IBM Cognos Analytics – Working with Multiple Relational Data Sources

Nature of Document: **Guideline**

Product(s): **Framework Manager, Reporting**

Area of Interest: **Modeling, Reporting**

IBM Cognos Analytics

Table of Contents

1 Introduction

1.1 Purpose

This document will provide guidelines around working with multiple relational data sources in Framework Manager and recommendations on how to improve performance in various scenarios.

1.2 Applicability

This document was written and tested against IBM Cognos 8.x and IBM Cognos 10.x, but the concepts apply to Cognos Analytics 11.x as well.

1.3 Exclusions and Exceptions

This document does not cover working with multiple OLAP sources.

1.4 Assumptions

This document assumes readers have experience with IBM Cognos Analytics, in particular Reporting and Framework Manager.

2 Overview

Many organizations are faced with disparate data sources and the task of trying to consolidate this information for reporting and analysis. Although you may want to consolidate all your data into a single relational database for reporting purposes, it may not be feasible, financially viable, or may take quite some time to implement and require an interim solution.

IBM Cognos Analytics (CA) provides facilities to bring these disparate data sources together for reporting purposes, but there are several considerations that should be taken into account before embarking on a CA project.

This document will look at the different ways of dealing with disparate data sources and identify the pros and cons to each method. It will also provide tips on how to reduce performance impact where applicable.

3 Data Access Strategies

Before beginning data access to your company's data for reporting purposes, you may want to ask some of the following questions with regard to the data and how it will be consumed.

- How fresh does the data need to be? Are you reporting on yearly, quarterly, monthly, weekly, daily, hourly, or even up to the second data? Knowing this information can help you decide how to access or store your reporting data. Is a data mart or warehouse the way to go, can a cube be used, or must you go against live data?
- Will users of the data mainly look at summarized data or require an interactive experience (drill up and down through the data)? If so, then perhaps an OLAP package or dimensionally modeled relational (DMR) package should be considered. See DMR note below for more information.
- Is the data available able to support your business questions? For example, lack of conformed product data throughout the company can prevent querying product metrics across the divisions in the company. Product data in one division should mean and be represented the same way in all divisions. This way metrics from all divisions can be compared through the conformed product data.

These questions are just some examples of the challenges you may be faced with when trying to implement a CA project. Although OLAP technology has been mentioned, this document will focus on accessing multiple relational data sources only.

Note: Dimensionally modeled relational (DMR) modeling is a capability that IBM Cognos Framework Manager provides allowing you to specify dimensional information for relational metadata. This allows for OLAP-style queries. Where possible, using OLAP technologies will typically provide better performance, but DMR may sometimes be appropriate where smaller data sets are concerned (one less technology to implement in the project) or if OLAP-style queries are required on live data.

4 Do you Really Need to Create Joins Between Data-bases?

One of the first questions you will want to ask when dealing with disparate data sources is, “do I actually need to join these data sources together?” Do you simply require displaying the data in the same dashboard or report, but without a physical join? Each database can have it's own query in a dashboard or dashboard-style report and be displayed along side each other without a physical joining of the data. In other words, co-locate the data. For example, executives may want to see the weekly correlation between weather statistics and outdoor product sales. Each is contained in a different database. A join between these two databases may not be necessary if they just want to see what the weather was on a given day of sales and quickly see a visual trend of increased sales on warmer days. Each of these queries can be represented in a list or graph and displayed side-by-side in a dashboard or dashboard-style report.

In these types of scenarios, performance concerns due to local join processing of data on the CA servers are avoided.

5 Are Joins or Master/Detail Relationships Better?

There will be instances where you simply must relate the data from disparate data sources for certain reports. For example, each region of a company may manage it's own account information in one database while transactional sales figures are recorded in a central system. Each region may want to use their account data to filter the sales records.

CA supports relating disparate databases in either Framework Manager or Reporting. In the example we are using, a relationship can be created between the regional Account table from one database to the Sales table in the other database. The effect of this, however, will produce local data processing on the CA servers. Each database will be queried for a potentially significant amount of data and then joined locally to filter out unwanted records. To see if performance is acceptable, you may want to apply some basic math to anticipated queries and, of course, test them in your environment. What is the amount of data in table X that you would request and then join to a certain amount of data in table Y? For smaller data volumes or cases where reports are run infrequently (perhaps scheduled once a week during off peak hours), creating cross-database joins may be a perfectly acceptable and simple solution.

For larger data sources however, this may present longer wait times for report output and may not be an optimal solution. Consider the example where the Sales table has several million rows, but each region is only interested in a few hundred thousand, the extra processing may not yield acceptable wait times. In these cases, Reporting authors can use master/detail relationships as an alternative. Master/detail relationships will retrieve the master data first, in this case from Accounts, and then for each account retrieve the related Sales records. This technique will generated more queries to the detail table database, but each query will be filtered.

