IBM Db2 Web Query Adapters

Release 2.4.0

Contents

Chapter 1

Using the Web Console

The Web Console enables you to remotely view and manage the server environment. From a single, easy-to-use interface, you can:

- Select, add, and configure data adapters.
- Create and manage adapter metadata.

The console offers online help with two kinds of links to appropriate sections of this documentation:

- Click *Help* on the menu bar and choose *Contents and Search* to display the full Web Console help system.
- Click the *?* icon, when available, to display contextual help specific to the item.

Chapter 2

Determining SQL Optimization

You can determine which functions are optimized to their SQL counterparts by adapter in the SQL Optimization Report available from the Web Console or Data Management Console.

In this chapter:

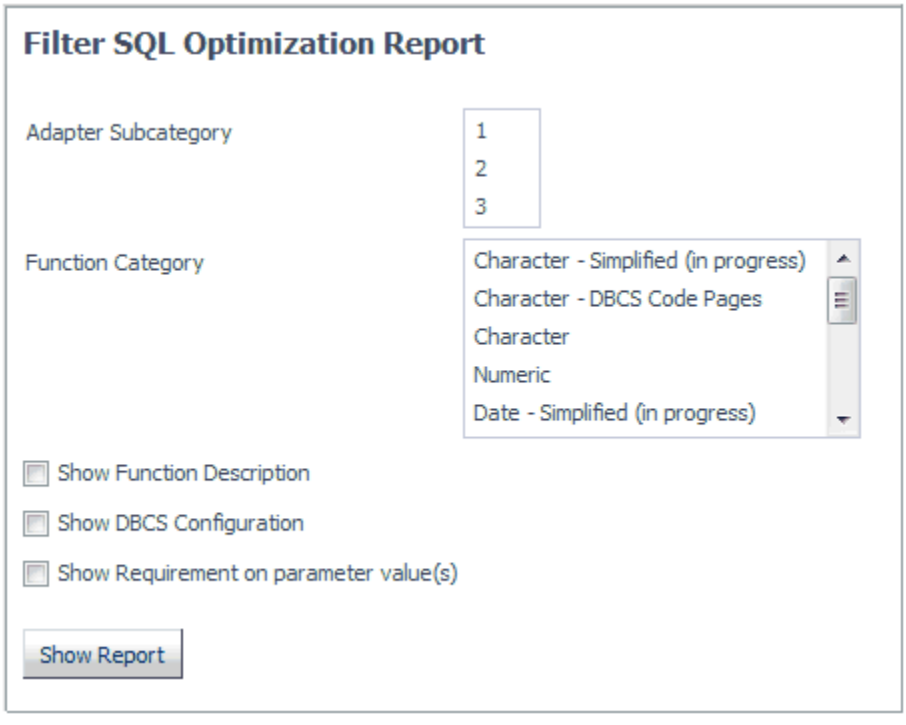
SQL Optimization Report

The following procedure describes how to display the SQL Optimization Report.

Procedure: How to Display the SQL Optimization Report

1. From the Web Console or Data Management Console, select *Adapters*.
2. From the Information section of the ribbon, click the *SQL Optimization Report* button.

The Filter SQL Optimization Report page opens, as shown in the following image.



3. Optionally, select an *Adapter Subcategory* from the corresponding list box.
4. Optionally, select a *Function Category* from the corresponding list box.
5. Optionally, select the *Show Function Description*, *Show DBCS Configuration*, or *Show Requirement on parameter value(s)* check boxes.
6. Click *Show Report*.

The SQL Optimization Report shows the available functions for the specific relational database managment systems.

Note: Some relational database systems on the report may not be available on your system.

Chapter 3

Using the Adapter for Db2/Db2 Warehouse

The Adapter for Db2/Db2 Warehouse allows applications to access Db2 data sources. The adapter converts data or application requests into native Db2 statements and returns optimized answer sets to the requesting program.

The adapter can be used to connect to an IBM Integrated Analytics System (IIAS) system (also known as Sailfish).

The adapter supports the execution of Db2 stored procedures and Db2 Cube Views.

In this chapter:

Using the Adapter for Db2

The Adapter for Db2 allows applications to access Db2 data sources and convert data or application requests into native Db2 statements, returning optimized answer sets to the requesting program. Access to Db2 is available through CLI, CAF, and JDBC versions of the adapter.

The CLI version of the adapter supports the execution of Db2 stored procedures and Db2 Cube Views.

Configuring the Adapter for Db2

In the Web Query environment, the adapter is pre-configured. You can add connections to additional Db2 databases if you wish.

Declaring Connection Attributes

You can enter connection and authentication information from the Web Query environment. This information is stored in the global server profile.

Procedure: How to Add Connections to Db2 Databases From the Web Query Environment

In a Web Query environment, your initial connection is pre-configured to your local host (*LOCAL). However, you can add connections to other Db2 databases as follows:

1. Expand the *Domains* folder, then expand a *domain*.
2. Expand the *Reports* folder, right-click a request subfolder, and choose *Metadata* from the menu.
3. In the left-hand Adapter navigation pane, click the *DB2 cli* folder and choose *Add Connection* from the menu. The Add Connection for DB2 cli pane opens.
4. Enter the required parameter values.

   For a description of these parameters, see *Connection Attributes for Db2 With CLI* on page 14.
5. Click *Configure*.

Reference: Connection Attributes for Db2 With CLI

The following list describes the connection attributes for which you can supply values. To complete the attribute declaration, click the *Configure* button.

