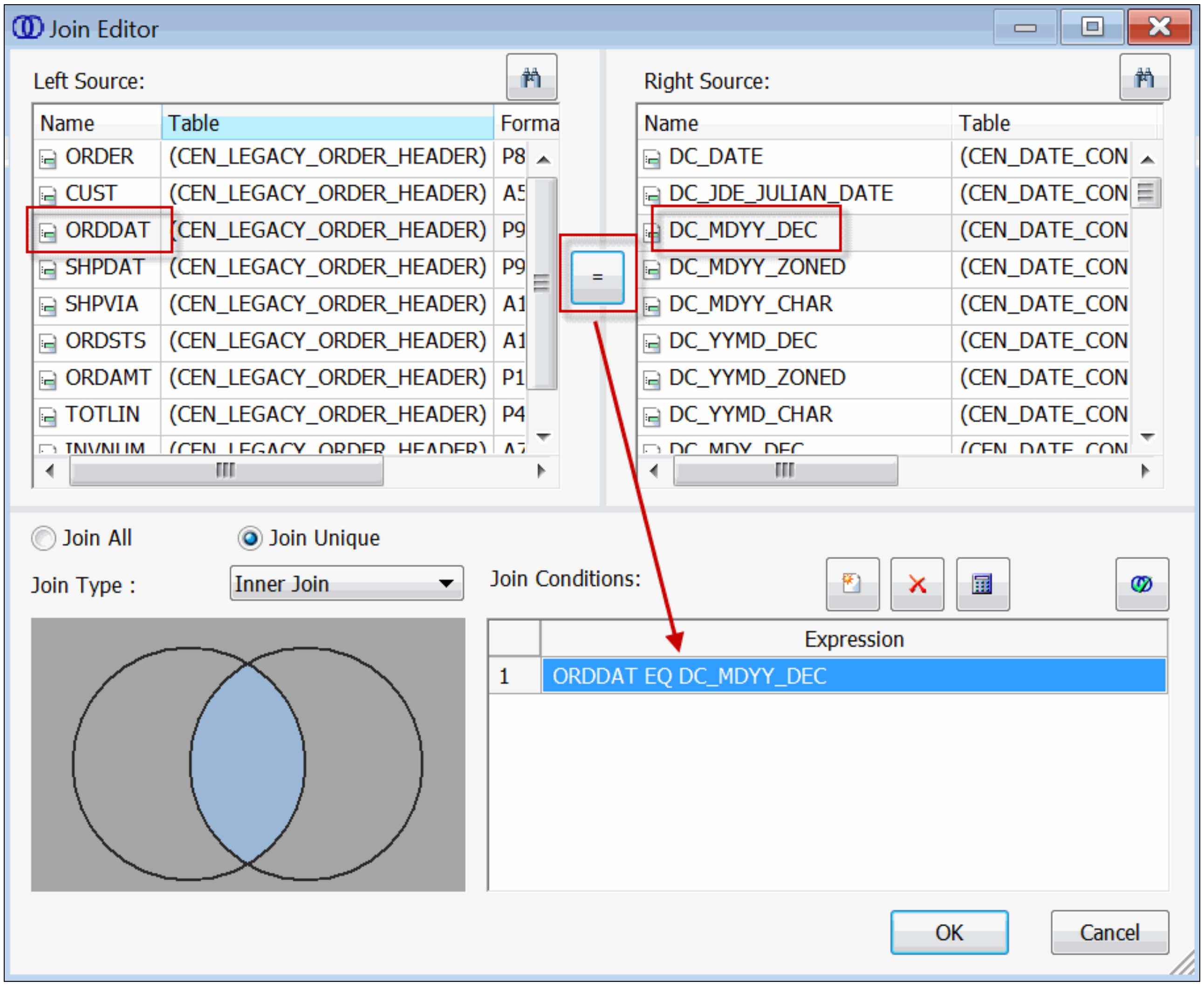
Figure A-18 Select Join Properties

10. In the Join Properties window, do the following steps:

   a. From the list of columns in the CEN_LEGACY_ORDER_HEADER segment, select **ORDDAT** so that it is highlighted.

   b. From the list of columns in the CEN_DATE_CONV segment, select **DC_MDYY_DEC** so that it is highlighted. The values in the ORDDAT are in this format, thus this is the column in the date conversion table to use as the join column.

   c. Click the **=** icon located between the two segment lists. This action should bring the selected join columns down into the expression pane. An example is shown in Figure A-19.



Left Source:

Name	Table	Forma
ORDER	(CEN_LEGACY_ORDER_HEADER)	P8
CUST	(CEN_LEGACY_ORDER_HEADER)	A5
ORDDAT	(CEN_LEGACY_ORDER_HEADER)	P9
SHPDAT	(CEN_LEGACY_ORDER_HEADER)	P9
SHPVIA	(CEN_LEGACY_ORDER_HEADER)	A1
ORDSTS	(CEN_LEGACY_ORDER_HEADER)	A1
ORDAMT	(CEN_LEGACY_ORDER_HEADER)	P1
TOTLIN	(CEN_LEGACY_ORDER_HEADER)	P4
INVNUM	(CEN_LEGACY_ORDER_HEADER)	A7

Right Source:

Name	Table
DC_DATE	(CEN_DATE_CON
DC_JDE_JULIAN_DATE	(CEN_DATE_CON
DC_MDYY_DEC	(CEN_DATE_CON
DC_MDYY_ZONED	(CEN_DATE_CON
DC_MDYY_CHAR	(CEN_DATE_CON
DC_YYMD_DEC	(CEN_DATE_CON
DC_YYMD_ZONED	(CEN_DATE_CON
DC_YYMD_CHAR	(CEN_DATE_CON
DC_MDY_DEC	(CEN_DATE_CON

	Expression
1	ORDDAT EQ DC_MDYY_DEC

Figure A-19 Define join columns

11. Click **OK**. To help the report developers identify what legacy date field this date conversion segment represents, rename the segment to the name of the field that it is converting. If you do not do this, the segment names to the date conversion table many be ambiguous (your report developers may not be certain of what legacy date field they represent). This is particularly true if you have multiple legacy date fields in your synonym because to convert them you must define a join segment to date_conv for each one.

Tip: If you are very meticulous (and ambitious), you could even completely remove the ambiguity by renaming each of the individual column names in the date conversion segments (for example, rename DC_YEAR to ORDDAT_YEAR). This can be done by highlighting the column and selecting **Rename** from the right-click menu. That is really going the extra mile, but your report developers would very much appreciate this.

12. Highlight the segment and select **Rename** from the right-click menu. This is shown in Figure A-20.

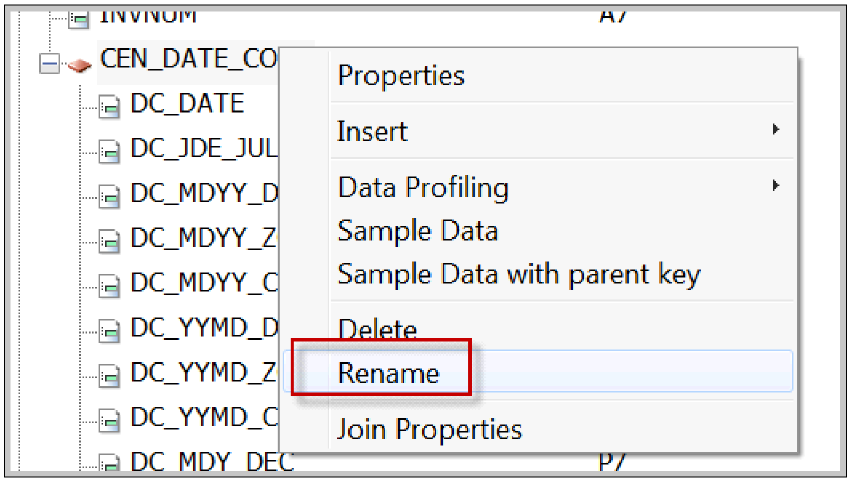
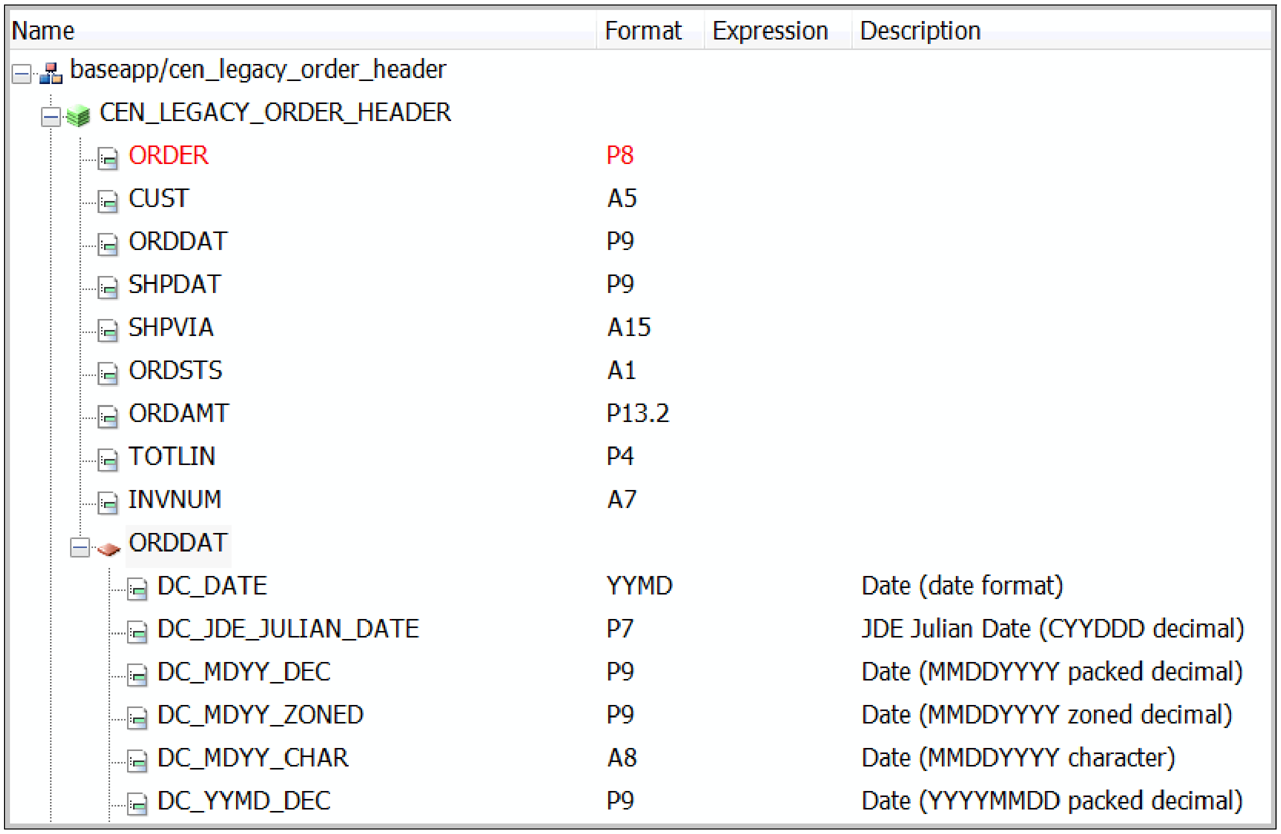


Figure A-20 Rename the segment

13. Specify **ORDDAT** as the segment name. When you are finished, your synonym will look like the example provided in Figure A-21.



Name	Format	Expression	Description
baseapp/cen_legacy_order_header			
CEN_LEGACY_ORDER_HEADER			
ORDER	P8		
CUST	A5		
ORDDAT	P9		
SHPDAT	P9		
SHPVIA	A15		
ORDSTS	A1		
ORDAMT	P13.2		
TOTLIN	P4		
INVNUM	A7		
ORDDAT			
DC_DATE	YYMD		Date (date format)
DC_JDE_JULIAN_DATE	P7		JDE Julian Date (CYYDDD decimal)
DC_MDYY_DEC	P9		Date (MMDDYYYY packed decimal)
DC_MDYY_ZONED	P9		Date (MMDDYYYY zoned decimal)
DC_MDYY_CHAR	A8		Date (MMDDYYYY character)
DC_YYMD_DEC	P9		Date (YYYYMMDD packed decimal)

Figure A-21 Synonym with new ORDDAT segment

14. Repeat step 5 on page 809 through step 13 on page 814 for the SHPDAT legacy date field. Be sure that you rename the segment SHPDAT.
15. Save the synonym.
16. Close the Edit synonym window.

Tip: For performance reasons, now would be a good time to create indexes over your join columns. The DB2 optimizer will be able to use these indexes for statistics and implementation during query execution. Here are the two SQL statements for creating indexes over the LEGACY_ORDER_HEADER and DATE_CONV tables:

```
CREATE INDEX LEGACY_ORDER_HEADER_INDEX_00001 ON LEGACY_ORDER_HEADER (ORDDAT
ASC);
CREATE INDEX DATE_CONV_INDEX_00001 ON DATE_CONV (DC_MDYY_DEC  ASC);
```

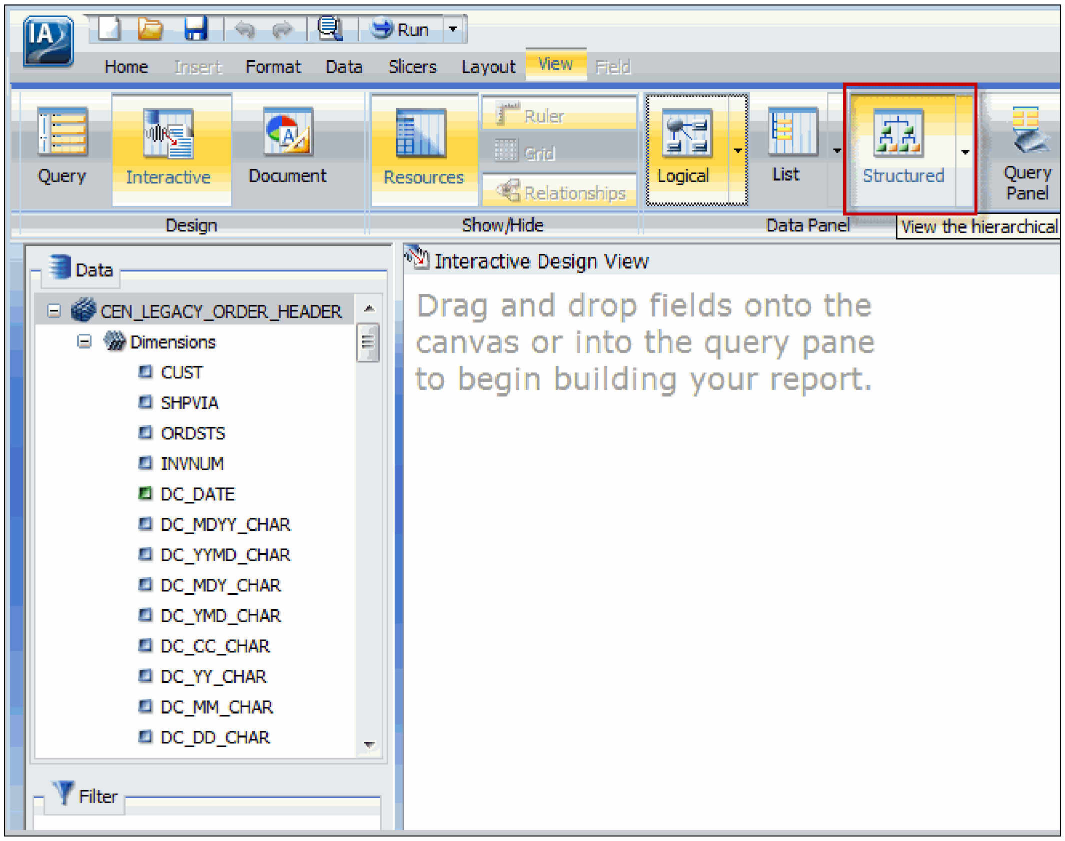
Creating reports using the date conversion table

Once the join has been defined in the synonym, the next step is to create a report over this updated synonym. Follow these steps:

1. Return to your DB2 Web Query browser session.
2. Create a new report in the Century Electronics folder using Info Assist.

3. Select the **CEN_LEGACY_ORDER_HEADER** synonym as the report data source.

   The Info Assist tool is presented. Notice that all of the columns from the date conversion table now appear in the list of fields to choose from. If your default view is set to Logical, you may notice that the list is now long and cluttered. To organize it better, click the Structured icon under the View ribbon, as shown in Figure A-22, so that the list is organized into a hierarchical tree view.



CEN_LEGACY_ORDER_HEADER
Dimensions
CUST
SHPVIA
ORDSTS
INVNUM
DC_DATE
DC_MDYY_CHAR
DC_YYMD_CHAR
DC_MDY_CHAR
DC_YMD_CHAR
DC_CC_CHAR
DC_YY_CHAR
DC_MM_CHAR
DC_DD_CHAR

Drag and drop fields onto the canvas or into the query pane to begin building your report.

Figure A-22 Change to Structured view

As shown in Figure A-23, taking this action groups the synonym columns under their respective segments and allows you to expand and collapse the list of columns under each segment.