By implementing master/detail relationships, performance can be increased by reducing the amount of records returned by the database on the detail side, however, as with any technique, there are thresholds. If the master query table has a large volume of data, perhaps millions of rows, then there might be an unacceptable amount of queries issued to the database table on the detail side.

6 Working with Single Instances of a Database Vendor

6.1 Understanding Differences in Relational Database Vendor Technologies

When dealing with relational databases and CA, you may encounter situations where you are joining items from the same database vendor instance, but through different CA data source connections defined in the CA portal. In these scenarios, the CA servers will locally join the data from the various defined CA data source connections. This behavior may or may not be desired. If it is not desired, you can configure Framework Manager to push the processing to the database vendor.

The following will explain how different database vendor technologies behave, how metadata appears during import in Framework Manager, and finally how to configure your model so that processing is pushed to the database if desired.

There are a few different scenarios for how database vendors deal with the qualification of their objects. For example, some vendors only have the concept of instance and a collection of objects such as tables, indexes, and so on. This document will focus on two of the most typical scenarios which will be labeled Scenario A and Scenario B.

6.1.3 Scenario A – Instance and Schemas

In Scenario A, the vendor has the concept of instance (the database software running on a computer) and schemas (also referred to as "user" by some vendors). Illustration 1 below illustrates a single instance named A containing four schemas named A, B, C, and D. A vendor example of this scenario is Oracle.

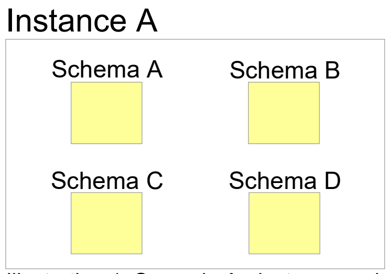


Illustration 1: Scenario A - instance and schemas

In CA, in this scenario, you can create one CA data source for the database vendor instance and have access to all schemas within it. Illustration 2 below shows the Select Objects screen of the Framework Manager Metadata Wizard import process where the instance name (Oracle) is at the top of the metadata tree and the various schemas such as CM, GOSALES, and so on, are direct children of the instance.