Connection name

Logical name used to identify this particular set of connection attributes. The default is CON01.

Datasource/DSN

Db2 data source name (DSN), is a locally registered alias of a remote DB2 database. There is no default data source name. You must enter a value. If you do not want to register the DSN, you can use a DSN-less connection on Linux, UNIX, or Windows, where you specify all of the required parameters. For information, see below.

For IBM i, this is the Remote Database Directory entry or *LOCAL (for local host).

DSN-less

Applies to Linux, UNIX, and Windows. Check this box to specify the following connection parameters instead of a DSN. Note that bulk load is not supported with this type of connection.

Host Name

Is the name of the host where the Db2 server is running.

Port Number

Is the port number on which the Db2 server is listening.

Database Name

Is the Db2 database name.

Security

There are three methods by which a user can be authenticated when connecting to a database server:

- **Explicit.** The user ID and password are explicitly specified for each connection and passed to the database, at connection time, for authentication.
- **Password Passthru.** The user ID and password received from the client application are passed to the database, at connection time, for authentication.
- **Trusted.** The adapter connects to the database using the database rules for an impersonated process that are relevant to the current operating system.

User

Primary authorization ID by which you are known to the data source.

Password

Password associated with the primary authorization ID.

Trusted Context

Select this check box to enable a Db2 trusted context, which is a database object that defines a trust relationship for a connection between the database and an external application server. The trust relationship is based upon the attributes defined in the trusted context object.

If you select Enable, the following is added after the password in the SET CONNECTION ATTRIBUTES command:

```
:'trusted_context=y'
```

For example, assume the Reporting server is configured with LDAP security and the Web Query Client is configured with SECURITY TRUSTED. From the Web Query Client, the Reporting Server browser interface is opened without an authentication prompt and connection to the server is trusted. The adapter connects to the Db2 database with the user ID in the connection string. It checks for a trusted context object and then switches users.

Additional connection string keywords (Optional)

You can enter any additional connection string keywords separated by semicolons (;). Each keyword consists of an attribute and value pair in the form attribute=value.

Select profile

Select a profile from the drop-down menu to indicate the level of profile in which to store the CONNECTION_ATTRIBUTES command. The global profile, edasprof.prf, is the default.

If you wish to create a new profile, either a user profile (*user*.prf) or a group profile if available on your platform (using the appropriate naming convention), choose *New Profile* from the drop-down menu and enter a name in the Profile Name field (the extension is added automatically).

Store the connection attributes in the server profile (edasprof).

Reference: Connection Attributes for Db2 With JDBC

The following list describes the connection attributes for which you can supply values. To complete the attribute declaration, click the *Configure* button.

Connection name

Logical name used to identify this particular set of connection attributes. The default is CON01.

URL

Location URL for the JDBC data source.

Driver name

Name of the JDBC driver.

See the driver documentation for the specific release you are using.

IBI_CLASSPATH

Defines the additional Java Class directories or full-path jar names that will be available for Java Services. The value may be set by editing the communications file or in the Reporting Server browser interface. Using the Reporting Server browser interface, you can enter one reference per line in the input field. When the file is saved, the entries are converted to a single string using colon (:) delimiters for all platforms.

Security

There are two methods by which a user can be authenticated when connecting to a database server:

- **Explicit.** The user ID and password are explicitly specified for each connection and passed to the database, at connection time, for authentication.
- **Password Passthru.** The user ID and password received from the client application are passed to the database, at connection time, for authentication.

User

Primary authorization ID by which you are known to the data source.

Password

Password associated with the primary authorization ID.

Select profile

Select a profile from the drop-down menu to indicate the level of profile in which to store the CONNECTION_ATTRIBUTES command. The global profile, edasprof.prf, is the default.

If you wish to create a new profile, either a user profile (*user*.prf) or a group profile if available on your platform (using the appropriate naming convention), choose *New Profile* from the drop-down menu and enter a name in the Profile Name field (the extension is added automatically).

Store the connection attributes in the server profile (edasprof).

Reference: Connection Attributes for Db2 With SQL

For a full IBM i installation, once the adapter is configured for IBM i you must go to the Change Settings pane and set an isolation level. Failure to set an explicit isolation level causes Db2 to generate multiple commit/rollback warning messages in the adapter's job log. Any isolation level may be selected for the purpose of stopping the multiple warning messages. For more information, see *Controlling Types of Locks on IBM i* on page 38.

Note: In the IBM i Web Query environment, isolation level is preset to NC (No Commit) so no action is required.

Syntax: How to Declare Connection Attributes Manually

Explicit authentication. The user ID and password are explicitly specified for each connection and passed to Db2, at connection time, for authentication.

Password passthru authentication. The user ID and password are explicitly specified for each connection and passed to Db2, at connection time, for authentication.

Trusted authentication. The adapter connects to Db2 as a Windows login using the credentials of the Windows user impersonated by the server data access agent.

The available parameters are:

where:

`DB2`

Indicates the adapter. You can omit this value if you previously issued the SQLENGINE command.

`connection`

Is the logical name used to identify this particular set of connection attributes.

Note that one blank space is required between *connection* and DSN_*name*.

`DSN_`*`name`*

Is the Db2 data source name (DSN) you wish to access. It must match an entry in the odbc.ini file.

`userid`

Is the primary authorization ID by which you are known to Db2.

`password`

Is the password associated with the primary authorization ID.

Overriding the Default Connection

Once connections have been defined, the connection named in the first CONNECTION_ATTRIBUTES command serves as the default connection.