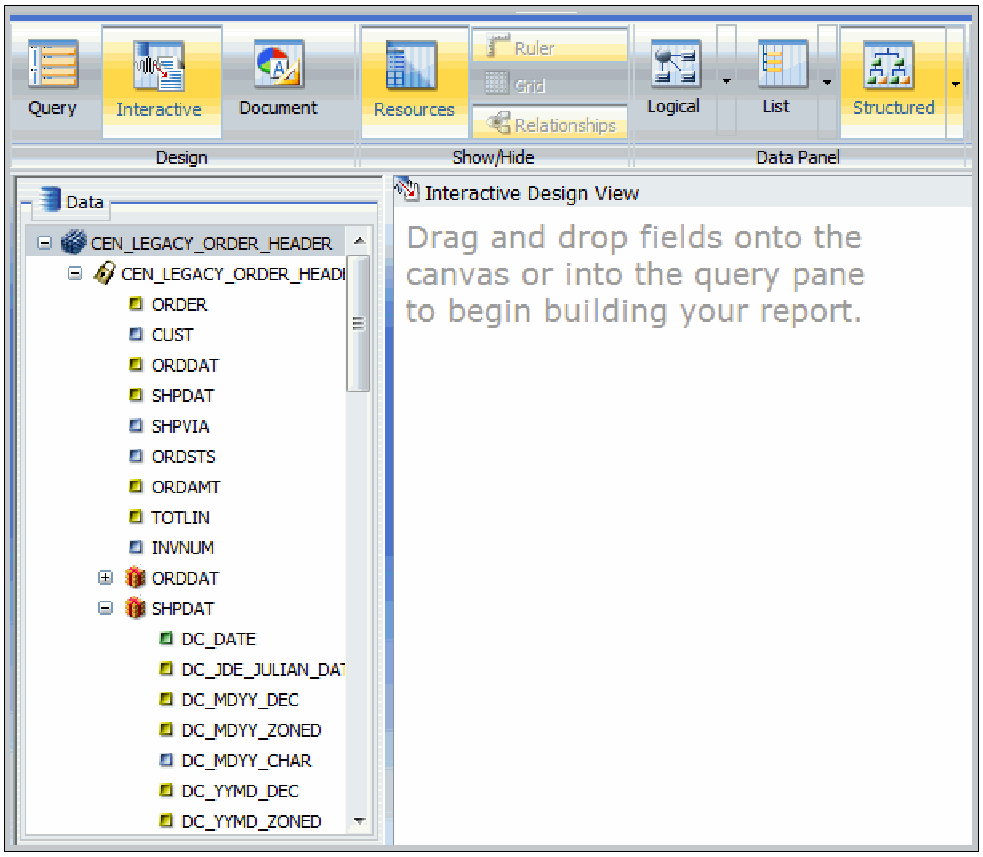


Figure A-23 Example of Structured view

4. Finish the report by performing the following steps:

   a. Drag the new **ORDDAT.DC_YEAR** field into the By field container in the Query pane.

   b. Drag the **ORDDAT.DC_DOW** field into the Across field container in the Query pane. Make it an invisible field as shown in Figure A-24. DC_DOW is the numeric value of the day of the week. You must sort by this field, but not by the name of the day of the week field (DC_DAY_NAME). Otherwise, the data for Friday would appear first (since it is the first day name alphabetically) and Wednesday would appear last. However, you probably do not want the numeric day of the week value to actually appear on the report, so hide this column in the report.

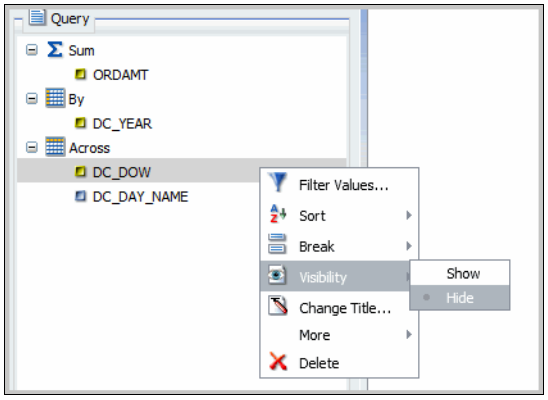


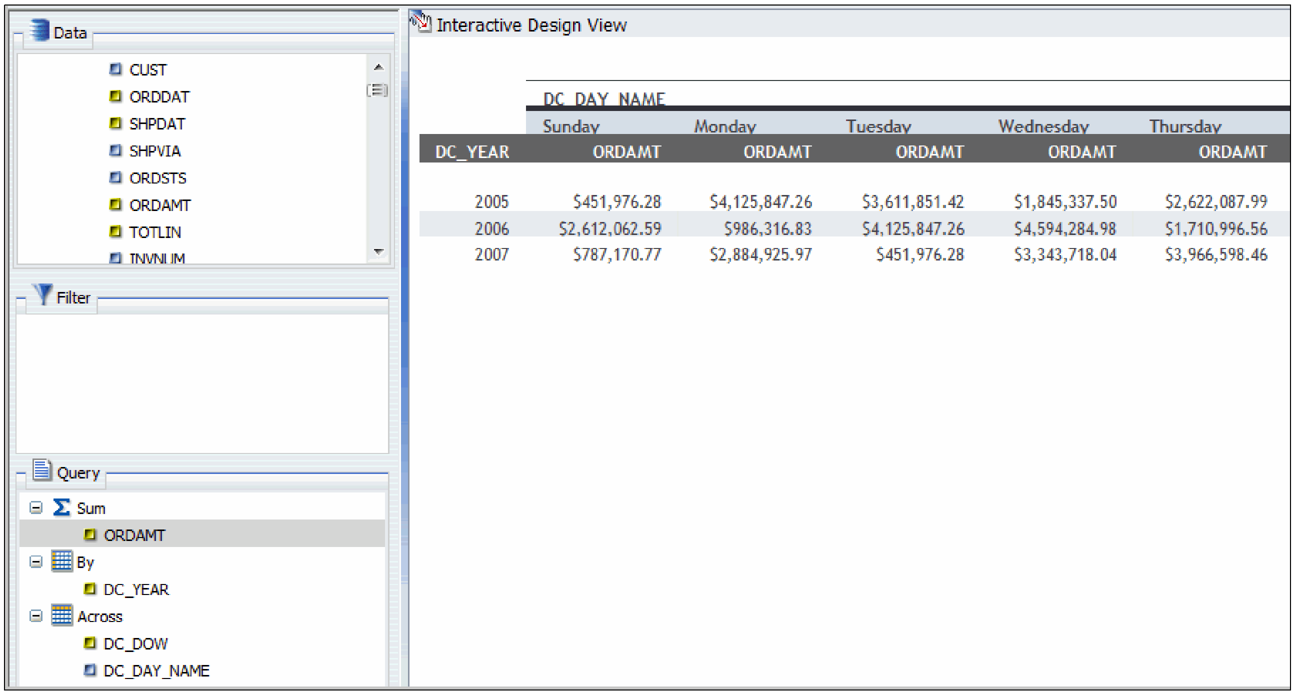
Figure A-24 Hide the DC_DOW column

c. Drag the **ORDDAT.DC_DAY_NAME** field into the Across field container in the Query pane. Make sure it is after **DC_DOW.**

d. Drag the **ORDDAT.ORDAMT** field into the Sum field container in the Query pane and select it. Click the Show field options icon for this field.

e. Add comma inclusion and floating currency formatting to the ORDAMT field in the report.

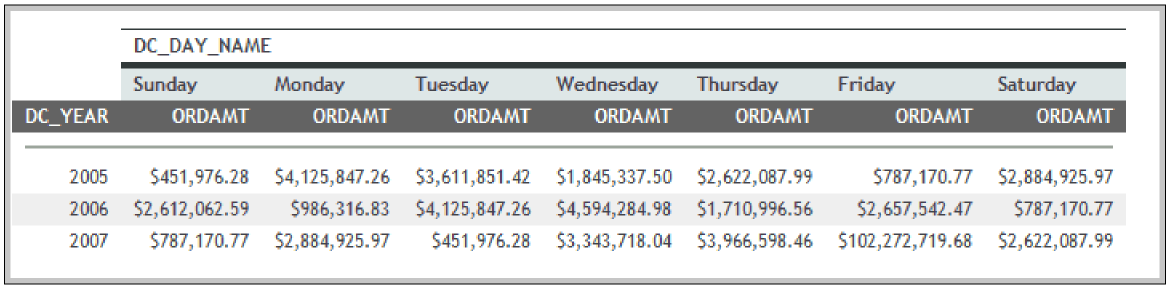
When finished, the report definition should look like the example provided in Figure A-25.



	DC_DAY_NAME				
	Sunday	Monday	Tuesday	Wednesday	Thursday
DC_YEAR	ORDAMT	ORDAMT	ORDAMT	ORDAMT	ORDAMT
2005	$451,976.28	$4,125,847.26	$3,611,851.42	$1,845,337.50	$2,622,087.99
2006	$2,612,062.59	$986,316.83	$4,125,847.26	$4,594,284.98	$1,710,996.56
2007	$787,170.77	$2,884,925.97	$451,976.28	$3,343,718.04	$3,966,598.46

Figure A-25 Report using date conversion table

5. Run the report. The results should look identical to those in Figure A-26.



	DC_DAY_NAME						
	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
DC_YEAR	ORDAMT	ORDAMT	ORDAMT	ORDAMT	ORDAMT	ORDAMT	ORDAMT
2005	$451,976.28	$4,125,847.26	$3,611,851.42	$1,845,337.50	$2,622,087.99	$787,170.77	$2,884,925.97
2006	$2,612,062.59	$986,316.83	$4,125,847.26	$4,594,284.98	$1,710,996.56	$2,657,542.47	$787,170.77
2007	$787,170.77	$2,884,925.97	$451,976.28	$3,343,718.04	$3,966,598.46	$102,272,719.68	$2,622,087.99

Figure A-26 Results of report using date conversion table

Other than the fields and techniques used for date conversion, this is the same report that you created in “Using DB2 Web Query functions to convert to dates” on page 797. Compare the results of those two reports. They should be identical.

6. Save your report as `Date conversion using date conversion table`.

Tip: Now is an opportune time to create indexes over the join columns.

One of the advantages of the date conversion table method is that all of the columns in the table will be brought into the synonyms segment and subsequently will appear in the list of available fields in the development tools. This gives report developers the ability to easily and effortlessly include a wide variety of date formats in your reports.

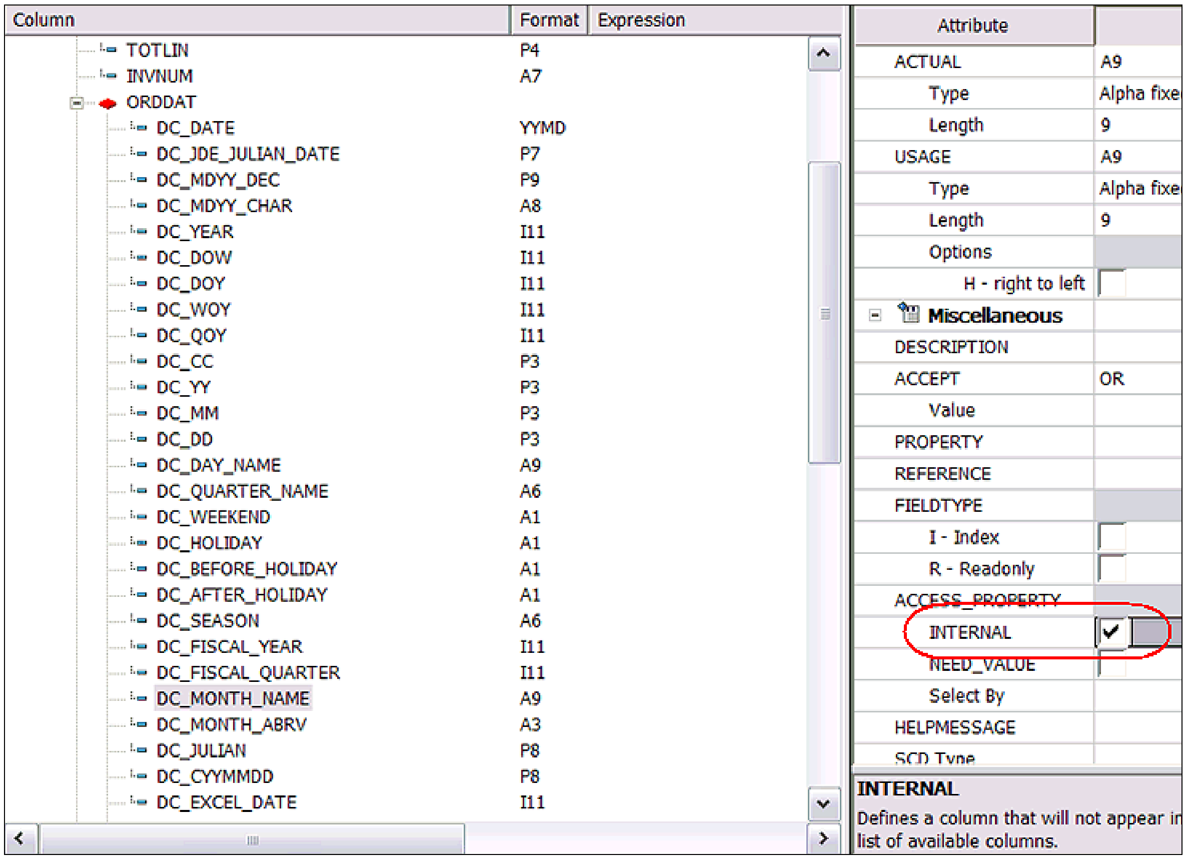
One of the disadvantages is that all of the columns in the table will appear in the list of available fields in the development tools. Why is this a disadvantage? Because there may be many columns in the table and some of them may never actually be used in the reports. These columns can clutter up the list, particularly if the synonym contains many other columns.

Furthermore, and as mentioned previously, if you have multiple legacy date fields in your file and you want to convert them to date fields, you must create join segments to the date conversion table for each one. This adds even more columns to the list of fields in the development tools.

Here are some suggestions for keeping the list of available fields to a manageable size:

- Create the date conversion table with fewer columns (only the ones that your use regularly).
- Create DB2 Web Query business views over your synonyms. This allows you to create and organize a subset of columns to display in the report development tools. See “Defining joins in SQL views” on page 58 for more information about business views.

- Set the column’s Access Property to INTERNAL to hide the infrequently used columns in the synonym. This gives you the ability to preserve the columns in the date conversion table, while not showing them in the development tools. If they are needed for some reports you can simply unhide them by unchecking the INTERNAL setting. Figure A-27 provides an example of how to hide DC_MONTH_NAME so that it does not appear in the list of available columns in the report development tools.



Column	Format	Expression
TOTLIN	P4	
INVNUM	A7	
ORDDAT		
DC_DATE	YYMD	
DC_JDE_JULIAN_DATE	P7	
DC_MDYY_DEC	P9	
DC_MDYY_CHAR	A8	
DC_YEAR	I11	
DC_DOW	I11	
DC_DOY	I11	
DC_WOY	I11	
DC_QOY	I11	
DC_CC	P3	
DC_YY	P3	
DC_MM	P3	
DC_DD	P3	
DC_DAY_NAME	A9	
DC_QUARTER_NAME	A6	
DC_WEEKEND	A1	
DC_HOLIDAY	A1	
DC_BEFORE_HOLIDAY	A1	
DC_AFTER_HOLIDAY	A1	
DC_SEASON	A6	
DC_FISCAL_YEAR	I11	
DC_FISCAL_QUARTER	I11	
DC_MONTH_NAME	A9	
DC_MONTH_ABRV	A3	
DC_JULIAN	P8	
DC_CYYMMDD	P8	
DC_EXCEL_DATE	I11	

Attribute	
ACTUAL	A9
Type	Alpha fixe
Length	9
USAGE	A9
Type	Alpha fixe
Length	9
Options	
H - right to left	☐
Miscellaneous	
DESCRIPTION	
ACCEPT	OR
Value	
PROPERTY	
REFERENCE	
FIELDTYPE	
I - Index	☐
R - Readonly	☐
ACCESS_PROPERTY	
INTERNAL	✓
NEED_VALUE	☐
Select By	
HELPMESSAGE	
SCD Type	

INTERNAL
Defines a column that will not appear in
list of available columns.

Figure A-27 Defining a column that does not appear in list of available columns

Attention: Existing reports that reference hidden columns (those whose INTERNAL setting for ACCESS PROPERTY is selected) will continue to run. However, if you attempt to open such a report in the report development tools, the request will fail with a `Field not found in master file` error. In those cases, you must edit the synonym and remove (uncheck) the INTERNAL setting for ACCESS PROPERTY.

Using SQL Views and functions to convert to dates

A third technique for performing date conversions is to leverage the power of SQL functions and reference them in SQL views. SQL functions such as DATE, YEAR, and DAYOFWEEK can be used in an SQL SELECT statement to efficiently perform the conversions. Depending on the legacy date data type and format, it may be necessary to use other functions such as DIGITS and SUBSTRING. In some cases, this may result in a SELECT statement that is somewhat lengthy. But you can hide this complexity from your report developers by creating an SQL view over this statement and then a DB2 Web Query synonym over the view.

An example SQL view that uses multiple SQL functions for date conversions (over the LEGACY_ORDER_HEADER table) is provided in the QWQCENT library. This view, named ORDER_HEADER_VIEW, is shown in Example A-1.

Example: A-1 ORDER_HEADER_VIEW definition

```
CREATE VIEW ORDER_HEADER_VIEW AS
WITH t1 (
   order ,
   cust ,
   orderdate ,
   shipdate ,
   shpvia ,
   ordsts ,
   ordamt ,
   totlin ,
   invnum ) AS
   (SELECT order, cust,
      DATE(SUBSTRING(DIGITS(orddat),5,4) || '-' || SUBSTRING(DIGITS(orddat),1,2)
         || '-' || SUBSTRING(DIGITS(orddat),3,2)),
      DATE(SUBSTRING(DIGITS(shpdat),5,4) || '-' || SUBSTRING(DIGITS(shpdat),1,2)
         || '-' || SUBSTRING(DIGITS(shpdat),3,2)),
      shpvia, ordsts, ordamt, totlin, invnum
   FROM legacy_order_header )

SELECT order, cust, orderdate, YEAR(orderdate) AS orderyear,
   DAYOFWEEK_ISO(orderdate) AS orderdow,
   CASE
      WHEN DAYOFWEEK_ISO(orderdate) = 1 THEN 'MON'
      WHEN DAYOFWEEK_ISO(orderdate) = 2 THEN 'TUE'
      WHEN DAYOFWEEK_ISO(orderdate) = 3 THEN 'WED'
      WHEN DAYOFWEEK_ISO(orderdate) = 4 THEN 'THU'
      WHEN DAYOFWEEK_ISO(orderdate) = 5 THEN 'FRI'
      WHEN DAYOFWEEK_ISO(orderdate) = 6 THEN 'SAT'
      WHEN DAYOFWEEK_ISO(orderdate) = 7 THEN 'SUN'
      ELSE ''
   END AS orderdayname,
   shipdate, shpvia, ordsts, ordamt, totlin, invnum FROM t1
```

This example performs the following tasks:

- Converts legacy date fields ORDDAT and SHPDAT into true dates and returns them in the derived columns named ORDERDATE AND SHIPDATE.
- Creates a second derived column named ORDERYEAR by referencing the derived column ORDERDATE in the SQL function YEAR. This column returns the year portion of the date.
- Creates a third derived column named ORDERDOW by referencing the derived column ORDERDATE in the SQL function DAYOFWEEK_ISO. This column returns the integer value of the day of the week.
- Creates a fourth derived column named ORDERDAYNAME by referencing the derived column ORDERDATE in the SQL function DAYOFWEEK_ISO and performing some logic in a CASE statement to return the name of the day of the week.