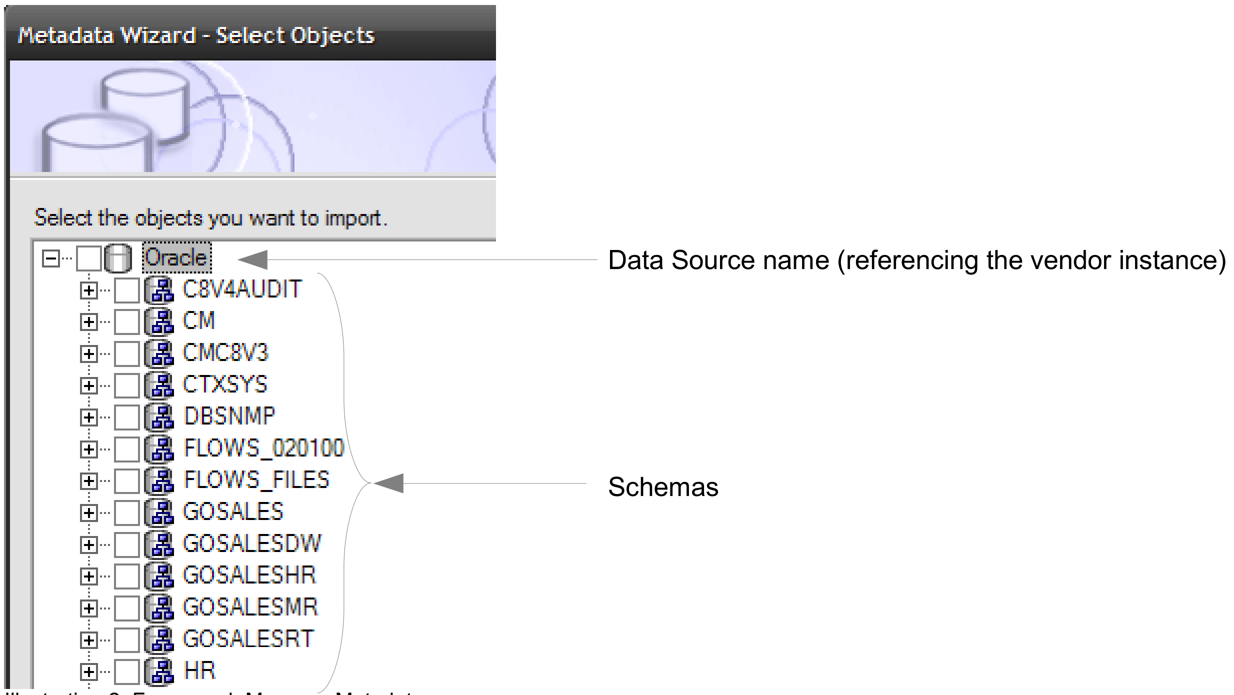

Metadata Wizard - Select Objects
Select the objects you want to import.
Oracle
C8V4AUDIT
CM
CMC8V3
CTXSYS
DBSNMP
FLOWS_020100
FLOWS_FILES
GOSALES
GOSALESDW
GOSALESHR
GOSALESMR
GOSALESRT
HR
Data Source name (referencing the vendor instance)
Schemas

Illustration 2: Framework Manager Metadata Wizard showing Oracle example of instance and schemas

You are free to import items from multiple schemas and join tables from the different schemas as required. In this scenario, CA will push cross-schema join processing to the database vendor providing it supports the query.

6.1.4 Scenario B – Instance, Catalogs, and Schemas

In Scenario B, the vendor has the concept of instance (again, the database software running on a computer), catalogs (also referred to as "database" by some vendors), and schemas (again, also referred to as "user" by some vendors). Illustration 3 below shows a single instance named A with two catalogs named A and B. Catalog A contains two schemas named A and B, and Catalog B contains two schemas named C and D. A vendor example of this scenario is Microsoft SQL Server.

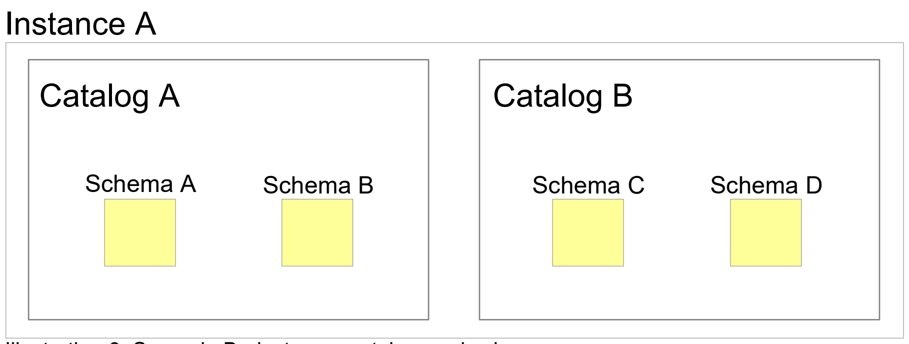


Illustration 3: Scenario B - instance, catalog, and schemas

In this scenario, in CA, you must create one CA data source per catalog you wish to import. Illustration 4 below shows the Select Objects screen of the Framework Manager Metadata Wizard import process where the instance name (GOSALES) is at the top of the metadata tree, followed by the catalog name (GOSALES), and below that the various schemas such as gosales, gosaleshr, and so on.