Syntax:

How to Change the Default Connection

You change this setting by expanding the Domains and Reports folders, then right-clicking a subfolder and choosing *Metadata*. Right-click *DB2* and select *Change Settings*. The Change Settings pane opens.

`DB2`

Indicates the adapter. You can omit this value if you previously issued the SQLENGINE command.

`connection`

Is the connection defined in a previously issued CONNECTION_ATTRIBUTES command. If this name was not previously declared, the following message is issued:

```
FOC1671, Command out of sequence
```

Note:

- If you use the DEFAULT_CONNECTION command more than once, the connection name specified in the *last* command serves as the default connection.
- The DEFAULT_CONNECTION command cannot be issued while an uncommitted transaction (LUW) is pending. In that case, the following message is issued:

  ```
  FOC1671, Command out of sequence.
  ```

Controlling Connection Scope

The AUTODISCONNECTcommand controls the persistence of connections when using the adapter.

Controlling Connection Scope With AUTODISCONNECT

AUTODISCONNECT completely detaches the users address space (or task) from Db2. After a DISCONNECT, the task must re-establish its connection to Db2 before performing any data source work. The tasks that frequently issue the DISCONNECT command are connected to Db2 for shorter periods of time, allowing other tasks to connect and acquire threads as needed. However, there is significant system overhead associated with frequently connecting and disconnecting, and the possibility exists that no thread will be immediately available when the task attempts to reconnect.

Syntax: How to Control Connection Scope With AUTODISCONNECT

You change this setting by expanding the Domains and Reports folders, then right-clicking a subfolder and choosing *Metadata*. Right-click *DB2* and select *Change Settings*. The Change Settings pane opens.

`DB2`

Indicates the adapter. You can omit this value if you previously issued the SQLENGINE command.

`FIN`

Disconnects automatically only after the session has been terminated. FIN is the default value.

`COMMIT`

Disconnects automatically only after COMMIT or ROLLBACK is issued as a native SQL command.

Managing Db2 Metadata

When the server accesses a data source, it needs to know how to interpret the data stored there. For each data source the server will access, you create a synonym that describes the structure of the data source and the server mapping of the Db2 data types.

Creating Synonyms

Synonyms define unique names (or aliases) for each Db2 table or view that is accessible from the server. Synonyms are useful because they hide the underlying data source location and identity from client applications. They also provide support for extended metadata features of the server, such as virtual fields and additional security mechanisms.

Using synonyms allows an object to be moved or renamed while allowing client applications to continue functioning without modification. The only modification required is a redefinition of the synonym on the server. The result of creating a synonym is a Master File and an Access File, which represent the server metadata.

Note that creating a synonym for a stored procedure is described with reporting against a stored procedure, in *Generating a Synonym for a Stored Procedure* on page 31.

Procedure: How to Create a Synonym

To create a synonym, you must have previously configured the adapter. You can create a synonym from the Applications or Adapters pages of the Reporting Server browser interface.

1. From the Reporting Server browser interface Applications page, click *Get Data*.
2. Right-click a connection for a configured adapter.

   Depending on the type of adapter you chose, one of the following options appears on the context menu.

   - **Show DBMS objects.** This opens the page for selecting synonym objects and properties.
   - **Create metadata objects.** This opens the page for selecting synonym objects and properties.
   - **Show files.** This opens a file picker. After you choose a file of the correct type, the page for selecting synonym objects and properties opens.
   - **Show local files.** This opens a file picker. After you choose a file of the correct type, the page for selecting synonym objects and properties opens.
3. Enter values for the parameters required by the adapter as described in the chapter for your adapter.
4. After entering the parameter values, click the *Add* button.

   The button may be labeled *Next*, *Create Synonym*, *Create Base Synonyms*, *Create Cluster Synonym*, or *Update Base Synonyms*.

The synonym is created and added under the specified application directory.

Note:

- When creating a synonym, if you choose the Validate check box (where available), the server adjusts special characters and checks for reserved words.

Procedure: How to Create a Synonym From the Web Query Environment

1. Expand the *Domains* folder, then expand a domain.
2. Expand the *Reports* folder, right-click a request subfolder, and choose *Metadata* from the menu.
3. In the left-hand Adapter navigation pane, click the database connection (*LOCAL by default) and choose *Create Synonym* from the menu. The first of a series of synonym creation pages opens.

   Enter values for the parameters required for the adapter.

For information about these parameters, see *Synonym Creation Parameters for Db2* on page 22.

4. After entering parameter values, click *Create Synonym*.

   The synonym is stored in the baseapp directory. You can use any of the available reporting tools to create a report using the generated synonym.

Reference: Synonym Creation Parameters for Db2

The following list describes the synonym creation parameters for which you can supply values.

Object Type

Restrict candidates for synonym creation based on the selected object type(s): Tables, Views, External SQL Scripts, and any other supported objects.

Choosing *External SQL Scripts* from the drop-down list enables you to represent an SQL Query as a synonym for read-only reporting. A Synonym candidate can be any file that contains one (and only one) valid SQL Query and does not contain end-of-statement delimiters (";" or "/") and comments.

Depending on the adapter, you can further restrict your search by choosing

- *Aliases* as a filtering option and the Alias points to a Remote Db2 system, it must be Version 8 or higher. Only one level of Remote Db2 is supported.

Important: If you select Stored Procedures as your object type, the input parameters will be a little different from those described here. For details, see *Reporting Against a Db2 Stored Procedure* on page 31.