Notice the use of the WITH keyword to create SQL common table expressions. If you are not familiar with common table expressions, they can be thought of as temporary views that only exist during the execution of the query. By using them, you can define a result table with a table-identifier (T1 above) that can be specified as the table name in any FROM clause of the SELECT statement. Because the example uses derived columns that reference other derived columns (ORDERDATE), a common table expression is used. It allows you to define ORDERDATE in one place and reduce the query's complexity by minimizing the amount of text, especially when ORDERDATE is referenced multiple times in the query.

To create the same type of report (as the previous examples) that uses this SQL view, take the following steps:

1. Create a DB2 Web Query synonym over the view. Specify `cen_` for the metadata prefix.
2. Create a new report in Info Assist, using the new view synonym.
   a. Drag the **ORDERYEAR** field into the By field container in the Query pane.
   b. Drag the **ORDERDOW** field into the Across field container in the Query pane. Similar to the report in the previous section, make it an invisible field so that the columns are sorted by the numeric value of the day of the week rather than alphabetically by the name of the day of the week.
   c. Drag the **ORDERDAYNAME** field into the Across field container in the Query pane. Make sure it is after **ORDERDOW.**
   d. Drag the **ORDAMT** field into the Sum field container in the Query pane and select it.
   e. Add comma inclusion and floating currency formatting to the ORDAMT field in the report.
3. When finished, the report definition should look like the example provided in Figure A-28.

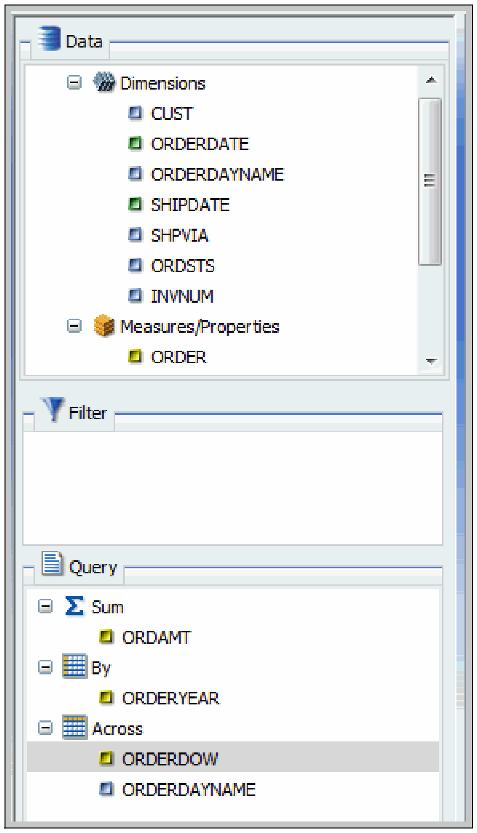
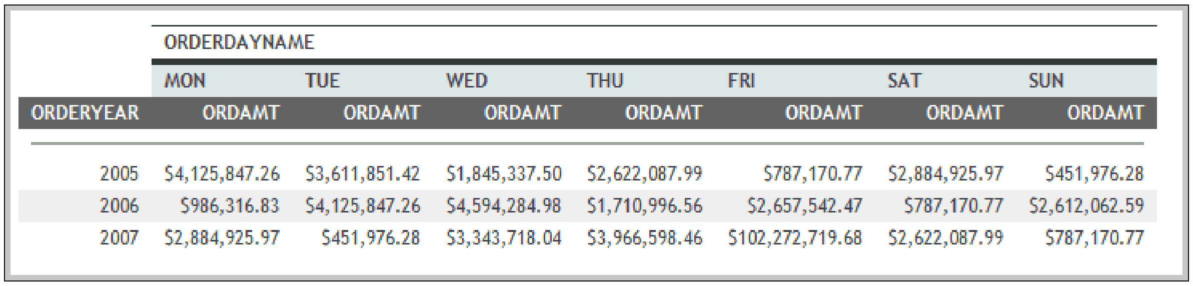


Figure A-28 Report using SQL view for date conversions

4. Run the report. It should look like the example provided in Figure A-29.



	ORDERDAYNAME						
	MON	TUE	WED	THU	FRI	SAT	SUN
ORDERYEAR	ORDAMT	ORDAMT	ORDAMT	ORDAMT	ORDAMT	ORDAMT	ORDAMT
2005	$4,125,847.26	$3,611,851.42	$1,845,337.50	$2,622,087.99	$787,170.77	$2,884,925.97	$451,976.28
2006	$986,316.83	$4,125,847.26	$4,594,284.98	$1,710,996.56	$2,657,542.47	$787,170.77	$2,612,062.59
2007	$2,884,925.97	$451,976.28	$3,343,718.04	$3,966,598.46	$102,272,719.68	$2,622,087.99	$787,170.77

Figure A-29 Results of report using SQL view for date conversions

5. Save your report as `Date conversion using SQL view`.

Comparing the date conversion techniques

At this point, you may be wondering whether it matters what date conversion technique you implement. From a functional standpoint, the answer is no. All three methods will provide reliable date conversion results. However, from a report-performance perspective, the answer is yes. Pushing the date conversion logic down to the database engine does provide significant report performance advantages, but exactly to what degree will always vary, depending on factors such as system resources, system activity, available indexes, the number of rows retrieved, and the number of columns converted. But to give you some idea of what gains can potentially be realized, a mini-benchmark was found to compare the performance results of the two techniques. The benchmark used a 6,000,000 row ORDERS table. The DB2 Web Query reports used were essentially identical to the two that you created in the above date conversion exercise.

Each version of the report was run 20 times, and the run time (the time between when the user selected the report and when the report was displayed to the browser) was recorded and averaged. The results are listed in Table A-7.

Table A-7 Results

Date conversion technique	Average run time (20 executions)
DB2 Web Query functions (BIFs)	1 minute, 55 seconds
Date conversion table	5.2 seconds
SQL functions using SQL. prefix	5.4 seconds
SQL functions in an SQL view	5.2 seconds

The reason why the date conversion table and SQL function techniques perform so much better (than the DB2 Web Query BIF's) is because the conversion logic processing is pushed down to the DB2 engine. However, these are two vastly different approaches:

- Performing database joins to locate the row with the data that is already converted
- Performing the conversions dynamically

If SQE processes the requests, both have the ability to cache the results so that subsequent requests perform better. As you can see in the foregoing benchmark results, the date dimension table performed slightly better, particularly as more conversion functions were added.

In addition, performing the conversion in SQL views and functions may require a fairly elaborate view definition and does require some degree of SQL knowledge to implement. The date conversion table, on the other hand, can be implemented by someone with limited SQL skills.

Performance notwithstanding, the best technique is purely subjective. For simplicity, most report developers would probably prefer to use the DB2 Web Query BIFs. Use of the BIFs may provide acceptable performance in your environment, particularly for smaller databases. Consequently, the suggested approach is to try using the DB2 Web Query BIFs first. If the performance is not satisfactory, then consider using one of the other two techniques.

Attention: IBM and Information Builders development teams are working on ways to improve the performance of the DB2 Web Query built-in functions, particularly those used for date conversion. The goal is to pass as much of that logic to the database engine as possible. DB2 Web Query's ability to internally translate the BIF to the appropriate SQL function would provide the best of both worlds; report developers could use the BIFs for date conversions and deliver a report that performs well.

When deciding between the date conversion table or SQL function/view method, most people would likely find the date conversion table method to be a bit more flexible and easier to implement. Because the information resides in a table that can be viewed and maintained, it is easier for many to conceptualize. Implementing customized columns like fiscal year/quarter and a Super Bowl Sunday flag is also easy because you can manually set these values in the table and view the data. And after all, there is no SUPERBOWLSUNDAY SQL function (not yet anyway), though one could add syntax to the view to join it with another table that has this information.

Converting date fields in Oracle's JD Edwards World application

Oracle's JD Edwards World application has a very large install base on the IBM i platform. The database for this system stores dates in a specific nonstandard format. Consequently, many JD Edwards World customers that use DB2 Web Query must know how to handle this specific type of date conversion.

Date fields in World are typically defined as ZONED (6,0) and are in the format CYYDDD broken down as follows:

- C represents the century indicator:
  - 0 for any date prior to January 2000
  - 1 for any date on or after January 1, 2000
- YY represents the 2-digit year. Dates for the year 2012 would be stored as 12.
- DDD represents the number of the day of the year, counting from January 1. So for August 27, 2012, these digits would contain the value 239.

  For August 27, 2012, the full value for this field is 112239 and for August 27, 1999, it is 009239.

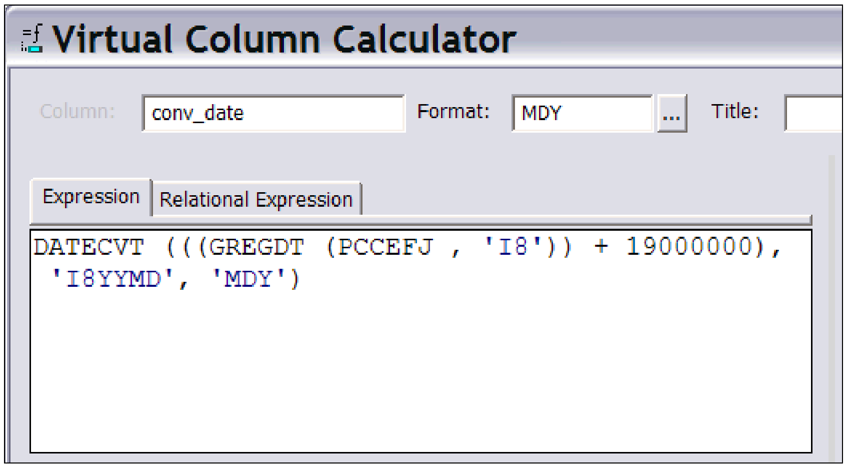
To convert this field to a true DB2 Web Query date, use one of the following (assume that the JDE World file is F43090 and the date field is PCCEFJ):

- DB2 Web Query BIF:

  Use the following expression (where PCCEFJ is the World date field):

  ```
  DATECVT (((GREGDT (PCCEFJ , 'I8')) + 19000000), 'I8YYMD', 'MDY')
  ```

  An example is shown in Figure A-30.



Column: conv_date Format: MDY Title:

```
DATECVT (((GREGDT (PCCEFJ , 'I8')) + 19000000),
 'I8YYMD', 'MDY')
```

Figure A-30 Converting JDE World date

- Date conversion table:

  To implement the date conversion table method for JDE World date fields, create a join from the World file to the date conversion table using the World date field (in this case PCCEFJ) and DC_JDE_JULIAN_DATE as the join columns.

  Figure A-31 shows an example of how to define this join in the DB2 Web Query synonym.

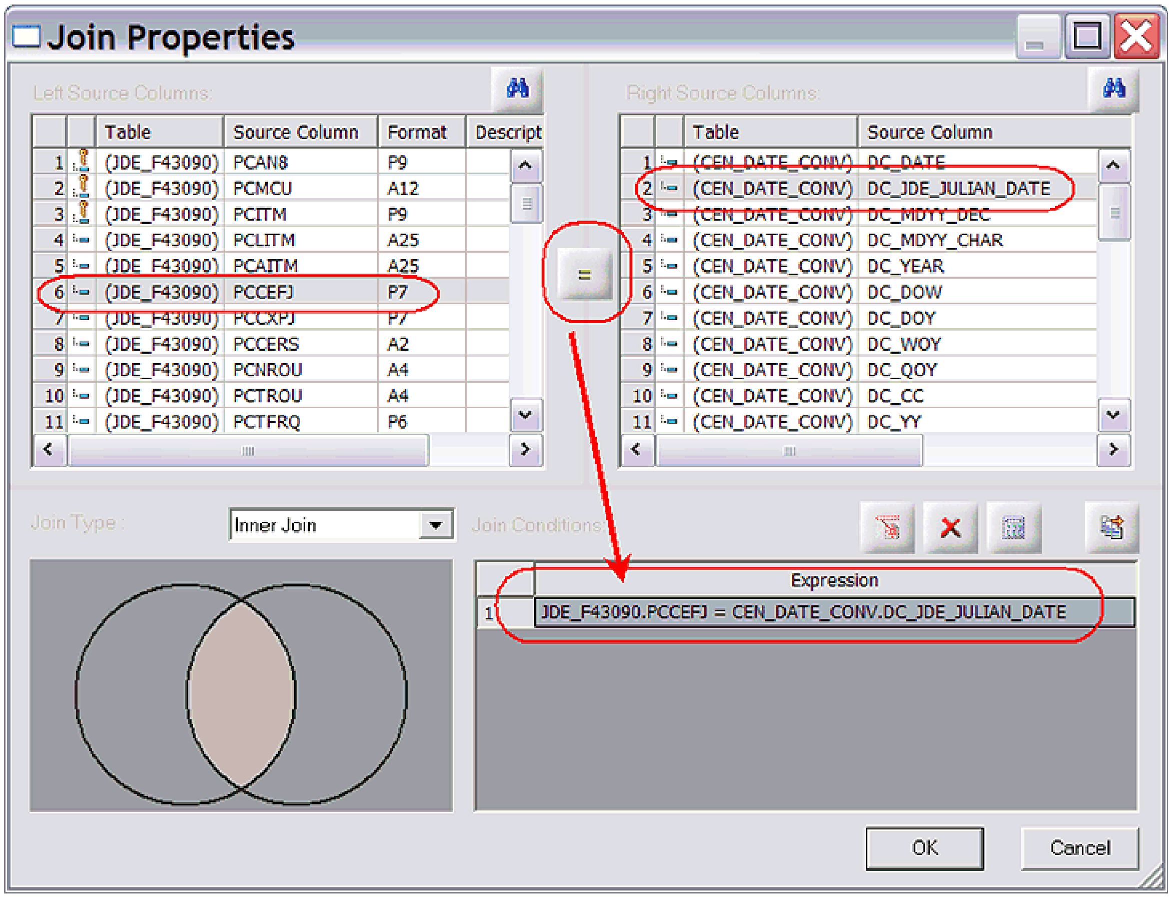


	Table	Source Column	Format	Descript
1	(JDE_F43090)	PCAN8	P9	
2	(JDE_F43090)	PCMCU	A12	
3	(JDE_F43090)	PCITM	P9	
4	(JDE_F43090)	PCLITM	A25	
5	(JDE_F43090)	PCAITM	A25	
6	(JDE_F43090)	PCCEFJ	P7	
7	(JDE_F43090)	PCCXPJ	P7	
8	(JDE_F43090)	PCCERS	A2	
9	(JDE_F43090)	PCNROU	A4	
10	(JDE_F43090)	PCTROU	A4	
11	(JDE_F43090)	PCTFRQ	P6	

	Table	Source Column
1	(CEN_DATE_CONV)	DC_DATE
2	(CEN_DATE_CONV)	DC_JDE_JULIAN_DATE
3	(CEN_DATE_CONV)	DC_MDYY_DEC
4	(CEN_DATE_CONV)	DC_MDYY_CHAR
5	(CEN_DATE_CONV)	DC_YEAR
6	(CEN_DATE_CONV)	DC_DOW
7	(CEN_DATE_CONV)	DC_DOY
8	(CEN_DATE_CONV)	DC_WOY
9	(CEN_DATE_CONV)	DC_QOY
10	(CEN_DATE_CONV)	DC_CC
11	(CEN_DATE_CONV)	DC_YY

Join Type : Inner Join

	Expression
1	JDE_F43090.PCCEFJ = CEN_DATE_CONV.DC_JDE_JULIAN_DATE

Figure A-31 Using date conversion table to convert JDE World date

- SQL functions/view

  If you prefer the SQL functions and view technique, use something like the following to convert the field to a date:

```
SELECT DATE(DIGITS( DECIMAL(pccefj + 1900000 , 7 ,0))) AS conv_date FROM f43090
```

Converting other common legacy date formats

IBM i customers have represented their date fields in many different formats over the years. Some of the more common ones and how to convert them (for each of the three techniques) are provided in Table A-8.