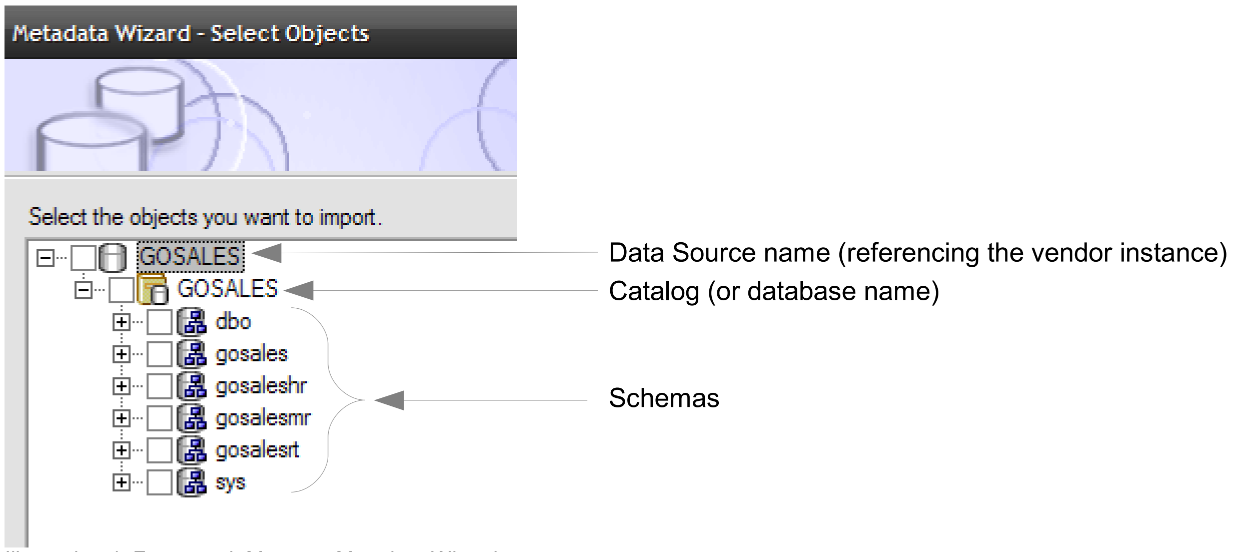

Metadata Wizard - Select Objects
Select the objects you want to import.
GOSALES
GOSALES
dbo
gosales
gosaleshr
gosalesmr
gosalesrt
sys
Data Source name (referencing the vendor instance)
Catalog (or database name)
Schemas

Illustration 4: Framework Manager Metadata Wizard showing Microsoft SQL Server example of instance, catalog, and schemas

In Scenario B, if you wish to import additional metadata from other catalogs, you will need to do so through another CA data source connection for each catalog. Once you have the items imported from the various catalogs, you can join items from one catalog to another. For example, Illustration 5 below shows a table from the C8-Bursting catalog (database) joined to a table in the GOSALESDW catalog (database) from the same SQL Server instance.

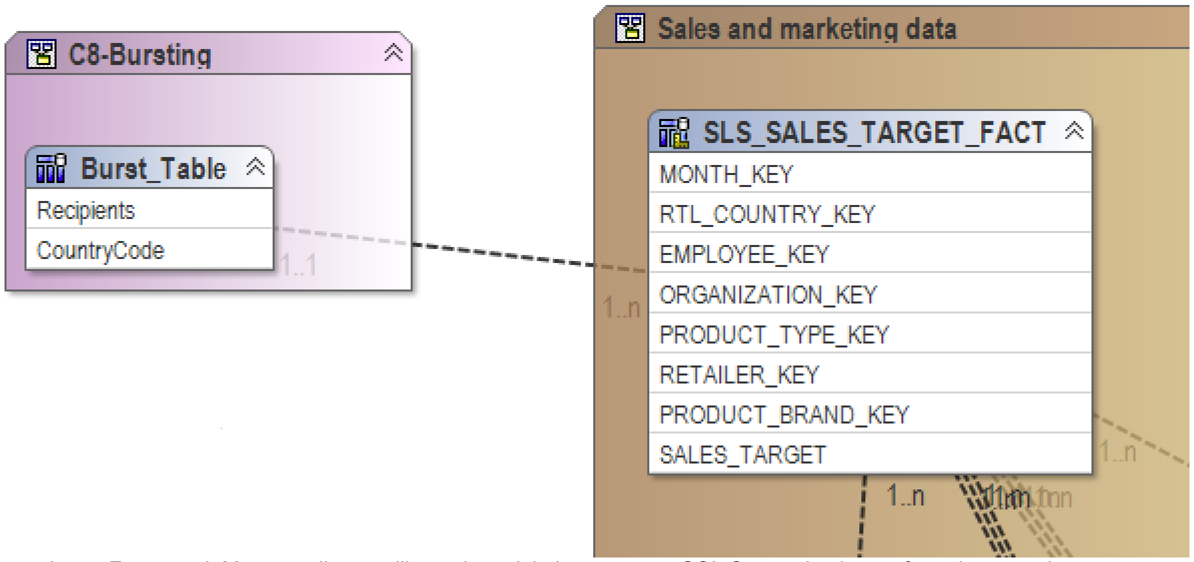
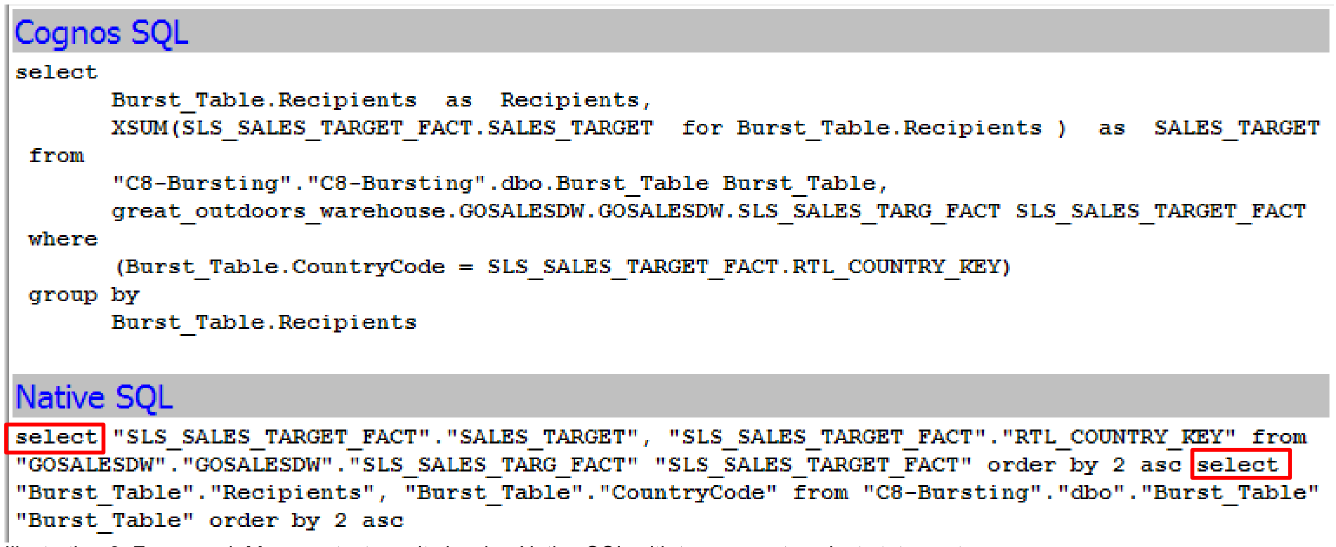