Owner/Schema

Selecting this option adds the Owner/Schema and Object Name parameters to the screen. Select an owner/schema from the drop-down list or type a string for filtering the selection, inserting the wildcard character (%) as needed at the beginning and/or end of the string. Then click *Search*. For example, enter: ABC% to select tables or views whose owner/schema begin with the letters ABC; %ABC to select tables or views whose owner/schema end with the letters ABC; %ABC% to select tables or views whose owner/schema contain the letters ABC at the beginning, middle, or end.

Object Name

Selecting this option adds the Owner/Schema and Object Name parameters to the screen. Type a string for filtering the object names, inserting the wildcard character (%) as needed at the beginning and/or end of the string. Then click *Search*. For example, enter: ABC% to select all objects whose names begin with the letters ABC; %ABC to select all whose names end with the letters ABC; %ABC% to select all whose names contain the letters ABC at the beginning, middle, or end.

Filter by Library/Object Name (IBM i)

To avoid the return of an extremely large and potentially unmanageable list, always supply a value for Library or Object Name:

- **Library.** Type a string for filtering the Library/Db2 Schema, inserting the wildcard character (%) as needed at the beginning and/or end of the string. For example, enter: ABC% to select tables or views whose library name begins with the letters ABC; %ABC to select tables or views whose library name ends with the letters ABC; %ABC% to select tables or views whose library name contains the letters ABC at the beginning, middle, or end.
- **Object name.** Type a string for filtering the table, view, or object names, inserting the wildcard character (%) as needed at the beginning and/or end of the string. For example, enter: ABC% to select all tables, views, or objects whose names begin with the letters ABC; %ABC to select all whose names end with the letters ABC; %ABC% to select all whose names contain the letters ABC at the beginning, middle, or end.

Note: When creating synonyms for Db2 objects on the IBM i platform, the SQL naming convention that allows for long names is used. For table objects, this means the TABLE_NAME (not the SYSTEM_TABLE_NAME) as defined in the system catalog QSYS2/SYSTABLES is displayed in all dialogs.

Location of External SQL Scripts

If you specify *External SQL Scripts* in the *Restrict Object type to* field, these additional fields are displayed.

The following standard naming conventions apply for UNIX, IBM i IFS, and z/OS HFS:

- In the *Base Location* field, specify the physical directory location of the file that contains the SQL Query. You can type a directory name or click on the ellipsis. This opens the Select Base Location dialog box.
- In the *Document Name* field, enter the file name with or without wild card characters.
- In the *Document Extension* field, enter the extension of the script files to filter the list of candidates.

On IBM i, you can use alternative IFS naming conventions to access library members. The following entry illustrates this method:

- In the *Base Location* field, enter:

  `/QSYS.LIB/MYLIBRARY.LIB/MYSRC.FILE`

- The *Document Extension* is understood to be MBR. You can enter this value explicitly or leave the input box blank.

During synonym generation, the adapter issues native API calls to obtain a list of elements in the select list and builds the Master File with a field for each element. The generated Access File references the location of the SQL script in the DATASET attribute, which contains the full path, including the name and extension of the file containing the SQL Query. For example,

```
DATASET=/ul/home2/apps/report3.sql
```

When a Web Query report is created, the SQL Query is used to access data.

Row Limit

Select the number of objects to display on the Create Synonym page.

For Subquery

Only available when *External SQL Scripts* is selected from the object type drop-down menu. When selected, a SUBQUERY keyword is added to the Access File of the generated synonym. If the corresponding SQL string has valid syntax that can be used in the FROM statement of the generated SQL (what is known as a Derived Table), then the SQL SCRIPT will be processed as a subquery embedded into a FROM clause. This usage allows for more flexibility. For example, the synonym can be used as a target for a JOIN.

If the SQL SCRIPT has parameter markers, such as ? or :, or the syntax contains constructs that are invalid for a derived table, for example ORDER BY, then this keyword should not be selected. At runtime, if SUBQUERY=Y is present and it is determined that the SQL SCRIPT cannot be used in the FROM statement, the setting will be ignored, and a FOC1782 warning message will be issued. The default is selected (SUBQUERY=Y).

Application

The default value is baseapp.

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 the prefix and suffix fields blank.

Customize data type mappings

To change the data type mappings from their default settings, select this check box. The customizable mappings are displayed.

For information about customizable mappings, see *Data Type Support Report* on page 124.

Create: Cluster Synonym or Base Synonyms

Select the button for the type of synonym you want to create.

One-part name (IBM i)

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 for a TABLE or VIEW:

```
TABLENAME=MYLIB/MYTABLE
```

For a Stored Procedure, this would be a Db2 RPC and identified in the Access File as:

```
STPNAME=MYLIB/MYPROC
```

With this explicit type of entry in the Access File, at run-time the library is directly referenced and the object opened.

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 as follows:

```
TABLENAME=MYTABLE
```

or

```
STPNAME=MYPROC
```

With this type of entry in the Access File (one-part name), the Db2 library search path will be used at run time. The exact composition varies depending on if Db2 is configured as a CLI or SQL connection.

For CLI configured servers, the search path will be the default library of Db2 for a CLI connected user (usually QSYS and QSYS2 plus the default library of the user). It is also controllable by use of a passthru command setting the path, such as SLQ DB2 SET PATH “QSYS”,”QSYS2”,”FOO”,”EXTRAFOO” as issued within a requests FOCEXCEC procedure or a profile. For SQL configured servers, the search path will be *LIBL. The library path of the process can be controlled by commands, such as ADDLIBLE, either before server start or as issued within a requests FOCEXCEC procedure or a profile.