Table A-8 Legacy date conversion table

Example legacy date field name	Legacy data type	Legacy date Format	Legacy example (August 31, 2012)	DB2 Web Query BIF Expression	Date conversion table	SQL function
CH_MDYY	CHAR(8)	MDYY	'08312012'	DATECVT(CH_MDYY, 'A8MDYY', 'MDYY')[a]	Join CH_MDYY to DC_MDYY_CHAR in DATE_CONV table	DATE(SUBSTRING(CH_MDYY,5,4) || '-' || SUBSTRING(CH_MDYY,1,2)|| '-' || SUBSTRING(CH_MDYY,3,2))
CH_YYMD	CHAR(8)	YYMD	'20120831'	DATECVT(CH_YYMD, 'A8YYMD', 'MDYY')[a]	Join CH_YYMD to DC_YYMD_CHAR in DATE_CONV table	DATE(SUBSTRING(CH_YYMD,1,4) || '-' || SUBSTRING(CH_YYMD,5,2)|| '-' || SUBSTRING(CH_YYMD,7,2))
ZN_MDYY	ZONED (8,0)	MDYY	08312012	DATECVT(ZN_MDYY, 'I8MDYY', 'MDYY')[a]	Join ZN_MDYY to DC_MDYY_ZONED in DATE_CONV table	DATE(SUBSTRING(DIGITS(ZN_MDYY),5,4) || '-' || SUBSTRING(DIGITS(ZN_MDYY),1,2) || '-' || SUBSTRING(DIGITS(ZN_MDYY),3,2))
ZN_YYMD	ZONED (8,0)	YYMD	20120831	DATECVT(ZN_YYMD, 'I8YYMD', 'MDYY')[a]	Join ZN_YYMD to DC_YYMD_ZONED in DATE_CONV table	DATE(SUBSTRING(DIGITS(ZN_YYMD),1,4) || '-' || SUBSTRING(DIGITS(ZN_YYMD),5,2) || '-' || SUBSTRING(DIGITS(ZN_YYMD),7,2))
PK_MDYY	PACKED (8,0)	MDYY	08312012	DATECVT(PK_MDYY, 'P8MDYY', 'MDYY')[a]	Join PK_MDYY to DC_MDYY_DEC in DATE_CONV table	DATE(SUBSTRING(DIGITS(PK_MDYY),5,4) || '-' || SUBSTRING(DIGITS(PK_MDYY),1,2) || '-' || SUBSTRING(DIGITS(PK_MDYY),3,2))
PK_YYMD	PACKED (8,0)	YYMD	20120831	DATECVT(PK_YYMD, 'P8YYMD', 'MDYY')[a]	Join PK_YYMD to DC_YYMD_DEC in DATE_CONV table	DATE(SUBSTRING(DIGITS(pk_yymd),1,4) || '-' || SUBSTRING(DIGITS(pk_yymd),5,2) || '-' || SUBSTRING(DIGITS(pk_yymd),7,2))
ZN_JUL5	ZONED (5,0)	YDDD Julian Date	12243	GREGDT(ZN_JUL5, 'I6YMD')	Join ZN_JUL5 to DC_JULIAN in DATE_CONV table	DATE(CASE WHEN INTEGER(SUBSTRING(DIGITS(ZN_JUL5),1,2))>40 THEN '19' ELSE '20' END || SUBSTR(CHAR(ZN_JUL5),1,5))) Note: Year threshold in this example is set to 40
ZN_JUL7	ZONED (7,0)	YYDDD Julian Date	2012243	GREGDT(ZN_JUL7, 'I8YYMD')	Join ZN_JUL to DC_JULIAN in DATE_CONV table	DATE(SUBSTR(CHAR(ZN_JUL),1,7))

Example legacy date field name	**Legacy data type**	**Legacy date Format**	**Legacy example (August 31, 2012)**	**DB2 Web Query BIF Expression**	**Date conversion table**	**SQL function**

`CH_MDY`	`CHAR(6)`	`MDY`	`‘083112’`	`DATECVT(CH_MDY, 'A6MDY', 'MDYY')`	`Join CH_MDY to DC_MDY_CHAR in DATE_CONV table`	`DATE(CASE`
						`WHEN`
						`INTEGER(SUBSTRING(CH_MDY,5,2))`
						`> 40`
						`THEN '19'`
						`ELSE '20'`
						`END`
						`||`
						`SUBSTRING(CH_MDY,5,2) ||`
						`'-' ||`
						`SUBSTRING(CH_MDY,1,2)|| '-'`
						`||`
						`SUBSTRING(CH_MDY,3,2))`
						Note: Year threshold in this example is set to 40.

`CH_YMD`	`CHAR(6)`	`YMD`	`‘120831’`	`DATECVT(CH_YMD, 'A6YMD', 'MDYY')`	`Join CH_YMD to DC_YMD_CHAR in DATE_CONV table`	`DATE(CASE`
						`WHEN`
						`INTEGER(SUBSTRING(CH_YMD,1,2))`
						`> 40 THEN '19'`
						`ELSE '20'`
						`END`
						`||`
						`SUBSTRING(CH_YMD,1,2) ||`
						`'-' ||`
						`SUBSTRING(CH_YMD,3,2)|| '-'`
						`||`
						`SUBSTRING(CH_YMD,5,2))`
						Note: Year threshold in this example is set to 40.

`ZN_MDY`	`ZONED (6,0)`	`MDY`	`083112`	`DATECVT( ZN_MDY, 'P6MDY', 'MDYY' )`	`Join ZN_MDY to DC_MDY_ZONED in DATE_CONV table`	`DATE(CASE`
						`WHEN`
						`INTEGER(SUBSTRING(DIGITS(ZN_MD`
						`Y),5,2)) > 40`
						`THEN '19'`
						`ELSE '20'`
						`END`
						`||`
						`SUBSTRING(DIGITS(ZN_MDY),5,2)`
						`|| '-' ||`
						`SUBSTRING(DIGITS(ZN_MDY),1,2)|`
						`| '-' ||`
						`SUBSTRING(DIGITS(ZN_MDY),3,2))`
						Note: Year threshold in this example is set to 40.

`ZN_YMD`	`ZONED (6,0)`	`YMD`	`120831`	`DATECVT( ZN_MDY, 'P6YMD', 'MDYY' )`	`Join ZN_YMD to DC_YMD_ZONED in DATE_CONV table`	`DATE(CASE`
						`WHEN`
						`INTEGER(SUBSTRING(DIGITS(ZN_YM`
						`D),5,2)) > threshold THEN '19'`
						`ELSE '20'`
						`END`
						`||`
						`SUBSTRING(DIGITS(ZN_YMD),5,2)`
						`|| '-' ||`
						`SUBSTRING(DIGITS(ZN_YMD),1,2)|`
						`| '-' ||`
						`SUBSTRING(DIGITS(ZN_YMD),3,2))`
						Note: Year threshold in this example is set to 40.

Example legacy date field name	Legacy data type	Legacy date Format	Legacy example (August 31, 2012)	DB2 Web Query BIF Expression	Date conversion table	SQL function
PK_MDY	PACKED (6,0)	MDY	083112	DATECVT(PK_MDY, 'P6MDY', 'MDYY')	Join PK_MDY to DC_MDY_DEC in DATE_CONV table	DATE(CASE WHEN INTEGER(SUBSTRING(DIGITS(PK_MD Y),5,2)) > 40 THEN '19' ELSE '20' END || SUBSTRING(DIGITS(PK_MDY),5,2) || '-' || SUBSTRING(DIGITS(PK_MDY),1,2)| | '-' || SUBSTRING(DIGITS(PK_MDY),3,2)) Note: Year threshold in this example is set to 40.
PK_YMD	ZONED (6,0)	YMD	120831	DATECVT(PK_YMD, 'P6YMD', 'MDYY')	Join PK_YMD to DC_YMD_DEC in DATE_CONV table	DATE(CASE WHEN INTEGER(SUBSTRING(DIGITS(PK_YM D),5,2)) > threshold THEN '19' ELSE '20' END || SUBSTRING(DIGITS(PK_YMD),5,2) || '-' || SUBSTRING(DIGITS(PK_YMD),1,2)| | '-' || SUBSTRING(DIGITS(PK_YMD),3,2)) Note: Year threshold in this example is set to 40.
ZN_CEN ZN_YEAR ZN_MONTH ZN_DAY (Each component of the date is stored in a separate zoned decimal field.)	ZONED (2,0) ZONED (2,0) ZONED (2,0) ZONED (2,0)	CC YY MM DD	20 12 08 31	DATECVT(((ZN_CEN * 1000000) + (ZN_YEAR * 10000) + (ZN_MONTH * 100) + ZN_DAY), 'I8YYMD', 'YYMD')	Join legacy date fields to the following columns in DATE_CONV table: DC_CC DC_YY DC_MM DC_DD	DATE(DIGITS(ZN_CEN) || DIGITS(ZN_YEAR) || '-' || DIGITS(ZN_MONTH)|| '-' || DIGITS(ZN_DAY))
CH_CEN CH_YEAR CH_MONTH CH_DAY (Each component of the date is stored in a separate character field.)	CHAR(2) CHAR(2) CHAR(2) CHAR(2)	CC YY MM DD	20 12 08 31	DATECVT((CH_CEN || CHYEAR || CH_MONTH || CH_DAY), 'A8YYMD', 'YYMD')	Join legacy date fields to the following columns in DATE_CONV table: DC_CC_CHAR DC_YY_CHAR DC_MM_CHAR DC_DD_CHAR	DATE(CH_CEN || CH_YEAR || ‘-’ || CH_MONTH|| ‘-’ || CH_DAY)

a. Denotes that the DB2 Web Query function is translated to SQL for optimal performance

DB2 Web Query date built-in functions

In this section, we present a few of the more common DB2 Web Query built-in functions that you can do with dates. These functions are described in more detail in the help text included with Developer Workbench.

DPART: Extracting a date component from a date field

The DPART function extracts a specified component from a date field and returns it in numeric format.

The format for DPART is as follows:

```
DPART (datevalue, 'component', output)
```

- datevalue (Date) - A full component date.
- component (Alphanumeric) - The name of the component to be retrieved enclosed in single quotation marks. Valid values are as follows:
  - YEAR, YY - Returns the year component
  - MONTH, MM - Returns the month component
  - DAY, DAY-OF-MONTH, DD - Returns day of month
  - QUARTER, QQ - Returns quarter
- output (Integer) - The field that contains the result, or the integer format of the output value enclosed in single quotation marks.

Example assuming SHIPDATE column contains the date value 2012/09/22:

```
DPART(SHIPDATE, 'DD', 'I2')
```

This returns 22.

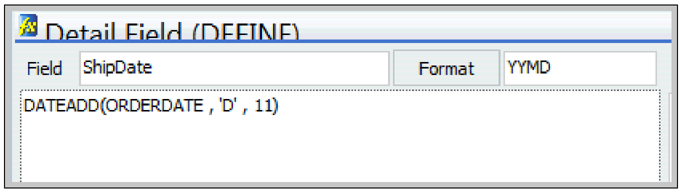
Note: The use of DPART results in the generation of an SQL statement to submit to the DB2 engine (to carry out the date arithmetic). This is important because it will result in better query performance. In fact for this very reason, the Date Decompose feature in the Metadata Editor uses DPART to break up the date components.

DATEADD: Adding or subtracting a date unit to or from a date

You can add or subtract years, months, days, weekdays, or business days from your date. Business days can take a holiday file as input. By default, a business day and a weekday are the same concept. In the following function, the date field must have a format like YYMD, MDY, or JUL. Increment must be an integer.

```
DATEADD(date, ‘Y/M/D/WD/BD’, increment)
```

Figure A-32 demonstrates adding 11 days ('D') to the order date to calculate the ship date:



Field: ShipDate
Format: YYMD

```
DATEADD(ORDERDATE , 'D' , 11)
```

Figure A-32 DATEADD

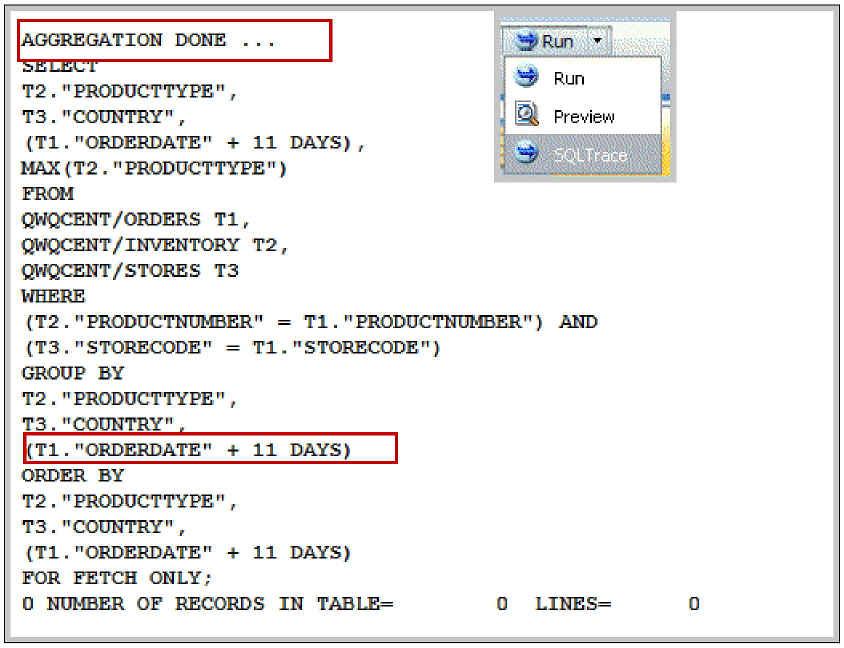
The DATEADD function is not required if you simply want to add months or days to a date. To add months, your result field must be in a format similar to 'YYM' or 'MY'. To add days, your result field must contain days, for example, YYMD.

If you want to add 11 days to ORDERDATE in the previous example and do not need to worry about business or weekdays, you can replace the DATEADD with the following values:

```
ORDERDATE + 11
```

This creates a SHIPDATE of 11 days in the future.

Note: In many cases, the use DATEADD will result in the generation of an SQL statement to submit to the DB2 engine (to carry out the date arithmetic). This is important because it will result in better query performance. Some components such as 'BD' (Business Days) have no SQL equivalent, and thus can not be translated. You can verify SQL translation by requesting a Run With SQL trace request within Info Assist. A translated request would look similar to the one shown in Figure A-33.

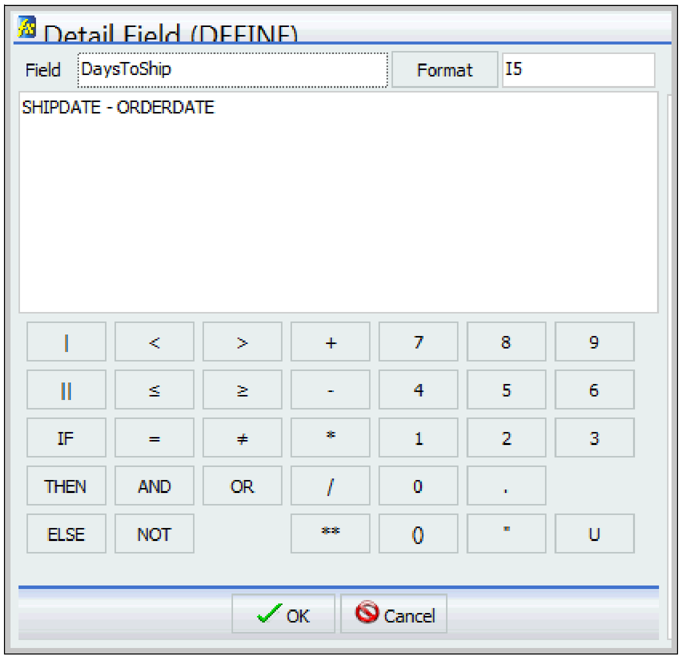


```
AGGREGATION DONE ...
SELECT
T2."PRODUCTTYPE",
T3."COUNTRY",
(T1."ORDERDATE" + 11 DAYS),
MAX(T2."PRODUCTTYPE")
FROM
QWQCENT/ORDERS T1,
QWQCENT/INVENTORY T2,
QWQCENT/STORES T3
WHERE
(T2."PRODUCTNUMBER" = T1."PRODUCTNUMBER") AND
(T3."STORECODE" = T1."STORECODE")
GROUP BY
T2."PRODUCTTYPE",
T3."COUNTRY",
(T1."ORDERDATE" + 11 DAYS)
ORDER BY
T2."PRODUCTTYPE",
T3."COUNTRY",
(T1."ORDERDATE" + 11 DAYS)
FOR FETCH ONLY;
0 NUMBER OF RECORDS IN TABLE=          0  LINES=        0
```

Figure A-33 Translated DATEADD to SQL request

DATEDIF: Calculating the difference between two dates

A similar function to DATEADD is DATEDIF. This function allows you to find the difference between two dates in terms of years, months, days, weekdays, and business days. If you only want to find the difference in months or days, you do not need to use DATEDIF. You can create a Define field that simply subtracts two dates, as shown in Figure A-34.



Field: DaysToShip
Format: I5

```
SHIPDATE - ORDERDATE
```

Figure A-34 Simple subtraction of two date fields

```
ORDERDATE - SHIPDATE
```

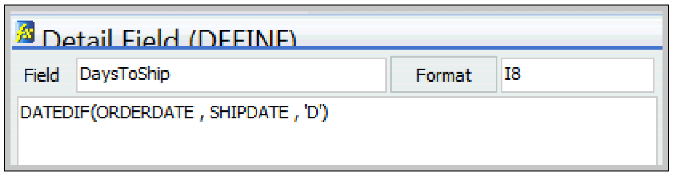
The resulting value is based on the least significant component in the order and ship dates. For example, if one of the dates is defined as YYM, then your results are in months. If both dates are defined down to the day level, then your results are in days.

The DATEDIF function provides the same capability but allows the developer to provide a specific unit component for the output. The format for DATEDIF is as follows:

```
DATEDIF('from_date', 'to_date', 'component')
```

- from_date (Date) - The start date from which to calculate the difference. Can be a field or a constant, for example, ‘20120101’.
- to_date (Date) - The end date from which to calculate the difference. Can be a field or a constant, for example, ‘20120922’.
- component (Alphanumeric). The unit of the value returned by the function. Can be one of the following (enclosed in single quotation marks):
  - Y indicates a year unit.
  - M indicates a month unit.
  - D indicates a day unit.
  - WD indicates a weekday unit.
  - BD indicates a business day unit.

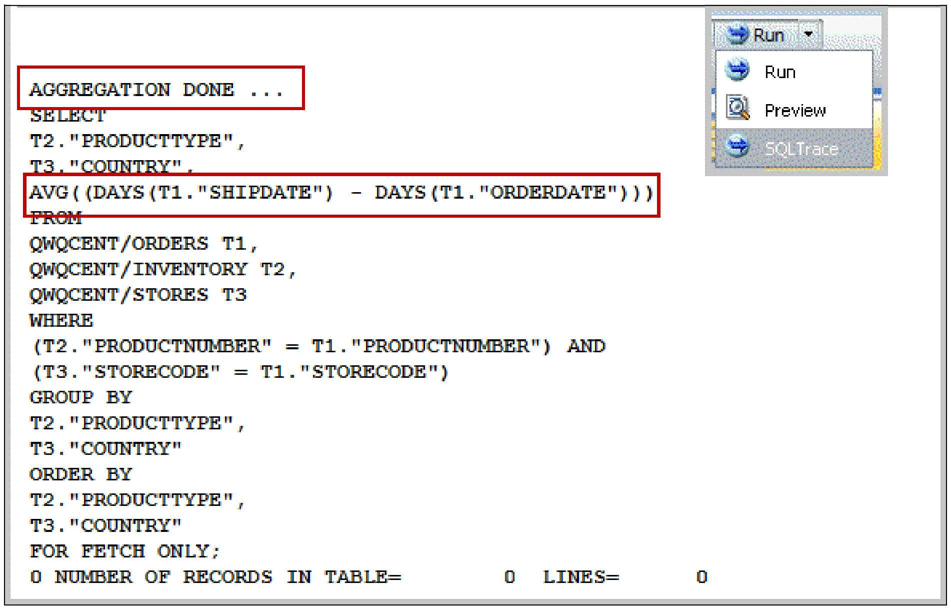
When using DATEDIF, the format of the Define field must be defined as an I8 field as shown in Figure A-35:



```
Field   DaysToShip        Format   I8
DATEDIF(ORDERDATE , SHIPDATE , 'D')
```

Figure A-35 DATEDIF

Note: Once again, for query performance considerations, it is important to know that both techniques described in this section will result in the generation of an SQL statement to submit to the DB2 engine (to carry out the date arithmetic). You can verify this by requesting a Run With SQL trace request within Info Assist. You should see a result similar to the one shown in Figure A-36.



```
AGGREGATION DONE ...
SELECT
T2."PRODUCTTYPE",
T3."COUNTRY",
AVG((DAYS(T1."SHIPDATE") - DAYS(T1."ORDERDATE")))
FROM
QWQCENT/ORDERS T1,
QWQCENT/INVENTORY T2,
QWQCENT/STORES T3
WHERE
(T2."PRODUCTNUMBER" = T1."PRODUCTNUMBER") AND
(T3."STORECODE" = T1."STORECODE")
GROUP BY
T2."PRODUCTTYPE",
T3."COUNTRY"
ORDER BY
T2."PRODUCTTYPE",
T3."COUNTRY"
FOR FETCH ONLY;
0 NUMBER OF RECORDS IN TABLE=        0  LINES=       0
```

Figure A-36 Example of date arithmetic passed to SQL and DB2 engine

DATEMOV: Moving the date to a significant point

DATEMOV moves your date field to a significant point such as the end of the week or the beginning or the quarter. Table A-9 lists the possible values for move-point.

```
DATEMOV(date, ‘move-point’)
```

The date field must be a full date, for example, MDYY or YYJUL.