Illustration 5: Framework Manager diagram illustrating a join between two SQL Server databases from the same instance

However, when you query items from the joined query subjects from each catalog, CA will treat each catalog as a different data source instance since there is not enough information to indicate that it is the same instance. In fact, even if you created two data sources in CA to the same instance and catalog and joined between them, CA would still treat them as separate instances. In any case, this will prevent CA from pushing the join between the catalogs to the database vendor. This can be seen by examining the native SQL of such a query.

For example, a query with Recipients from Burst_Table and SALES_TARGET from SLS_SALES_TARGET_FACT produces the SQL shown in Illustration 6. The Cognos SQL in this Illustration shows a join between the tables, but the native SQL does not. It shows two separate select statements (highlighted below). This indicates that two separate result sets will be retrieved and joined locally on the CA servers.



Cognos SQL

```
select
       Burst_Table.Recipients  as  Recipients,
       XSUM(SLS_SALES_TARGET_FACT.SALES_TARGET  for Burst_Table.Recipients )  as  SALES_TARGET
 from
       "C8-Bursting"."C8-Bursting".dbo.Burst_Table Burst_Table,
       great_outdoors_warehouse.GOSALESDW.GOSALESDW.SLS_SALES_TARG_FACT SLS_SALES_TARGET_FACT
 where
       (Burst_Table.CountryCode = SLS_SALES_TARGET_FACT.RTL_COUNTRY_KEY)
 group by
       Burst_Table.Recipients
```

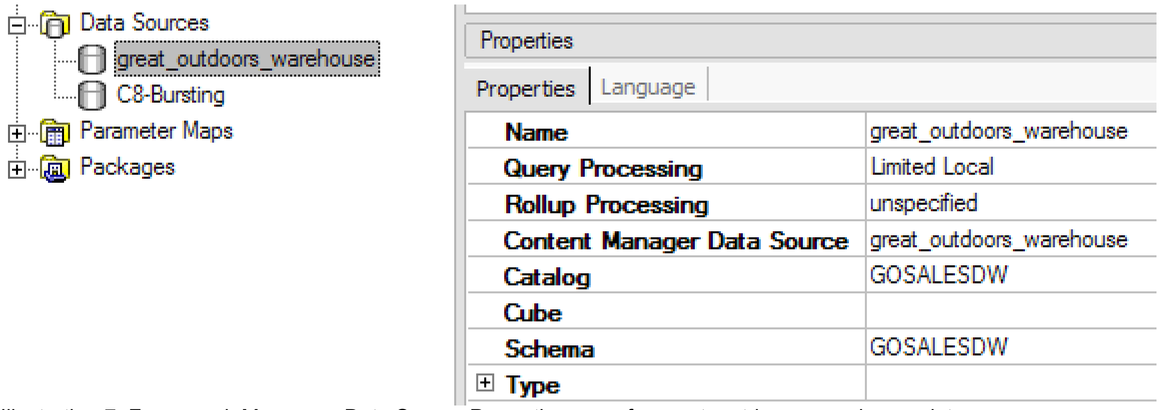
Native SQL

```
select "SLS_SALES_TARGET_FACT"."SALES_TARGET", "SLS_SALES_TARGET_FACT"."RTL_COUNTRY_KEY" from
"GOSALESDW"."GOSALESDW"."SLS_SALES_TARG_FACT" "SLS_SALES_TARGET_FACT" order by 2 asc select
"Burst_Table"."Recipients", "Burst_Table"."CountryCode" from "C8-Bursting"."dbo"."Burst_Table"
"Burst_Table" order by 2 asc
```