Default Synonym Name

This column displays the name that will be assigned to each synonym. To assign a different name, replace the displayed value.

Owner/Schema

The user account that created the object or a collection of objects owned by a user.

Fact or Dimension

You can check Fact or Dimension to generate the segment as a fact table or a dimension segment. If you are creating a cluster synonym, you can right-click a selected fact table and select *Show Related Dimensions* or *Add Related Dimensions* to show a list of related dimensions or add related dimensions to the synonym.

Table name

Is the name of the underlying object.

Type

The object type (Table, View, and so on).

Select tables

Select tables for which you wish to create synonyms:

- When creating base synonyms, you can select all tables in the list by clicking the *Select All* check box.
- To select specific tables, select the corresponding check boxes.

Once you have selected the objects for which you want to create synonyms, click the *Create Base Synonyms* or *Create Cluster Synonym* button on the ribbon.

Example: Sample Generated Synonym

An Adapter for Db2 synonym comprises a Master File and an Access File. This is a synonym for the table nf29004.

Generated Master File nf29004.mas

```
FILE=DIVISION, SUFFIX=DB2 ,$
SEGNAME=SEG1_4, SEGTYPE=S0 ,$
FIELD=DIVISION4,    DIVISION4,    I9,  I4  ,MISSING=OFF ,$
FIELD=DIVISION_NA4, DIVISION_NA4, A25, A25 ,MISSING=ON  ,$
FIELD=DIVISION_HE4, DIVISION_HE4, I9,  I4  ,MISSING=ON  ,$
```

Generated Access File nf29004.acx

```
SEGNAME=SEG1_4,TABLENAME=EDAQA.NF29004,
CONNECTION=DB1,KEYS=1,WRITE=YES,$
```

Reference: Mapping Db2 Table and Column Attributes Into a Synonym

These mappings are OS-system specific:

Mapping Db2 Table Attributes

Platform	Db2 Table Attribute	Mapping in Synonym	Notes
UNIX	COMMENT	REMARKS	

Platform	Db2 Table Attribute	Mapping in Synonym	Notes
IBM i	REMARK LABEL	REMARKS	The synonym creation facility picks up a Db2 table TABLE level REMARK or LABEL value, whichever is not null, for use as a Master File REMARKS= value. If the Db2 table has both a populated REMARK and a populated LABEL, the REMARK value will be used.
z/OS	COMMENT LABEL	REMARKS	The synonym creation facility picks up a Db2 table TABLE level COMMENT or LABEL value, whichever is not blank, for use as a Master File REMARKS= value. If the Db2 table has both a populated COMMENT and a populated LABEL, the COMMENT value will be used.

Mapping Db2 Column Attributes

Platform	Db2 Column Attribute	Mapping in Synonym	Notes
UNIX	COMMENT	DESCRIPTION	
IBM i	LABEL COMMENT	TITLE DESCRIPTION	
z/OS	LABEL COMMENT	TITLE DESCRIPTION	

Reference: Access File Keywords

This chart describes the keywords in the Access File.

Keyword	Description
SEGNAME	Value must be identical to the SEGNAME value in the Master File.
TABLENAME	Name of the table or view. This value can include a location or owner name as follows: For IBM i, the syntax is: `TABLENAME=[library/]tablename`
CONNECTION	Indicates a previously declared connection. The syntax is: `CONNECTION=connection` `CONNECTION=' '` indicates access to the local database server. Absence of the CONNECTION attribute indicates access to the default database server.
KEYS	Indicates how many columns constitute the primary key for the table. Corresponds to the first *n* fields in the Master File segment. This attribute requires the columns that constitute the key to be described first in the Master File. See the KEY attribute below for information about specifying the key fields without having to describe them first in the Master File.
KEY	Specifies the columns that participate in the primary key without having to describe them as the first fields in the Master File. The syntax is: `KEY=fld1/fld2/.../fldn`
WRITE	Specifies whether write operations are allowed against the table.

Keyword	Description
KEYFLD IXFLD	Supply the names of the primary key and foreign key fields that implement the relationships established by the multi-table Master File. Together, KEYFLD and IXFLD identify the field shared by a related table pair. ❏ KEYFLD is the FIELDNAME of the common column from the parent table. ❏ IXFLD is the FIELDNAME of the common column from the related table. KEYFLD and IXFLD must have the same data type. It is recommended, but not required, that their lengths also be the same. **Note:** An RDBMS index on both the KEYFLD and IXFLD columns provides the RDBMS with a greater opportunity to produce efficient joins. The columns must have the same data type. If their length is the same, the RDBMS handles the join more efficiently.
INDEX_NAME INDEX_UNIQUE INDEX_COLUMNS INDEX_ORDER	Indicate a name of the index in a database, uniqueness, name, and order of the indexed column(s).

Reference: **Managing Synonyms**

Once you have created a synonym, you can right-click the synonym name in the navigation pane to access the available options.

Db2 Data Type Support

SQL Data Type mapping options are available in a report available from the Reporting Server browser interface.

Reporting Against a Db2 Stored Procedure

You can use a reporting tool, such as a SELECT statement or TABLE command, to execute Db2 stored procedures and report against the output parameters and answer set of a procedure. Among the benefits of this method of executing a stored procedure are:

- The retrieval of output parameters, OUT parameters, and INOUT parameters in OUT mode, as well as the answer set. (Other methods of invocation retrieve the answer set only.)
- The ease with which you can process, format, and display output parameters and the answer set, using TABLE and other reporting tools.