Note: The DATEMOV function is NOT translated to SQL. This can result in less than optimal performance because the DB2 Web Query reporting server engine (and not the DB2 for i engine) processes the request.

Table A-9 Values for move-point

Move-point	Meaning
EOM	End of month
BOM	Beginning of month
EOQ	End of quarter
BOQ	Beginning of quarter
EOY	End of year
BOY	Beginning of year
EOW	End of week
BOW	Beginning of week
NWD	Next weekday
NBD	Next business day
PWD	Prior weekday
PBD	Prior business day
WD-	Current weekday or prior weekday (if weekend)
BD-	Current business day or prior business day if current is a non-business day
WD+	Current weekday or next weekday (if weekend)
BD+	Current business day or next business day if current is a non-business day

Example: Dynamic Date Range report

In this section, you create a report that gives the user the ability to select a dynamic range of rows based on two input parameters:

- An input date
- A number of days

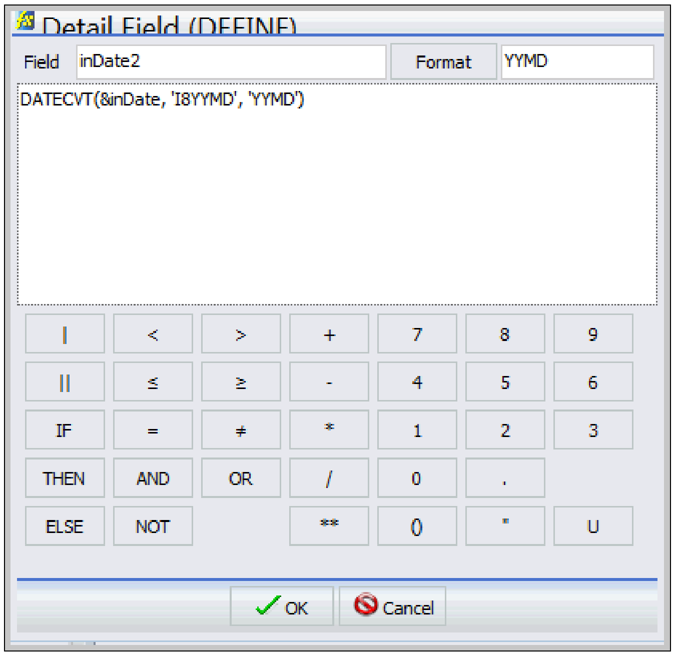
The report will calculate a date range based on the two input fields. The date range is defined as follows:

- beginRange: the specified number of days before the input date
- endRange: the specified number of days after the input date

The report will then return rows in which the order date within is the specified date range. For example, if the user entered 20120915 for the date and 10 for the number of days, the report would return rows on or between September 5, 2012, and September 25, 2012.

To create this report, take the following steps:

1. Create a new report using Info Assist and select **cen_orders** at the data source.
2. Create a DEFINE field named `inDate2`. The format is YYMD and the expression is DATECVT(&inDate, 'I8YYMD', 'YYMD'). An example is shown in Figure A-37.



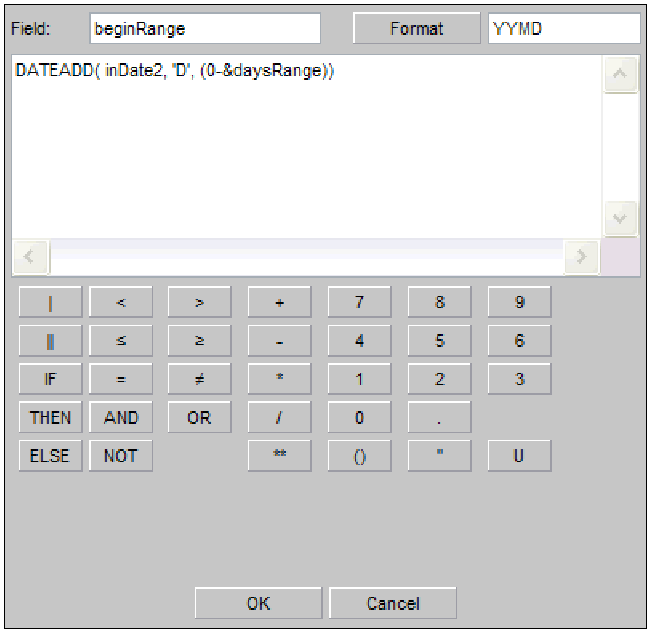
DATECVT(&inDate, 'I8YYMD', 'YYMD')

Figure A-37 DEFINE field for inDate2

This Define field automatically creates the prompt for an input parameter named &inDate (when the report is run). This is an example of an implicit input parameter. It then converts the input parameter to a true date field.

Attention: Do *not* create an input parameter for &inDate in the Selection criteria tab.

3. Create a DEFINE field named `beginRange`. The format is YYMD and the expression is DATEADD(inDate2, 'D', (0-&daysRange)), as shown in Figure A-38.

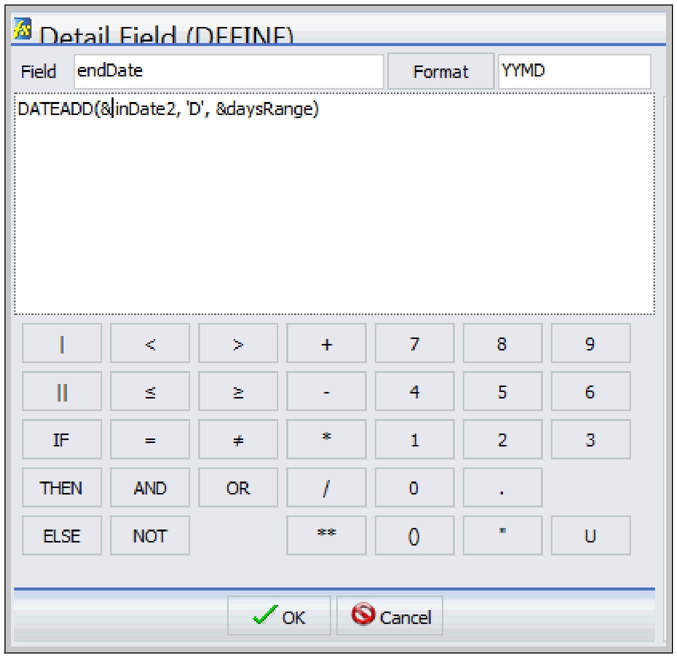


DATEADD(inDate2, 'D', (0-&daysRange))

Figure A-38 DEFINE field beginRange

As in the foregoing example, this Define field will create the prompt for an input parameter named &daysRange (when the report is run). Again, you do *not* create an explicit input parameter for &daysRange. The DEFINE field will return a date that is the specified number of days less than the specified input date.

4. Create a DEFINE field named `endRange`. The format is YYMD and the expression is DATEADD(inDate2, 'D', &daysRange), as shown in Figure A-39.



Field: endDate
Format: YYMD

```
DATEADD(&inDate2, 'D', &daysRange)
```

Figure A-39 DEFINE field endRange

This DEFINE field will return a date that is the specified number of days greater than the specified input date.

Important: The order in which you create these DEFINE fields is very important. This is because the second and third DEFINE fields contain expressions that reference the first DEFINE field. If you define them out of order, the expression will fail because it will be unable to find and resolve DEFINE field inDate2.

5. Click the **Data** tab to and select Advanced Filter icon as shown in Figure A-40.

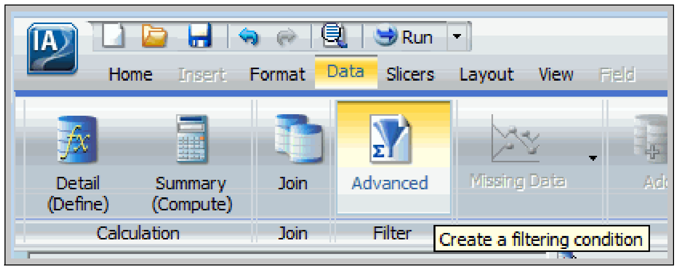
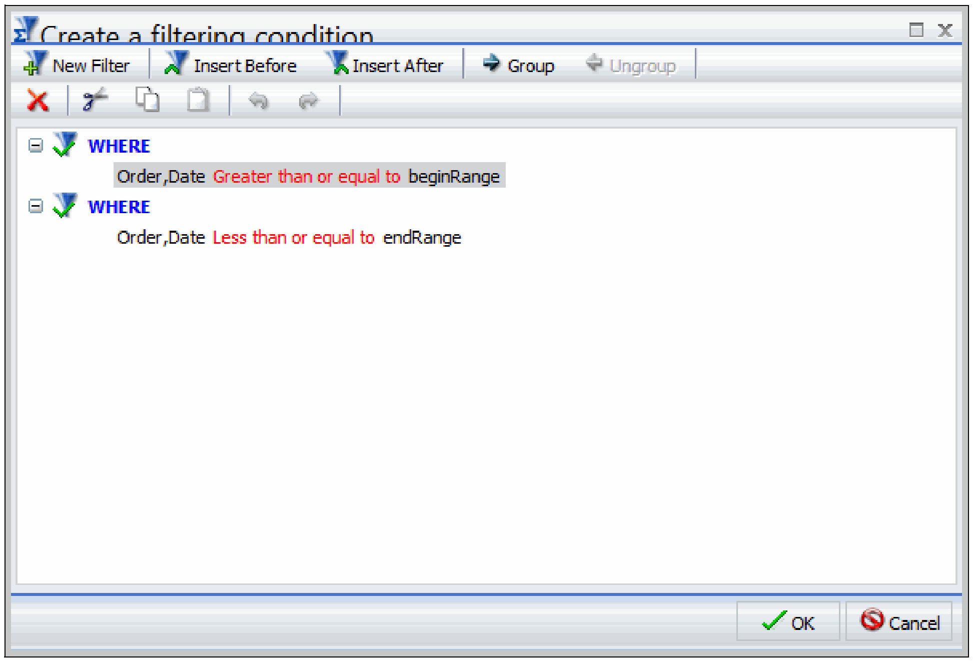


Figure A-40 Advanced Filter

6. Create two screening conditions for the ORDERDATE column:
   - The first condition is **GREATER THAN OR EQUAL TO** the beginRange DEFINE field
   - The second condition is **LESS THAN OR EQUAL TO** the endRange DEFNE field.

   Select **WHERE** for both conditions. An example is shown in Figure A-41.

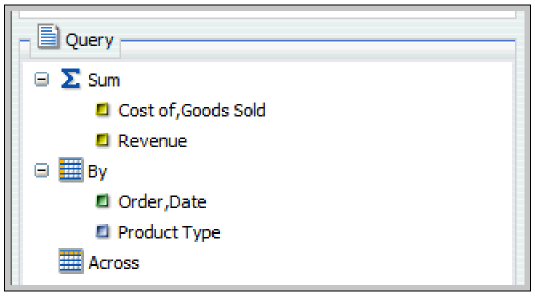


WHERE
Order,Date Greater than or equal to beginRange
WHERE
Order,Date Less than or equal to endRange

Figure A-41 Dynamic Date Range Report selection criteria

7. Return to the Field Selection tab and complete the report by selecting the following Sort-By and Sum report columns:
   - Sort-By: **ORDERDATE**, **PRODUCTTYPE**
   - Sum: **COSTOFGOODSSOLD**, **REVENUE**

   An example is provided in Figure A-42.

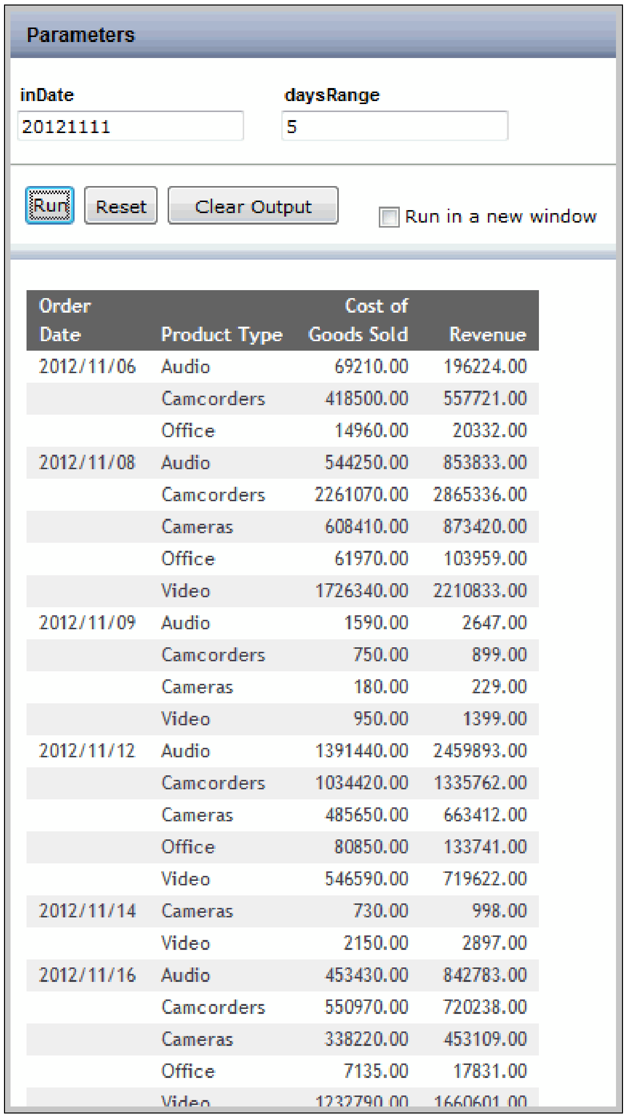


Query
Sum
Cost of,Goods Sold
Revenue
By
Order,Date
Product Type
Across

Figure A-42 Sort-By and Sum columns

8. Save your report as `Dynamic Date Range`.
9. Click **Quit** to return to the DB2 Web Query launch page.
10. Run the new Dynamic Date Range report.
11. Specify `20121111` for the inDate parameter and `5` for daysRange.

12. Click **Run**. The report returns rows for orders between 11/06/2012 and 11/16/2012, as shown in Figure A-43.



inDate: 20121111

daysRange: 5

Order Date	Product Type	Cost of Goods Sold	Revenue
2012/11/06	Audio	69210.00	196224.00
	Camcorders	418500.00	557721.00
	Office	14960.00	20332.00
2012/11/08	Audio	544250.00	853833.00
	Camcorders	2261070.00	2865336.00
	Cameras	608410.00	873420.00
	Office	61970.00	103959.00
	Video	1726340.00	2210833.00
2012/11/09	Audio	1590.00	2647.00
	Camcorders	750.00	899.00
	Cameras	180.00	229.00
	Video	950.00	1399.00
2012/11/12	Audio	1391440.00	2459893.00
	Camcorders	1034420.00	1335762.00
	Cameras	485650.00	663412.00
	Office	80850.00	133741.00
	Video	546590.00	719622.00
2012/11/14	Cameras	730.00	998.00
	Video	2150.00	2897.00
2012/11/16	Audio	453430.00	842783.00
	Camcorders	550970.00	720238.00
	Cameras	338220.00	453109.00
	Office	7135.00	17831.00
	Video	1232790.00	1660601.00

Figure A-43 Dynamic Date Range report results

B

DB2 Web Query system variables

System variables are predefined and automatically supplied by the system when a procedure references them. System variables have names that begin with a single ampersand (&). Examples of these variables are &FOCFEXNAME, which indicates the name of the report that is running, and &DATE, which indicates the current date.

Table B-1 lists the system variables available in DB2 Web Query.

Table B-1 System variables

Variable	Format or value	Description
`&DATE`	`MM/DD/YY`	Returns the current date.
`&DATE`*`fmt`*	`Any date format.`	Returns the current date, where *fmt* can be any valid date format. Because many date format options can be appended to the prefix *DATE* to form one of these variable names, you should avoid using DATE as the prefix when creating a variable name.
`&DMY`	`DDMMYY`	Returns the current date.
`&DMYY`	`DDMMCCYY`	Returns the current (four-digit year) date.
`&FOCCODEPAGE`	`Any integer value`	Returns the code page being used by the server.
`&FOCFEXNAME`		Returns the name of the FOCEXEC running. This variable differs from the &FOCFOCEXEC variable because &FOCFOCEXEC returns the name of the calling FOCEXEC only.

`&FOCFIELDNAME`	`NEW`
	`OLD`
	`NOTRUNC`
		Returns a string indicating whether long and qualified field names are supported. A value of OLD means that they are not supported; NEW means that they are supported; and NOTRUNC means that they are supported, but unique truncations of field names cannot be used.