Illustration 6: Framework Manager test result showing Native SQL with two separate select statements

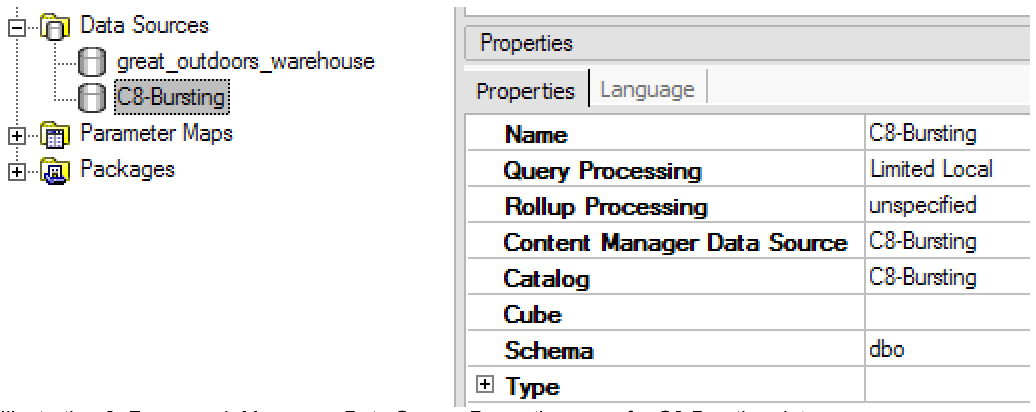
If the desired approach is to push the join to the database instance, you simply need to change the Data Source properties in the Framework Manager model to all point to one CA data source connection.

In this example there are two data sources involved, each with different property settings as seen in Illustrations 7 and 8. The Content Manager Data Source property for the great_outdoors_warehouse data source points to great_outdoors_warehouse and the same property for the C8-Bursting data sources points to C8-Bursting.



Name	great_outdoors_warehouse
Query Processing	Limited Local
Rollup Processing	unspecified
Content Manager Data Source	great_outdoors_warehouse
Catalog	GOSALESDW
Cube	
Schema	GOSALESDW
Type	

Illustration 7: Framework Manager - Data Source Properties pane for great_outdoors_warehouse data source



- Data Sources
  - great_outdoors_warehouse
  - C8-Bursting
- Parameter Maps
- Packages

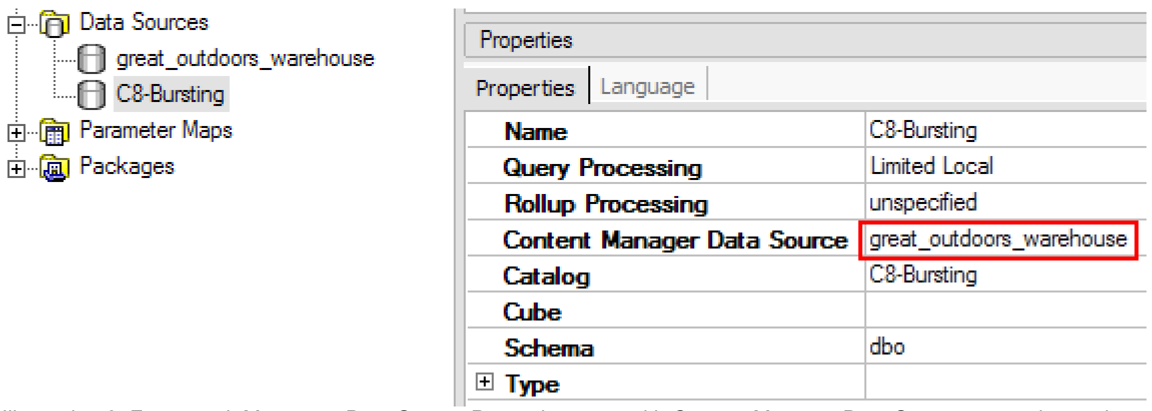
Properties

Properties | Language

Name	C8-Bursting
Query Processing	Limited Local
Rollup Processing	unspecified
Content Manager Data Source	C8-Bursting
Catalog	C8-Bursting
Cube	
Schema	dbo
Type	

Illustration 8: Framework Manager - Data Source Properties pane for C8-Bursting data source

To push the join between the two catalogs down to the database vendor, simply have both Content Manager Data Source properties point to the same CA data source name. This tells CA to use the same data source instance but different catalogs within that instance. In this example, the C8-Bursting data source will have its Content Manager Data Source property changed to point to great_outdoors_warehouse as shown in Illustration 9.