To report against a stored procedure:

1. **Generate a synonym** for the stored procedure answer set, as described in *Generating a Synonym for a Stored Procedure* on page 31.
2. **Create a report procedure,** as described in *Creating a Report Against a Stored Procedure* on page 35.
3. **Run the report.** This executes the stored procedure and reports against any output parameters (OUT and INOUT in OUT mode), and any answer set fields, specified in the report.

Generating a Synonym for a Stored Procedure

A synonym describes the stored procedure parameters and answer set.

An answer set structure may vary depending on the input parameter values that are provided when the procedure is executed. Therefore, you need to generate a separate synonym for each set of input parameter values that will be provided when the procedure is executed at run time. For example, if users may execute the stored procedure using three different sets of input parameter values, you need to generate three synonyms, one for each set of values. (Unless noted otherwise, "input parameters" refers to IN parameters and to INOUT parameters in IN mode.)

There is an exception: if you know the internal logic of the procedure, and are certain which range of input parameter values will generate each answer set structure returned by the procedure, you can create one synonym for each answer set structure, and for each synonym simply provide a representative set of the input parameter values necessary to return that answer set structure.

A synonym includes the following segments:

- INPUT, which describes any IN parameters and INOUT parameters in IN mode.

  If there are no IN parameters or INOUT parameters in IN mode, the segment describes a single dummy field.

- OUTPUT, which describes any OUT parameters and INOUT parameters in OUT mode.

  If there are no OUT parameters or INOUT parameters in OUT mode, the segment is omitted.

- ANSWERSET*n*, one for each answer set.

  If there is no answer set, the segment is omitted.

In IBM i, the metadata of a stored procedure that has an answer set will include column descriptions. These will appear as title fields in the synonym, as shown in the following example:

```
FILENAME=TEST_DESCRIPTIONS, SUFFIX=DB2 , $
  SEGMENT=INPUT, SEGTYPE=S0, $
    FIELDNAME=, ALIAS=DUMMY, USAGE=A1, ACTUAL=A1, MISSING=ON, $
  SEGMENT=ANSWERSET1, SEGTYPE=S0, PARENT=INPUT, $
    FIELDNAME=STORECODE, ALIAS=STORECODE, USAGE=A6, ACTUAL=A6,
      TITLE='Store,Code', $
    FIELDNAME=STORENAME, ALIAS=STORENAME, USAGE=A30, ACTUAL=A30,
      MISSING=ON, TITLE='Store,Name', $
    FIELDNAME=COUNTRY, ALIAS=COUNTRY, USAGE=A15, ACTUAL=A15,
      MISSING=ON, TITLE='Country', $
    FIELDNAME=REGION, ALIAS=REGION, USAGE=A25, ACTUAL=A25,
      MISSING=ON, TITLE='Region', $
```

Example: Synonym for Db2 Stored Procedure CustOrders

The following synonym describes a stored procedure, DB2SPR04 SP, with IN/OUT OUTPUT parameters.

```
FILENAME=SDB2SPR04, SUFFIX=DB2     , $
  SEGMENT=INPUT, SEGTYPE=S0, $
    FIELDNAME=PARM1, ALIAS=P0001, USAGE=I11, ACTUAL=I4,
      MISSING=ON, ACCESS_PROPERTY=(NEED_VALUE), $
    FIELDNAME=PARM2, ALIAS=P0002, USAGE=I6, ACTUAL=I4,
      MISSING=ON, ACCESS_PROPERTY=(NEED_VALUE), $
    FIELDNAME=PARM5, ALIAS=P0005, USAGE=YYMD, ACTUAL=DATE,
      MISSING=ON, ACCESS_PROPERTY=(NEED_VALUE), $
    FIELDNAME=PARM6, ALIAS=P0006, USAGE=HHIS, ACTUAL=HHIS,
      MISSING=ON, ACCESS_PROPERTY=(NEED_VALUE), $
    FIELDNAME=PARM7, ALIAS=P0007, USAGE=HYYMDm, ACTUAL=HYYMDm,
      MISSING=ON, ACCESS_PROPERTY=(NEED_VALUE), $
    FIELDNAME=PARM11, ALIAS=P0011, USAGE=P17.5, ACTUAL=P8,
      MISSING=ON, ACCESS_PROPERTY=(NEED_VALUE), $
    FIELDNAME=PARM13, ALIAS=P0013, USAGE=P20, ACTUAL=P10,
      MISSING=ON, ACCESS_PROPERTY=(NEED_VALUE), $
    FIELDNAME=PARM15, ALIAS=P0015, USAGE=F9.2, ACTUAL=F4,
      MISSING=ON, ACCESS_PROPERTY=(NEED_VALUE), $
    FIELDNAME=PARM16, ALIAS=P0016, USAGE=D20.2, ACTUAL=D8,
      MISSING=ON, ACCESS_PROPERTY=(NEED_VALUE), $
```

```
SEGMENT=OUTPUT, SEGTYPE=S0, PARENT=INPUT, $
  FIELDNAME=RETURN_CODE, ALIAS=P0000, USAGE=I11, ACTUAL=I4,
    MISSING=ON, TITLE='Return Code', $
  FIELDNAME=PARM1, ALIAS=P0001, USAGE=I11, ACTUAL=I4, MISSING=ON, $
  FIELDNAME=PARM2, ALIAS=P0002, USAGE=I6, ACTUAL=I4, MISSING=ON, $
  FIELDNAME=PARM3, ALIAS=P0003, USAGE=I11, ACTUAL=I4, MISSING=ON, $
  FIELDNAME=PARM4, ALIAS=P0004, USAGE=I6, ACTUAL=I4, MISSING=ON, $
  FIELDNAME=PARM5, ALIAS=P0005, USAGE=YYMD, ACTUAL=DATE, MISSING=ON, $
  FIELDNAME=PARM6, ALIAS=P0006, USAGE=HHIS, ACTUAL=HHIS,
    MISSING=ON, $
  FIELDNAME=PARM7, ALIAS=P0007, USAGE=HYYMDm, ACTUAL=HYYMDm,
    MISSING=ON, $
  FIELDNAME=PARM8, ALIAS=P0008, USAGE=YYMD, ACTUAL=DATE, MISSING=ON, $
  FIELDNAME=PARM9, ALIAS=P0009, USAGE=HHIS, ACTUAL=HHIS, MISSING=ON, $
  FIELDNAME=PARM10, ALIAS=P0010, USAGE=HYYMDm, ACTUAL=HYYMDm,
    MISSING=ON, $
  FIELDNAME=PARM11, ALIAS=P0011, USAGE=P17.5, ACTUAL=P8, MISSING=ON, $
  FIELDNAME=PARM12, ALIAS=P0012, USAGE=P17.5, ACTUAL=P8, MISSING=ON, $
  FIELDNAME=PARM13, ALIAS=P0013, USAGE=P20, ACTUAL=P10, MISSING=ON, $
  FIELDNAME=PARM14, ALIAS=P0014, USAGE=P20, ACTUAL=P10, MISSING=ON, $
  FIELDNAME=PARM15, ALIAS=P0015, USAGE=F9.2, ACTUAL=F4, MISSING=ON, $
  FIELDNAME=PARM16, ALIAS=P0016, USAGE=D20.2, ACTUAL=D8, MISSING=ON, $
  FIELDNAME=PARM17, ALIAS=P0017, USAGE=F9.2, ACTUAL=F4, MISSING=ON, $
  FIELDNAME=PARM18, ALIAS=P0018, USAGE=D20.2, ACTUAL=D8, MISSING=ON, $
```