Variable	Format or value	Description
`&FOCFOCEXEC`		Returns the fully qualified path for the procedure.
`&FOCMODE`	`AS400`	Identifies the operating environment.
`&FOCNEXTPAGE`	`0`	Variable whose value is determined by the last page number used by the last report. Its value is one more than the last page number used in the last report.
`&FOCQUALCHAR`	`. : ! % | \`	Returns the character used to separate the components of qualified field names.
`&FOCREL`	`release number`	Identifies the FOCUS Release number (for example, 6.5 or 6.8).
`&FOCSBORDER`	`ON` `OFF`	Whether solid borders are used in full-screen mode.
`&FOCUSER`		Returns the connected user ID. Similar to the GETUSER function.
`&MR_FULL_FEXNAME`		Returns the full FEX name. This is the same name that appears in the DB2 Web Query Managed Reporting (MR) interface from the web browser.
`&MDY`	`MMDDYY`	Returns the current date.
`&MDYY`	`MMDDCCYY`	Returns the current (four-digit year) date.
`&TOD`	`HH.MM.SS`	Returns the current time.
`&YMD`	`YYMMDD`	Returns the current date.
`&YYMD`	`CCYYMMDD`	Returns the current (four-digit year) date.

C

Change management considerations

Change management is the process of exporting DB2 Web Query resources from a source system and importing that content to a target DB2 Web Query environment. This appendix describes the export/import process using the provided change management utilities.

Change Management overview

To run Change Management, the following privileges are needed:

- Export:
  - Administrator: User who is in the WebQueryAdministrator group
  - Developer: User who is in the folder-dev group. This user will be able to add repository objects but will not be able to add metadata to a change management package. The Reporting Server node (from which metadata is selected) is only visible to Administrators.
- Import:
  - Administrator: User who is in the WebQueryAdministrator group.

Note: The target system must be running the same version of DB2 Web Query as the source system.

Using the Change Management utility, these types of resources in Web Query can be exported and imported:

- Repository objects within the top level folder:
  - Subfolders
  - Procedures (Reports, Charts, Dashboards, Documents)
  - HTML
  - Images
  - Stylesheets
  - Schedules
  - Distribution Lists
- Synonyms
  - .MAS (master files)
  - .ACX (access files

Note: A developer is only allowed to export repository objects because a developer does not have access to the Reporting Server node.

Change Management scenarios

The following scenarios describe where the Change Management utility can be leveraged:

1. Application development/test/delivery:

   Developers periodically make revisions to DB2 Web Query content and move those changes to a test environment for testing and user feedback. Once those changes are tested and approved, it is promoted to the production environment. This process could be implemented by Change Management.

2. ISV bundling an application:

   ISV develops applications and sells them to customers. Change Management allows them to export the applications as a package and then move the package to the system on the customer site. After importing the package, the applications are deployed on the target system.

3. Backup and restore:

   To prevent applications from being lost or badly changed, the applications should be backed up and restored periodically. The administrator can export applications to packages for backup purposes via Change Management and then restore them to another system or locally by importing the packages if recovery is needed.

Exporting a change management package

In the Change Management Export facility, a developer or administrator selects the objects to export in order to build a change management package. This package is a folder on the Integrated File System (IFS) which is deployed to the target environment to be imported. The template or manifest of this package is referred to as a scenario; it contains a list of all the objects to be included in the package. Therefore, the first step in the export process is to define the scenario.

Take the following steps to create a scenario:

1. Sign in to DB2 Web Query as an administrator or developer.
2. From the Resource Tree of the BI portal, left click **Change Management** to expand Change Management tree. Right-click **Export** and select **New Scenario** as in Figure C-1.

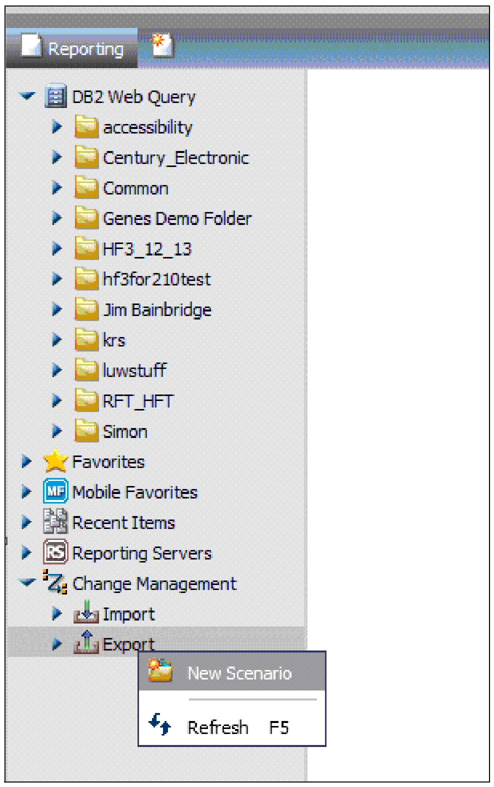
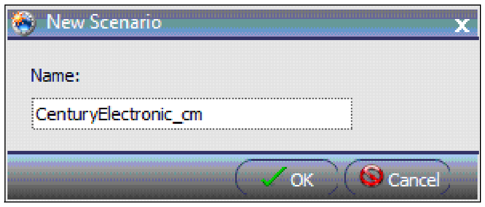


Figure C-1 Create New Scenario

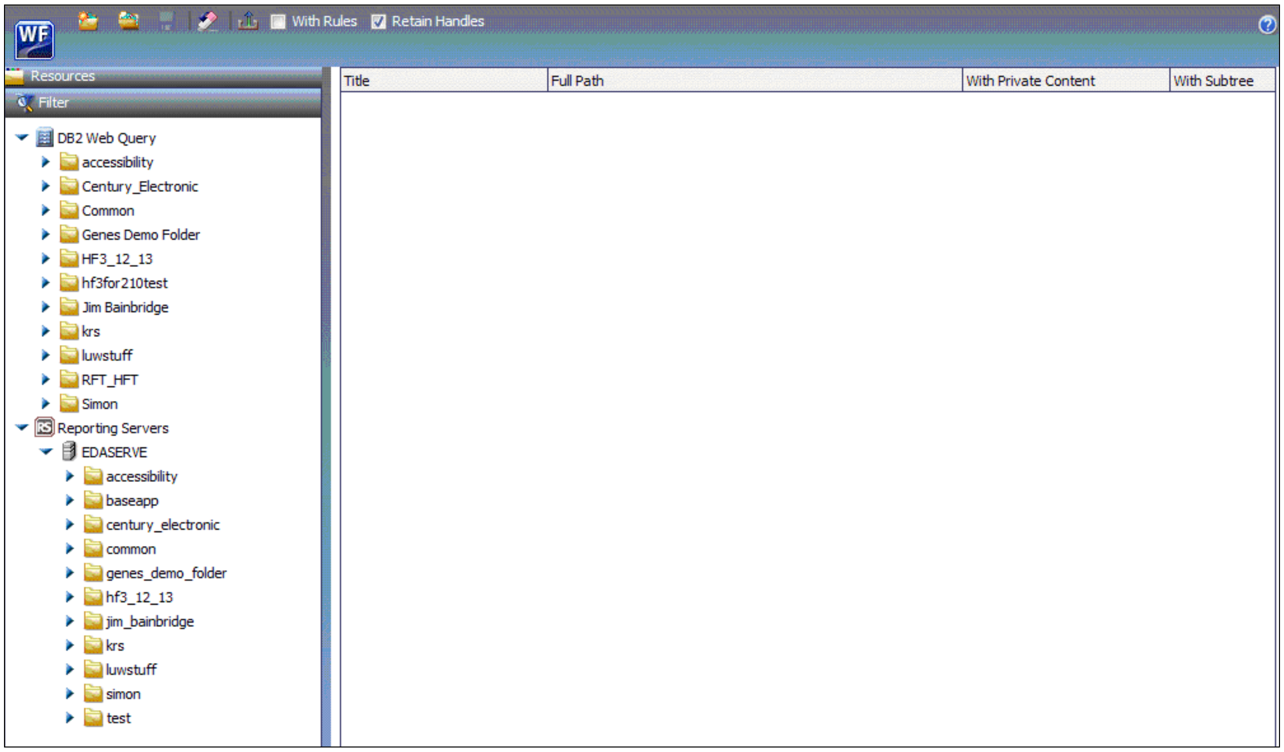
3. The New Scenario window appears. Type the scenario name **CenturyElectronic_cm** as shown in Figure 26-10 and then click **OK**.



Name: CenturyElectronic_cm

Figure 26-10 New Scenario

4. The Change Management export utility is presented as shown in Figure 26-11. From this interface, you select the resources (both metadata and reporting objects) to export to the target system. Because this is a new scenario, the list is initially empty.



- DB2 Web Query
  - accessibility
  - Century_Electronic
  - Common
  - Genes Demo Folder
  - HF3_12_13
  - hf3for210test
  - Jim Bainbridge
  - krs
  - luwstuff
  - RFT_HFT
  - Simon
- Reporting Servers
  - EDASERVE
    - accessibility
    - baseapp
    - century_electronic
    - common
    - genes_demo_folder
    - hf3_12_13
    - jim_bainbridge
    - krs
    - luwstuff
    - simon
    - test

Figure 26-11 Export Scenario

Note: If you do not see the Reporting Server node, this means you are not logged in as an Administrator. The Reporting Server node (from which metadata is selected) is only visible to Administrators.

Notice that in the resource tree there are two nodes from which you can expand and select elements:

– DB2 Web Query:

Contains objects from the DB2 Web Query repository such as reports, charts, dashboards, compound documents, and html files. You can select elements to export at the top level folder, subfolder, or individual repository object level.

– Reporting Servers:

Contains the metadata (synonyms). You can select metadata elements to export at the application folder level or by selecting individual synonyms.

There are two ways to select resources from the resource tree and add them to the scenario:

– Right-click the object and choose either **Select With Subtree** or **Select Folder Only**.
  - **Select With Subtree**: Selects the folder and all the content in the folder (including all subfolders and their contents).
  - **Select Folder Only**: Selects the folder with no content.

– Drag and drop the objects from the tree to the right panel. If you drag a folder to the panel, the Select with Subtree action is taken.

If Private content is selected, the **With Private Content** box is automatically selected. If a Published folder containing private content is selected, there is an option to include private content by checking the **With Private Content** box in the scenario listing. If checked, all of the private content in that folder and its subfolders will be exported.

Notes:

- If **With Private Content** box is selected, the private content will only be imported if the owner of that private content already exists in the target environment.
- If a subfolder is selected, its parent folder must exist in the target system.

– Also notice the following two settings at the top of the scenario interface as shown in Figure 26-12:
  - **With Rules**. This option is unselected by default and it should NOT be selected.
  - **Retain Handles**. This option should be selected if the source Web Query 2.1.0 environment was migrated from 1.1.1 or 1.1.2. During the migration, the 1.1.* version Hrefs are used as the 2.1.0 version handles. Moving these handles to the target environment will allow codes that contain the earlier style to continue to work.

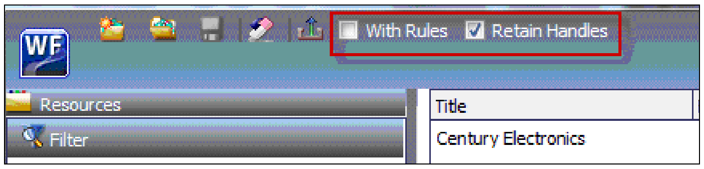
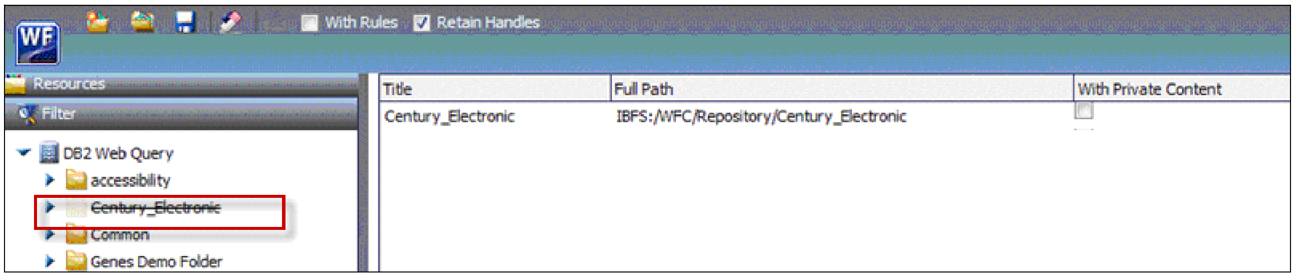


Figure 26-12 With Rules

5. Drag and drop the top level folder **Century_Electronic** in **DB2 Web Query** tree into the right panel.

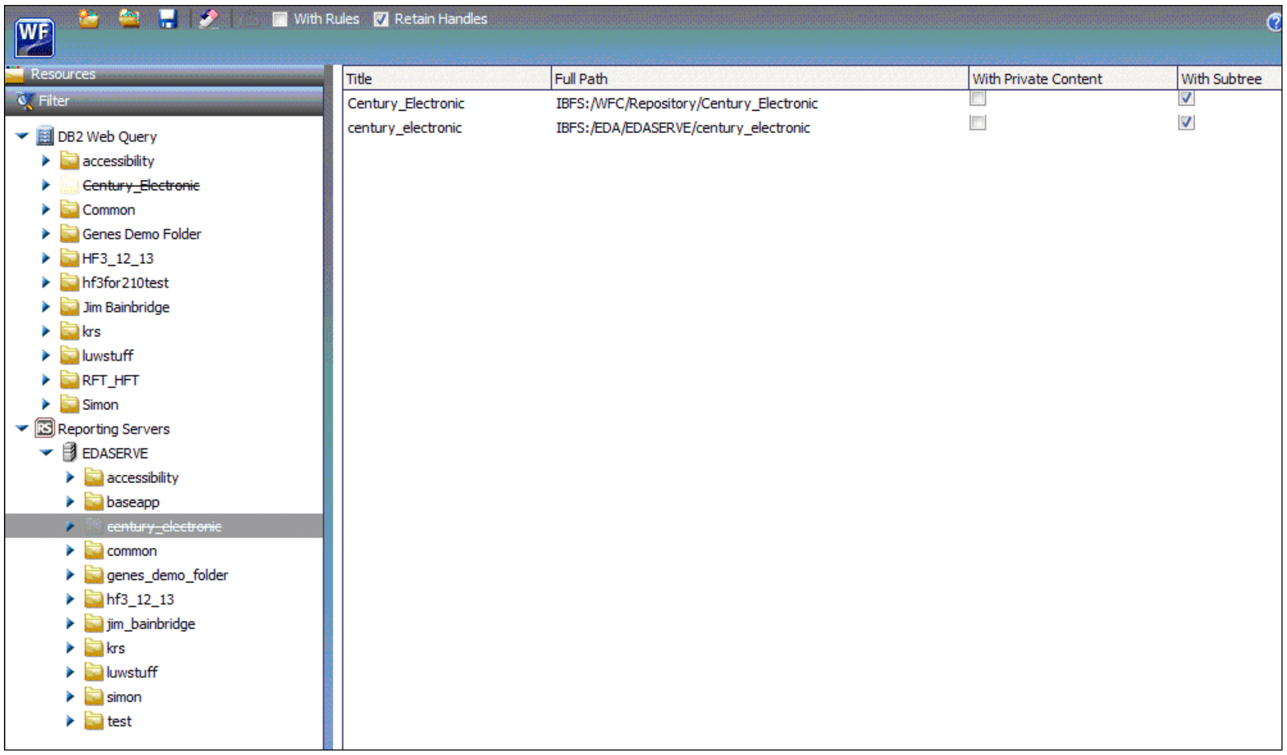
   Notice that the Century_Electronic top level folder now appears in the scenario listing in the right-hand panel. Also notice that in the Resource Tree, a strikethrough format has been applied to that top level folder name. This lets you know that the object is already part of the scenario (preventing you from adding it again). See Figure 26-13.



Title	Full Path	With Private Content
Century_Electronic	IBFS:/WFC/Repository/Century_Electronic	☐

Figure 26-13 Scenario listing

6. Select synonyms under **Century_Electronic** in **Reporting Servers** tree and add them to the right panel as shown in Figure 26-14.



Title	Full Path	With Private Content	With Subtree
Century_Electronic	IBFS:/WFC/Repository/Century_Electronic	☐	✓
century_electronic	IBFS:/EDA/EDASERVE/century_electronic	☐	✓

Figure 26-14 Add resources to the scenario

7. Save this scenario by left-clicking the **Save** button as shown in Figure 26-15.

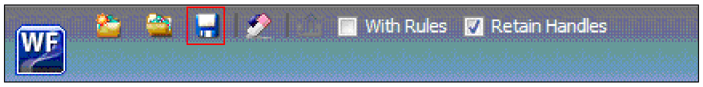


Figure 26-15 Save the scenario

8. The scenario has been saved, but as described previously, it is merely a template. To actually create a change management package based on this scenario, you must export it. Do this by clicking the **Export Scenario** button as shown in Figure 26-16.

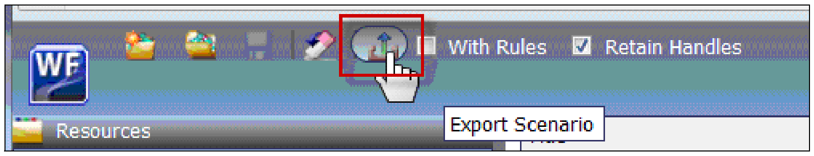


Figure 26-16 Export Scenario

9. After the export is finished, click **OK**.

Export Packages are stored in the IFS in the following directory (where a subfolder by the same name as the scenario is created):

```
/qibm/UserData/qwebqry/base80/cm/export/
```

Importing a change management package

In this section, we explain how to import a change management package:

1. On the source system, copy the entire subfolder **CenturyElectronic_cm** under:

   ```
   /qibm/UserData/qwebqry/base80/cm/export/
   ```

2. On the target system, paste the subfolder to the following subfolder

   ```
   /qibm/UserData/qwebqry/base80/cm/import/
   ```

3. Sign in to DB2 Web Query on the target system as an administrator.
4. From the resource tree of the BI portal, click **Change Management** to expand the Change Management tree. Under **Import** you should see the **CenturyElectronic_cm** package (placed there from step 2 above).