- Data Sources
  - great_outdoors_warehouse
  - C8-Bursting
- Parameter Maps
- Packages

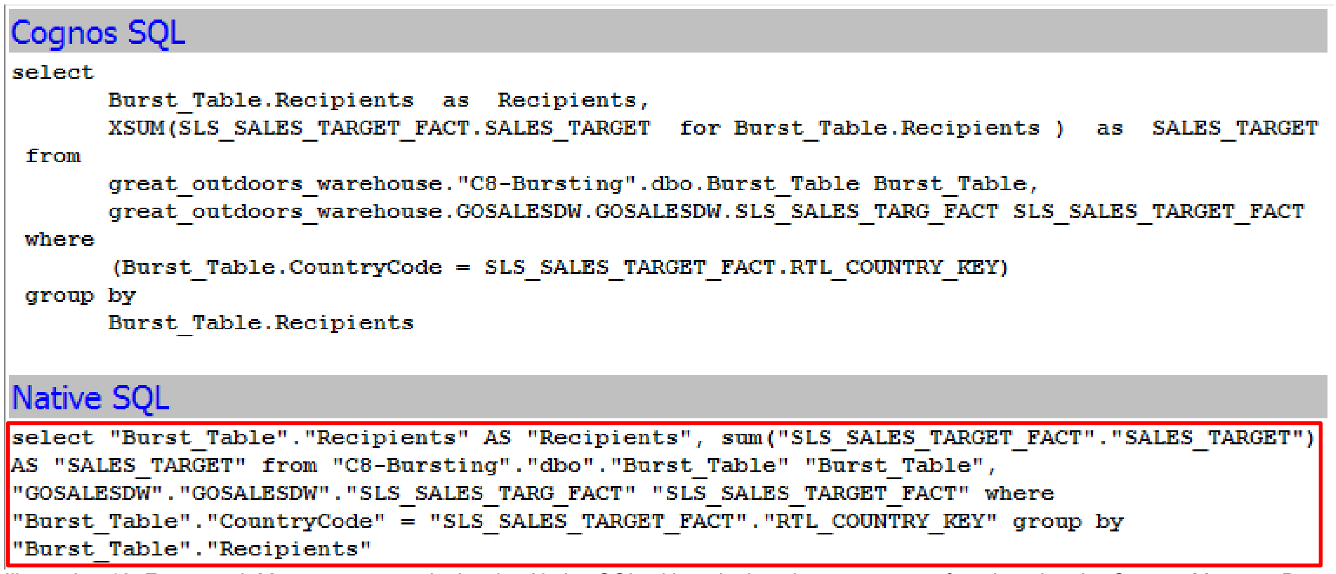
Properties

Properties | Language

Name	C8-Bursting
Query Processing	Limited Local
Rollup Processing	unspecified
Content Manager Data Source	great_outdoors_warehouse
Catalog	C8-Bursting
Cube	
Schema	dbo
Type	

Illustration 9: Framework Manager - Data Source Properties pane with Content Manager Data Source name changed to match the great_outdoors_warehouse data source name

Now, when the same query is run, the native SQL that is generated is a single select statement joining the two tables from the different catalogs at the database tier as shown in Illustration 10.



Cognos SQL

```
select
       Burst_Table.Recipients  as  Recipients,
       XSUM(SLS_SALES_TARGET_FACT.SALES_TARGET  for Burst_Table.Recipients )  as  SALES_TARGET
 from
       great_outdoors_warehouse."C8-Bursting".dbo.Burst_Table Burst_Table,
       great_outdoors_warehouse.GOSALESDW.GOSALESDW.SLS_SALES_TARG_FACT SLS_SALES_TARGET_FACT
 where
       (Burst_Table.CountryCode = SLS_SALES_TARGET_FACT.RTL_COUNTRY_KEY)
 group by
       Burst_Table.Recipients
```

Native SQL

```
select "Burst_Table"."Recipients" AS "Recipients", sum("SLS_SALES_TARGET_FACT"."SALES_TARGET")
AS "SALES_TARGET" from "C8-Bursting"."dbo"."Burst_Table" "Burst_Table",
"GOSALESDW"."GOSALESDW"."SLS_SALES_TARG_FACT" "SLS_SALES_TARGET_FACT" where
"Burst_Table"."CountryCode" = "SLS_SALES_TARGET_FACT"."RTL_COUNTRY_KEY" group by
"Burst_Table"."Recipients"
```

Illustration 10: Framework Manager test result showing Native SQL with a single select statement after changing the Content Manager Data Source property

A single select statement is generated and the join is pushed to the database.