Reference: Synonym Creation Parameters for Stored Procedures

The following list describes the synonym creation parameters for a stored procedure.

Object Type

Select *Stored Procedures*.

Owner/Schema and Object name (all platforms except IBM i)

Selecting these options adds the Owner/Schema and Object Name parameters to the screen.

- **Owner/Schema.** Type a string for filtering the selection, inserting the wildcard character (%) as needed at the beginning and/or end of the string. For example, enter: ABC% to select tables or views whose owner/schema begin with the letters ABC; %ABC to select tables or views whose owner/schema end with the letters ABC; %ABC% to select tables or views whose owner/schema contain the letters ABC at the beginning, middle, or end.

- **Object name.** Type a string for filtering the procedure names, inserting the wildcard character (%) as needed at the beginning and/or end of the string. For example, enter: ABC% to select all procedures whose names begin with the letters ABC; %ABC to select all whose names end with the letters ABC; %ABC% to select all whose names contain the letters ABC at the beginning, middle, or end.

Library

Object Name (IBM i)

To avoid the return of an extremely large and potentially unmanageable list, always supply a value for Library or Object Name:

- **Library.** Type a string for filtering the Library (or Db2 Collection), inserting the wildcard character (%) as needed at the beginning and/or end of the string. For example, enter: ABC% to select tables or views whose owner IDs begin with the letters ABC; %ABC to select tables or views whose owner IDs end with the letters ABC; %ABC% to select tables or views whose owner IDs contain the letters ABC at the beginning, middle, or end.
- **Object name.** Type a string for filtering the table, view, or object names, inserting the wildcard character (%) as needed at the beginning and/or end of the string. For example, enter: ABC% to select all tables, views, or objects whose names begin with the letters ABC; %ABC to select all tables, views, or objects whose names end with the letters ABC; %ABC% to select all tables, views, or objects whose names contain the letters ABC at the beginning, middle, or end.

Select

Select a procedure. You may only select one procedure at a time since each procedure will require unique input in the Values box on the next synonym creation pane.

Synonym Name

The name of the synonym, which defaults to the stored procedure name.

Application

The default value is baseapp.

Prefix/Suffix

If you have stored procedures with identical names, assign a prefix or a suffix to distinguish their corresponding synonyms. Note that the resulting synonym name cannot exceed 64 characters.

If all procedures have unique names, leave the prefix and suffix fields blank.

Customize data type mappings

To change the data type mappings from their default settings, select this check box. The customizable mappings are displayed.

Creating a Report Against a Stored Procedure

You can report against a stored procedure's answer set using the same facilities you use to report against a database table:

- **SQL SELECT statement.** For syntax, see *How to Report Against a Stored Procedure Using SELECT* on page 36.
- **TABLE and GRAPH commands.** For syntax, see *How to Report Against a Stored Procedure Using the TABLE Command* on page 35.

When joining from or to a stored procedure answer set, you can:

- **Join from** only OUTPUT and ANSWERSET segments in a host file.
- **Join to** only INPUT segments in a cross-referenced file.

Syntax: How to Report Against a Stored Procedure Using the TABLE Command

To execute a stored procedure using the TABLE command, use the following syntax

```
TABLE FILE synonym
PRINT [parameter [parameter] ... | *]
[IF in-parameter EQ value]
  .
  .
  .
END
```

where:

synonym

Is the synonym of the stored procedure you want to execute.

parameter

Is the name of a parameter whose values you want to display in the report. You can specify input parameters, output parameters, or input and output parameters.