5. Right-click **CenturyElectronic_cm** and select **Import** as shown in Figure 26-17.

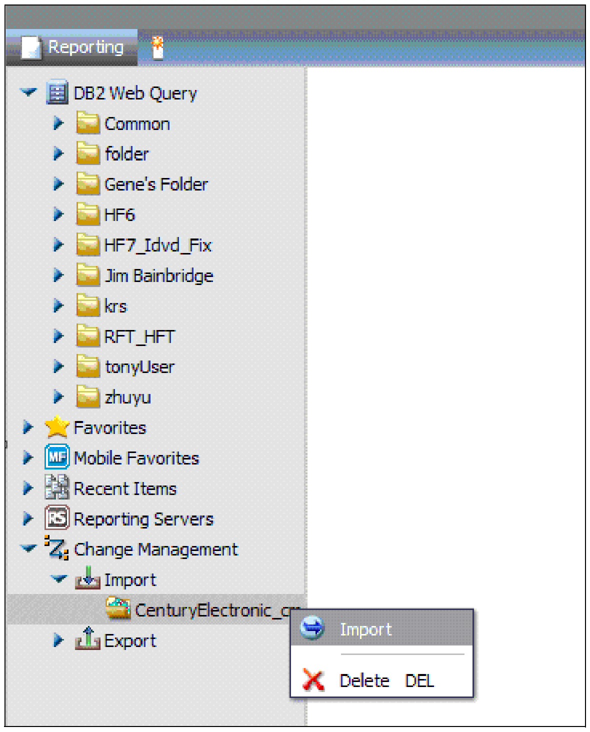
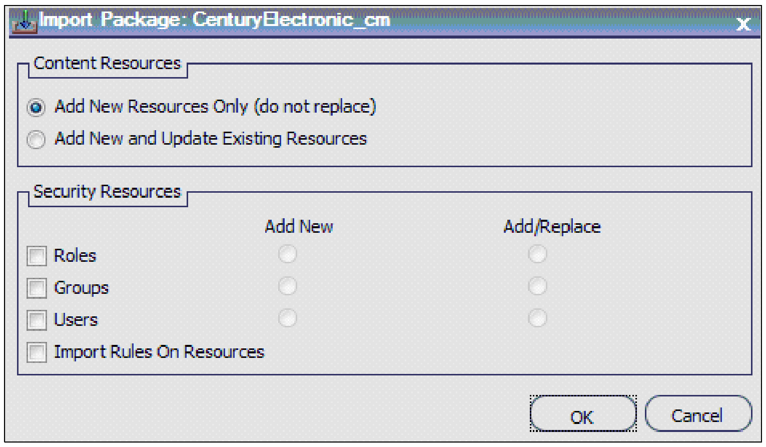


Figure 26-17 Menu to Import

The import package interface appears as shown in Figure 26-18.


Import Package: CenturyElectronic_cm

Content Resources

(●) Add New Resources Only (do not replace)
( ) Add New and Update Existing Resources

Security Resources

	Add New	Add/Replace
☐ Roles	○	○
☐ Groups	○	○
☐ Users	○	○
☐ Import Rules On Resources		

Figure 26-18 Import Package

The following options appear on the Import Package dialog window:

– Content Resource. Valid options are as follows:

  • **Add New Resources Only (do not replace)**: Does not update existing Resources on the target system

  • **Add New and Update Existing Resources**: Both add new resources and update existing resources.

– Security Resources. These options should NOT be selected and are reserved for future use.

6. Click **OK**. The message popped up indicates the status of importing. When finished, you should see the Import successful window as shown in Figure 26-19.

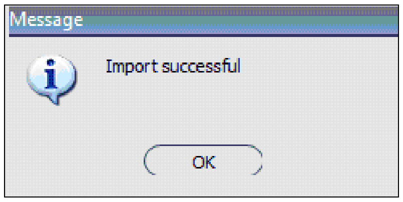


Figure 26-19 Import Successful message

7. Upon a successful import, the resources in CenturyElectronic_cm are imported as shown in Figure 26-20. You may need to refresh the contents of the resource tree to see the imported content.

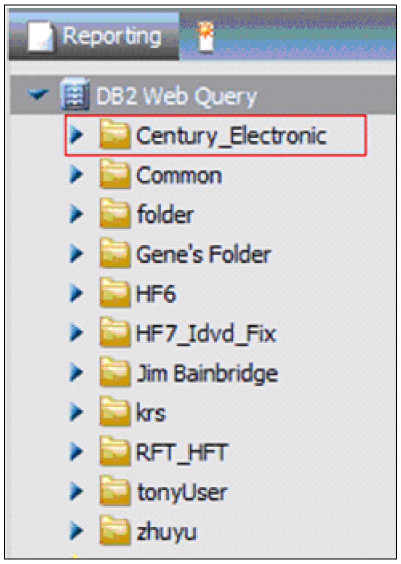


Figure 26-20 Imported Content

Note: If you want to see the details of the export or import requests, here is the log file of Change Management activity:

```
/qibm/Userdata/qwebqry/base80/logs/impex.log
```

D

Configuring Developer Workbench

Developer Workbench is the PC based tool that provides several DB2 Web Query development and administrative features that are not available from the product's browser interface. In this appendix, we review the tool's configuration steps and show how to set up several default options.

Configuring Developer Workbench

To configure Developer Workbench, proceed as follows:

1. Because Developer Workbench is a PC application, you must first install the licensed program 5733-WQX on the developer's PC. The Workbench product files, once delivered separately, are part of the product installation in V2.1 and can be found at the path depicted in Figure 26-21 on the integrated file system (IFS) after the licensed program install in complete:

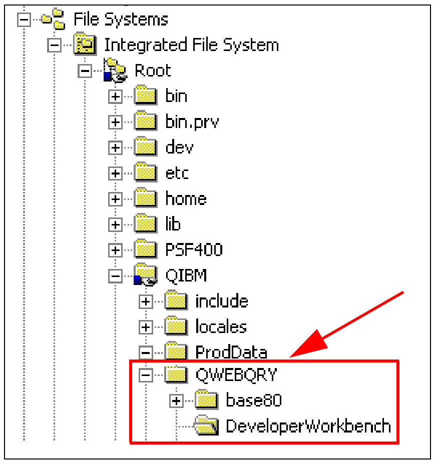


Figure 26-21 Location of Developer Workbench installation executable

2. After completing the Workbench install, when you first launch the program, you must configure the software to address your IBMi server. This is similar to the one-time setup required for other IBMi client server products like Access and Navigator. Figure 26-22 shows this initial screen. You will add a connection for your IBMi system under the WebFOCUS Environments following the next steps:

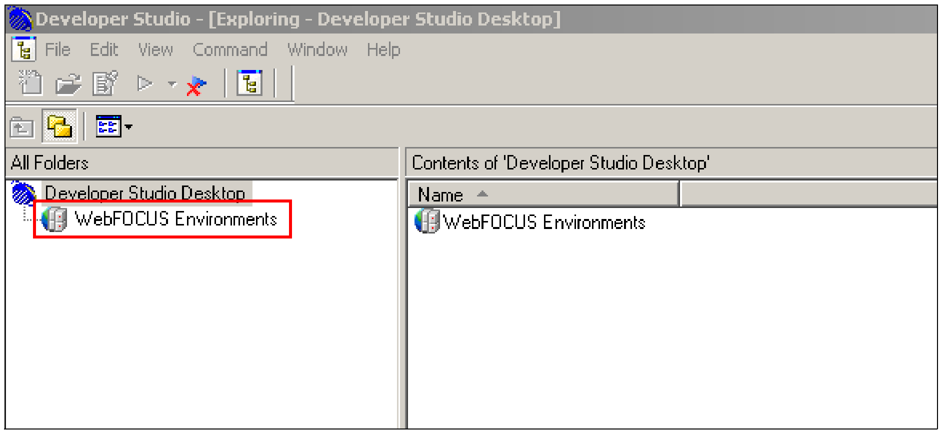

Developer Studio - [Exploring - Developer Studio Desktop]
File Edit View Command Window Help
All Folders
Developer Studio Desktop
WebFOCUS Environments
Contents of 'Developer Studio Desktop'
Name
WebFOCUS Environments

Figure 26-22 Initial Workbench pane

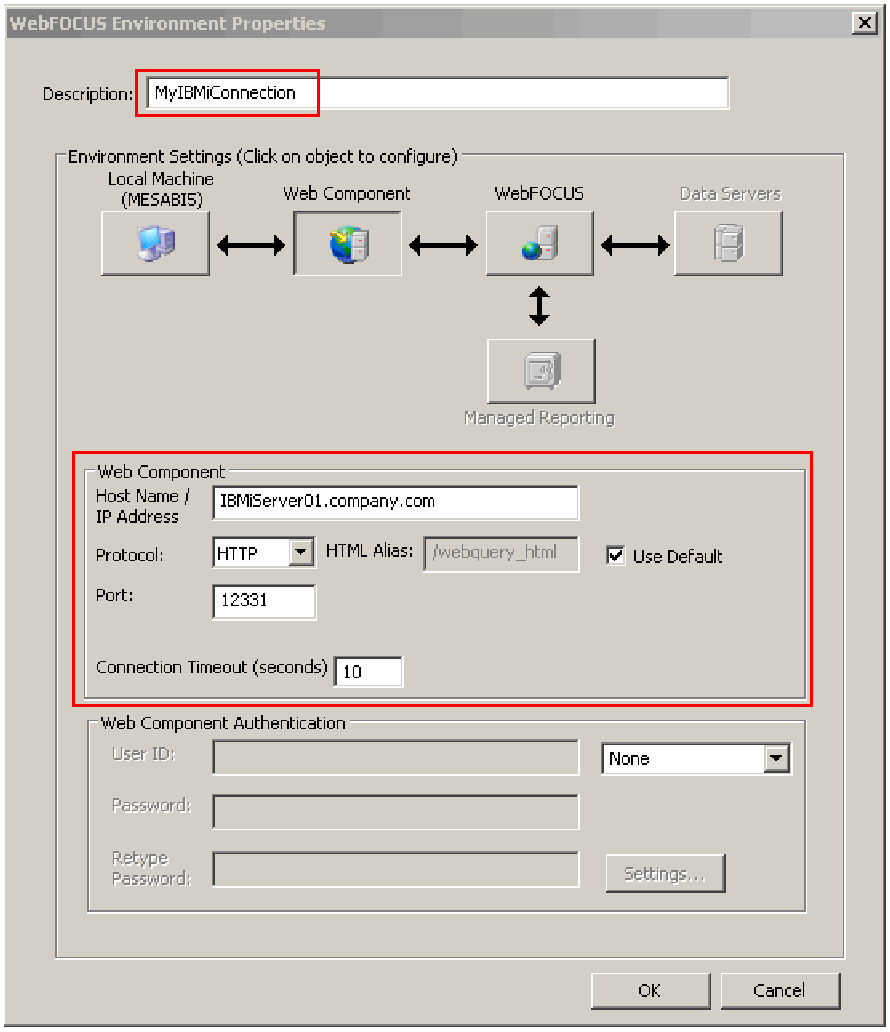
   a. Right-click **WebFOCUS** and select **Add**.

b. In the next window (Figure 26-23), complete these steps:

   i. Type a description of your System i environment.
   ii. For Host Name/IP Address, type a name.
   iii. For Port, verify that 12331 is selected.

   Tip: If you changed the DB2 Web Query default port number, you must clear the **Use Default** box and specify your installation's port number.

   iv. For HTML Alias, ensure that it is **/webquery_html**.
   v. Click **OK**.



Description: MyIBMiConnection

Host Name / IP Address: IBMiServer01.company.com

Protocol: HTTP   HTML Alias: /webquery_html   ☑ Use Default

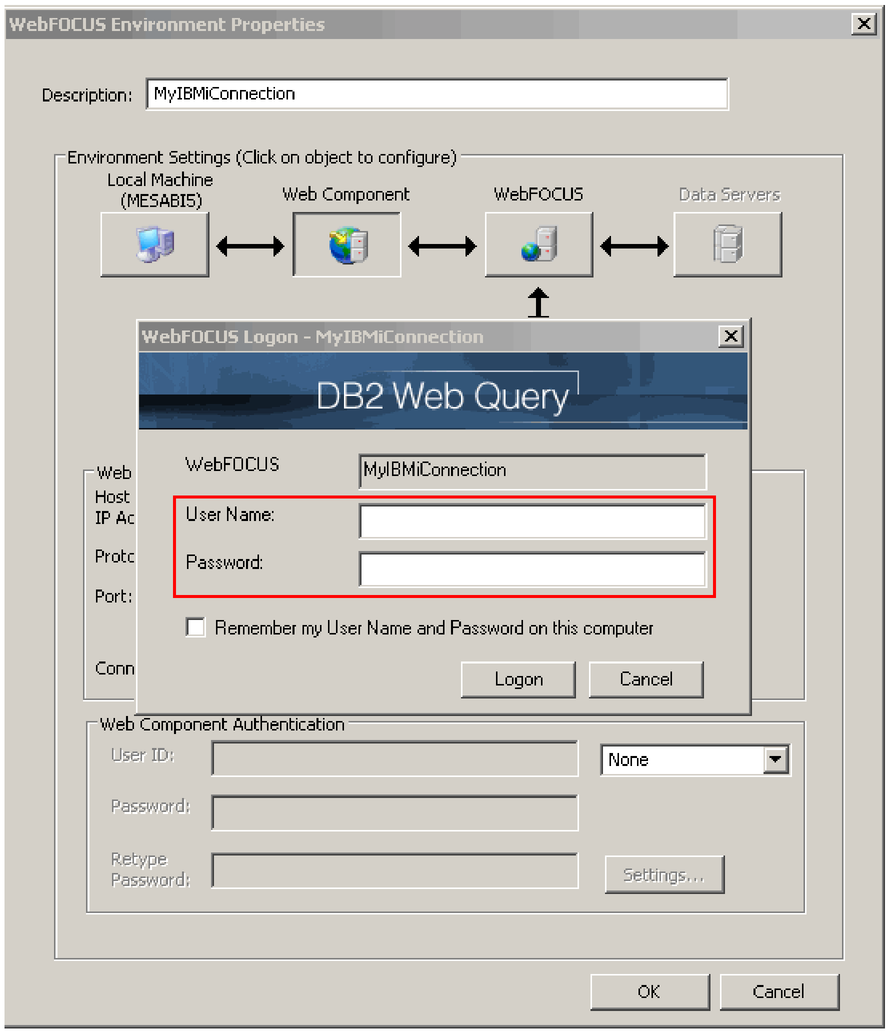
Port: 12331

Connection Timeout (seconds) 10

Figure 26-23 IP Address Configuration for the WorkBench to IBMi connection

3. The first time that you open the System i environment or connection that you just added, you will be prompted to enter your IBMi credentials, in a pop-up window. Sign on with your administrator or developer authorized profile. See Figure 26-24.

Tip: Select the **Remember my User Name and Password on this computer** option during this sign-on if you do not want to be prompted with these sign-on windows again for this IBMi connection.



Description: MyIBMiConnection

WebFOCUS Logon - MyIBMiConnection

WebFOCUS: MyIBMiConnection

User Name:

Password:

☐ Remember my User Name and Password on this computer

None

Figure 26-24 Workbench Logon window

Before going further with this chapter, you should verify you have created a folder under the Century Electronics folder and named it: “**Assignment 09 - Exploring and comparing other Dashboard options**”. Place your work from this chapter into this folder.

You will now be able to expand the menu trees under your connection and begin to review metadata or develop applications in the workbench. There are two primary areas beneath the connection: Data Servers and Repository as shown in Figure 26-25.

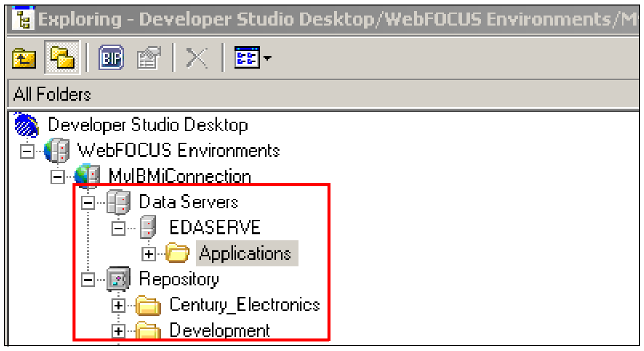
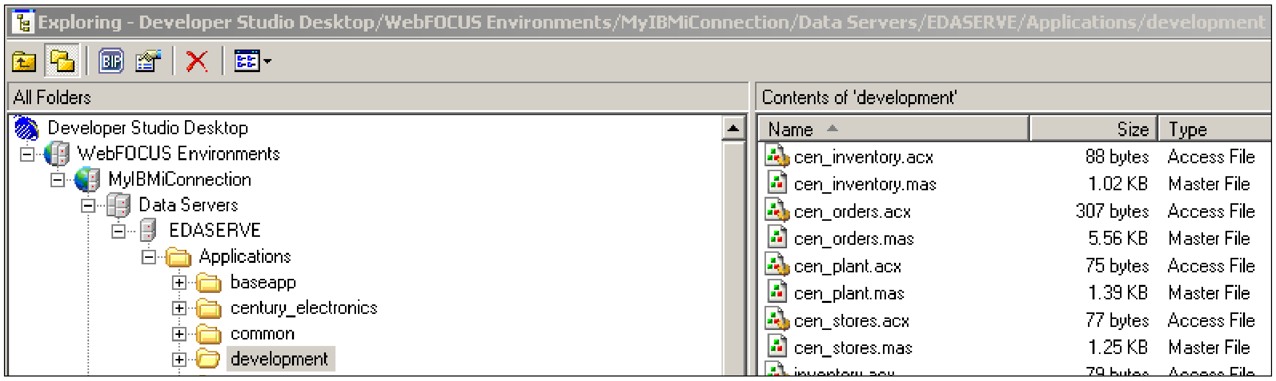


Figure 26-25 Expanding the tree - Web Query work area for your IBMi

In the right pane of Figure 26-26, you see the metadata files, you master (.mas) and access (.acx) files. Remember that the *master file* contains the field names and formats for your table. The *access file* contains information about the actual name and location of the physical table and the primary keys for the table. Nearly all your work is done with the master files.