7 Working with Multiple Database Instances or Vendors

As already stated earlier in this document, you can join disparate databases in Framework Manager, but by doing so, local processing of the joins between the databases will occur on the CA servers. If this is not desirable, you may want to consider federation technologies or use the DQM based join filter feature found in the CA 11.x versions of the product. This feature allows the modeller or report author to push the values of a join condition on the 1 side of a relationship into the WHERE clause of the query on the n side of the relationship. This can drastically filter the many side which may contain millions or more rows thereby allowing much reduced data sets for local processing.

The Join Filter feature is available in Framework Manager and Reporting in the properties of the join. The property is called Filter Type. In Data Modules, the feature is found in the Edit relationship dialog in the lower left corner. When you click on the gear icon, you will see a pop up and within that, an Optimization section. Please see corresponding documentation for the feature per component.

Beyond that, most major database vendors offer some form of federation technology which allows you to incorporate tables from other databases in your vendor's environment. This includes creating relationships with these tables and various forms of optimization such as hints or data caching to name a couple. This document will not go into detail about vendor specific federation technologies. You will need to refer to their documentation for specifics. The point of this section is to point out that federation technologies exist and may benefit you in terms of optimizing the disparate data sources for reporting in CA.

8 When to Consider Tactical ETL

There may be scenarios where most of the data you require for reporting exists in one database with the exception of a few other key tables. Rather than build a new warehouse to consolidate all the data into one database, perhaps it might make sense to simply extract transform and load (ETL) the data from the other system or systems into the main database of interest. There are several ETL tools on the market.

Consider the generic example in Illustration 11 where Database 1 has two tables joined together, Table 1 and Table 2, and Database 2 has three tables joined together, Table 3, Table 4 and Table 5.

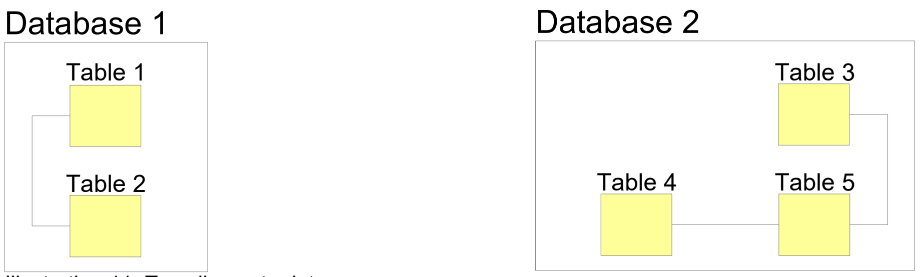


Illustration 11: Two disparate data sources

Imagine that Database 2 has the majority of data required for reporting but requires data from Table 1 located in Database 1. You can ETL the required data from Table 1 into Database 2, as seen in Illustration 12, and join the new table as required to ensure join processing is all done in Database 2.

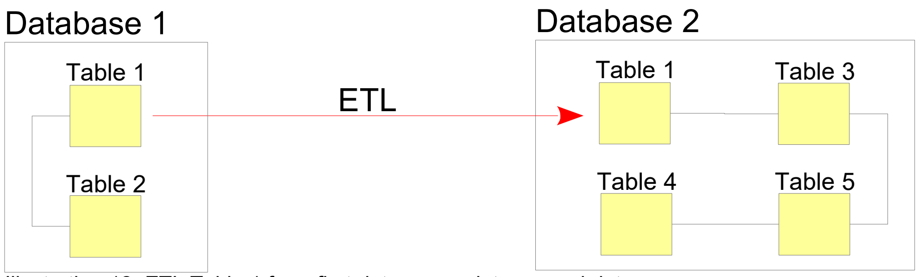


Illustration 12: ETL Table 1 from first data source into second data source

9 Use a Star Schema Data Mart or Warehouse

Up until this point we have looked at methods of disparate database reporting other than data marts or warehouses. This is because it is not always feasible to consolidate enterprise data into a data mart or warehouse. However, if time, expense, and resources are not a road block, the ideal scenario is to work towards a star schema design for a data mart or warehouse using an ETL tool and aggregate data where possible. When your enterprise data is consolidated and structured according to the industry standard star schema design, reporting on the data is easier to accomplish and performance is usually much better. Another benefit is that once the data is in this type of structure, it is easier to extend the data into cubes for OLAP querying since most of the dimensional design (measures and related dimensions) has been done.

This document will not go into details of star schema design. The intent is to provide guidance around what methods are available to consolidate multiple data sources for CA reporting.