Exploring - Developer Studio Desktop/WebFOCUS Environments/MyIBMiConnection/Data Servers/EDASERVE/Applications/development
All Folders
Developer Studio Desktop
WebFOCUS Environments
MyIBMiConnection
Data Servers
EDASERVE
Applications
baseapp
century_electronics
common
development
Contents of 'development'

Name	Size	Type
cen_inventory.acx	88 bytes	Access File
cen_inventory.mas	1.02 KB	Master File
cen_orders.acx	307 bytes	Access File
cen_orders.mas	5.56 KB	Master File
cen_plant.acx	75 bytes	Access File
cen_plant.mas	1.39 KB	Master File
cen_stores.acx	77 bytes	Access File
cen_stores.mas	1.25 KB	Master File

Figure 26-26 Master data and Access files

When you edit metadata with the synonym editor, you get a number of view choices for reviewing your data relationships. The following figures show two of these views for the CEN_ORDERS cluster. There are more detailed field-by-field views available as well. See Figure 26-27 and Figure 26-28.

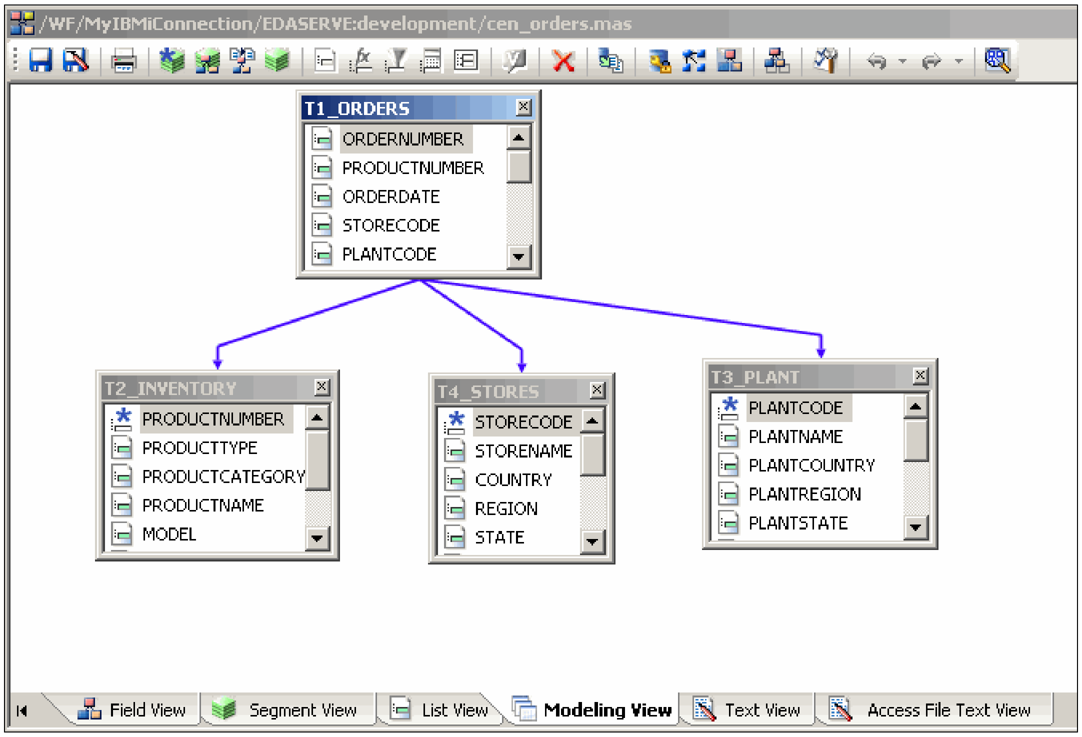
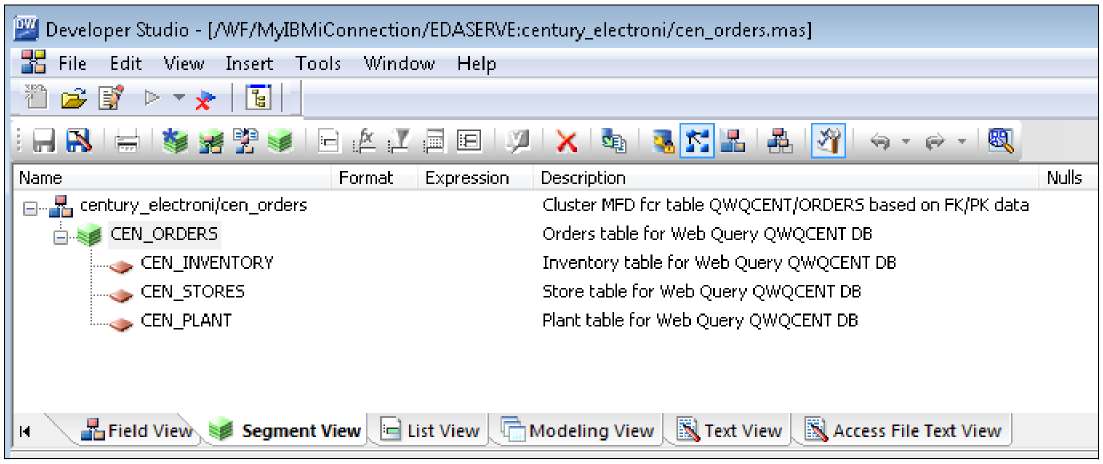


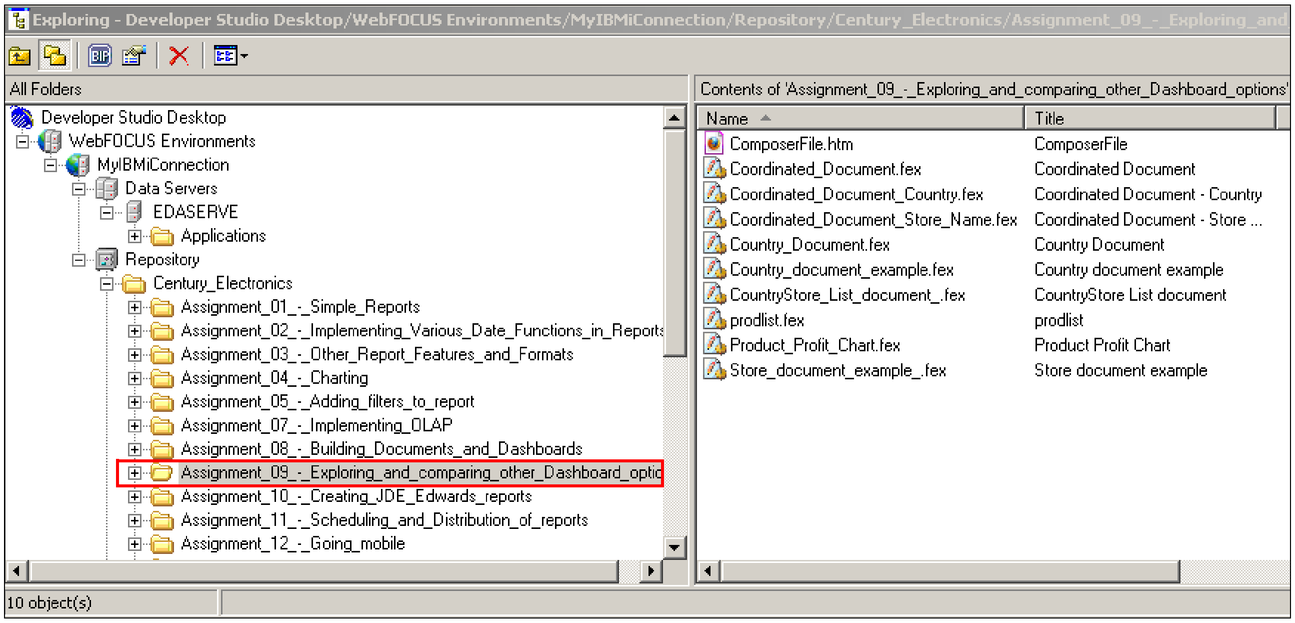
Figure 26-27 Table relationships - modeling view depicted in Workbench



Name	Format	Expression	Description	Nulls
century_electroni/cen_orders			Cluster MFD for table QWQCENT/ORDERS based on FK/PK data	
CEN_ORDERS			Orders table for Web Query QWQCENT DB	
CEN_INVENTORY			Inventory table for Web Query QWQCENT DB	
CEN_STORES			Store table for Web Query QWQCENT DB	
CEN_PLANT			Plant table for Web Query QWQCENT DB	

Figure 26-28 Table relationships - segment view depicted in Workbench

The source code or procedures that store your report definitions are located in DB2 tables in the repository library. This is a change from prior releases when these files were stored in the IFS. You may see performance and security improvements from this change. You see them as *.FEX files under the Repository tree in folders that you create. A suggested strategy is a company folder for production and a development folder for content creation. You will expand the Repository branch as shown in Figure 26-29 when you want to create a new report or modify an existing one in the workbench. The workbench will launch the Info Assist component.


Exploring - Developer Studio Desktop/WebFOCUS Environments/MyIBMiConnection/Repository/Century_Electronics/Assignment_09_-_Exploring_and

All Folders
- Developer Studio Desktop
  - WebFOCUS Environments
    - MyIBMiConnection
      - Data Servers
        - EDASERVE
          - Applications
      - Repository
        - Century_Electronics
          - Assignment_01_-_Simple_Reports
          - Assignment_02_-_Implementing_Various_Date_Functions_in_Reports
          - Assignment_03_-_Other_Report_Features_and_Formats
          - Assignment_04_-_Charting
          - Assignment_05_-_Adding_filters_to_report
          - Assignment_07_-_Implementing_OLAP
          - Assignment_08_-_Building_Documents_and_Dashboards
          - Assignment_09_-_Exploring_and_comparing_other_Dashboard_optic
          - Assignment_10_-_Creating_JDE_Edwards_reports
          - Assignment_11_-_Scheduling_and_Distribution_of_reports
          - Assignment_12_-_Going_mobile

Contents of 'Assignment_09_-_Exploring_and_comparing_other_Dashboard_options'

Name	Title
ComposerFile.htm	ComposerFile
Coordinated_Document.fex	Coordinated Document
Coordinated_Document_Country.fex	Coordinated Document - Country
Coordinated_Document_Store_Name.fex	Coordinated Document - Store ...
Country_Document.fex	Country Document
Country_document_example.fex	Country document example
CountryStore_List_document_.fex	CountryStore List document
prodlist.fex	prodlist
Product_Profit_Chart.fex	Product Profit Chart
Store_document_example_.fex	Store document example

10 object(s)

Figure 26-29 Tutorials under the Century top level folder

Setting Developer Workbench default options

Developer Workbench has a set of options that can be tailored to an individual user. To access these options, select **Window** → **Options** (Figure 26-30). We use the tailoring options in later tutorials.

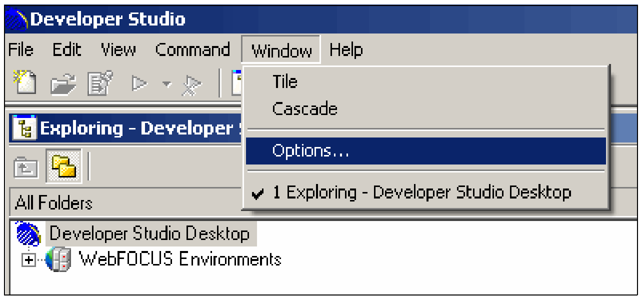
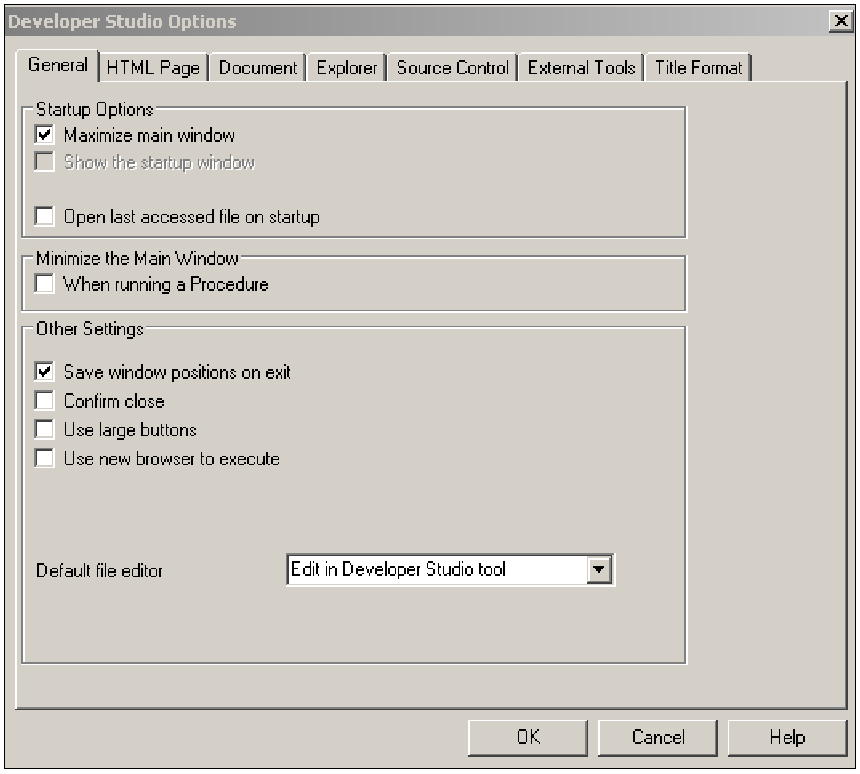


Figure 26-30 Displaying Developer Workbench options

Figure 26-31 shows the General options tab.



Startup Options
- [✓] Maximize main window
- [ ] Show the startup window
- [ ] Open last accessed file on startup

Minimize the Main Window
- [ ] When running a Procedure

Other Settings
- [✓] Save window positions on exit
- [ ] Confirm close
- [ ] Use large buttons
- [ ] Use new browser to execute

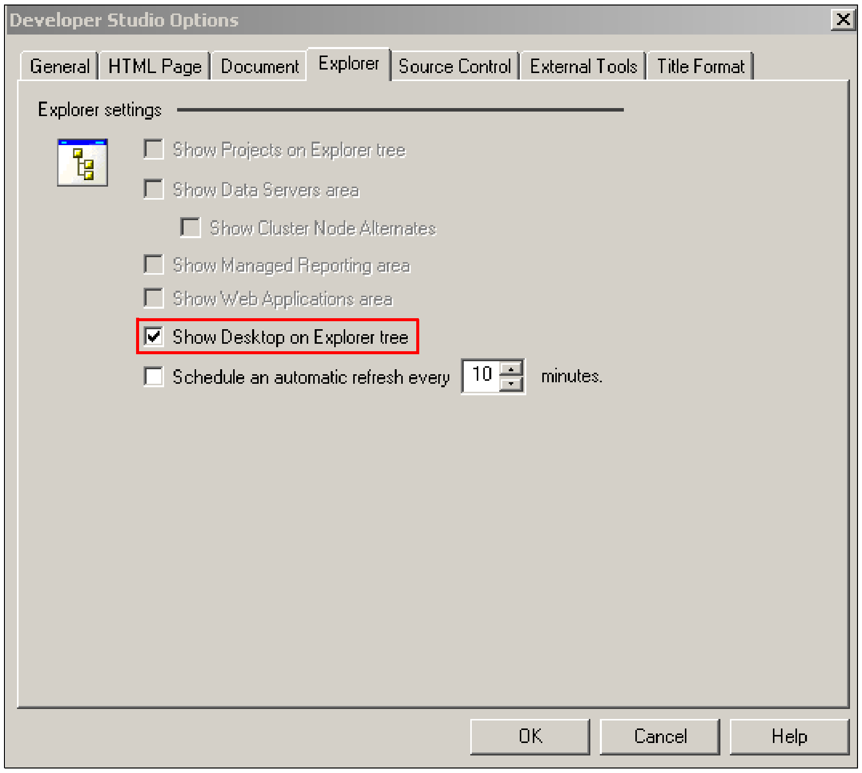
Default file editor: Edit in Developer Studio tool

Figure 26-31 General options tab

Developer Workbench allows you to show your Windows desktop explorer tree in the same window as your DB2 Web Query explorer tree. On the Explorer tab, select the **Show Desktop on Explorer tree** option (Figure 26-32).

Showing your Windows desktop tree within Developer Workbench has definite benefits. The obvious benefit is the ability to stay within one working environment as a programmer. The less that you have to flip back and forth between your development mode and Windows, the better.

An additional benefit is the ease with which you can copy items from your PC hard drive to the DB2 Web Query folder structure. You might want to move style sheets, company logos, pictures, and other PC files between the different file structures.



Explorer settings

- [ ] Show Projects on Explorer tree
- [ ] Show Data Servers area
  - [ ] Show Cluster Node Alternates
- [ ] Show Managed Reporting area
- [ ] Show Web Applications area
- [✓] Show Desktop on Explorer tree
- [ ] Schedule an automatic refresh every 10 minutes.

Figure 26-32 Explorer options

For more details about using Developer Workbench, refer to the online help text and the OLAP chapter in this document.

IBM DB2 Web Query for i Version 2.1: Implementation Guide

(1.5” spine)
1.5”<-> 1.998”
789 <->1051 pages

IBM DB2 Web Query for i Version 2.1 Implementation Guide

Follow the best practice guides to simplify report development

Take your reporting to the next level of Business Intelligence

Learn DB2 Web Query by using the easy-to-follow tutorials

Business Intelligence (BI) is a broad term relating to applications designed to analyze data for purposes of understanding and acting on the key metrics that drive profitability in an enterprise. Key to analyzing that data is providing fast, easy access to it while delivering it in formats or tools that best fit the needs of the end user.

At the core of any business intelligence solution are end user query and reporting tools that provide intuitive access to data supporting a spectrum of end users from executives to “power users,” from spreadsheet aficionados to the external Internet consumer.

IBM DB2 Web Query for i offers a set of modernized tools for a more robust, extensible, and productive reporting solution than the popular Query for System i tool (also known as Query/400). IBM DB2 Web Query for i preserves investments in the reports developed with Query/400 by offering a choice of importing definitions into the new technology or continuing to run existing Query/400 reports as-is. But it also offers significant productivity and performance enhancements by leveraging the latest in DB2 for i query optimization technology.

This IBM Redbooks publication provides a broad understanding of the new DB2 Web Query product. It entails a group of self-explanatory tutorials to help you get up to speed quickly. Overall, this book is designed for IT users. You can use Part 2, “Tutorials for DB2 Web Query” on page 161, as stand-alone tutorials for anyone who is developing their own queries.

SG24-8063-00 ISBN 0738437964