If the stored procedure does not require parameters, specify an asterisk (*). This displays a dummy segment, created when the synonym is generated, to satisfy the structure of the SELECT statement.

*

Indicates that you want to display all indicated parameters, or that there are no required parameters.

`IF`

Is an IF or WHERE keyword. Use this to pass a value to an IN parameter or an INOUT parameter in IN mode.

`in-parameter`

Is the name of an IN parameter, or INOUT parameter in IN mode, to which you want to pass a value.

Note: The length of in-parameters cannot exceed 1000 characters if the adapter is configured for Unicode support.

`value`

Is the value you are passing to a parameter.

Syntax: How to Report Against a Stored Procedure Using SELECT

```
SQL
SELECT [parameter [,parameter] ... | *] FROM synonym
[WHERE in-parameter = value]
  .
  .
  .
END
```

where:

`synonym`

Is the synonym of the stored procedure that you want to execute.

`parameter`

Is the name of a parameter whose values you want to display in the report. You can specify input parameters, output parameters, or input and output parameters.

If the stored procedure does not require parameters, enter an asterisk (*) in the syntax. This displays a dummy segment, created during synonym generation, to satisfy the structure of the SELECT statement.

`*`

Indicates that you want to display all indicated parameters, or that there are no required parameters.

`WHERE`

Is used to pass a value to an IN parameter or an INOUT parameter in IN mode. You must specify the value of each parameter on a separate line.

in-parameter

Is the name of an IN parameter, or INOUT parameter in IN mode, to which you want to pass a value.

value

Is the value you are passing to a parameter.

Customizing the Db2 Environment

The Adapter for Db2 provides several parameters for customizing the environment and optimizing performance. This topic provides an overview of customization options.

Controlling the Types of Locks

You can use the ISOLATION command to specify the isolation level of transactions created by the adapter. The isolation level controls the types of locks for objects referenced in the requests executed within the transaction.

If you are working in the IBM i environment, see *Controlling Types of Locks on IBM i* on page 38 for OS-specific variations of SET ISOLATION syntax. For related information about another use for the ISOLATION syntax, see *Creating and Updating Db2 Files on IBM i* on page 38.

Syntax: How to Control the Lock Type

You change this setting by expanding the Domains and Reports folders, then right-clicking a subfolder and choosing *Metadata*. Right-click *Db2* and select *Change Settings*. The Change Settings pane opens.

The available parameters are:

DB2

Indicates the adapter. You can omit this value if you previously issued the SQLENGINE command.

level

Sets the Db2 isolation level, which is mapped to the IBM RDBMS isolation level. If you do not specify an isolation level, the level is reset to the adapter default.

Db2 Isolation Level (CLI)	IBM RDBMS Isolation Level
RC (SQL_TXN_READ_COMMITTED)	CS (Cursor Stability)
SE (SQL_TXN_SERIALIZABLE_READ)	RR (Repeatable Read)

Db2 Isolation Level (CLI)	IBM RDBMS Isolation Level
RR (SQL_TXN_REPEATABLE_READ)	RS (Read Stability)
RU (SQL_TXN_READ_UNCOMMITTED)	UR (Uncommitted Read)
NC (SQL_TXN_NO_COMMIT)	Not applicable

- **RC.** Releases shared locks as the cursor moves on in the table. Use for read-only requests. RC is the default value. Maps to CS (Cursor Stability).
- **SE.** Locks the retrieved data until it is released by an SQL COMMIT WORK or SQL ROLLBACK WORK statement. Maps to RR (Repeatable Read).
- **RR.** Maps to RS (Read Stability). For more information, see the *Db2 Command and Utility Reference*.
- **RU.** Provides read-only access to records even if they are locked. However, these records may not yet be committed to the data source. Maps to UR (Uncommitted Read).
- **NC.** Commit and rollback operations have no effect on SQL statements. Any changes are effectively committed at the end of each successful change operation and can be immediately accessed. For more information, see *Creating and Updating Db2 Files on IBM i* on page 38 and the *Db2 Command and Utility Reference*.

Note:

- The adapter does not validate the isolation level values. If you issue the ISOLATION command with an invalid value, the adapter will not return a message.
- To display the isolation level setting, issue the ENGINE DB2 ? CONNECTINFO query command.

Reference: Controlling Types of Locks on IBM i

For the IBM i Web Query environment, the isolation level is preset to NC (no commit) so no action is required.

Reference: Creating and Updating Db2 Files on IBM i

The following information applies to HOLD FORMAT DB2 and procedures that update a Db2 object in a non-journaled collection.

While Db2 on IBM i supports CREATE TABLE operations to a non-journaled collection (a library with no journal receivers), Db2 normally considers this a commitment control error and issues an error message. When a HOLD FORMAT DB2 command is issued, the same error condition triggers an error message to the adapter. In response, the adapter creates the table, but does not perform the load step. However, if the server is configured with Db2 as a CLI-based adapter, you can use the ISOLATION setting of NC (No Commit) to prevent Db2 from triggering the error message, thereby enabling the table to load.

You can set the NC option server wide from the Adapter for Db2's Change Settings pane. (To access this pane, expand the Domains folder and the Reports folder, then click a request subfolder and choose *Metadata*, *Db2*, *Change Settings*. The Change Settings pane opens.)

After completing this task, revert to the original ISOLATION setting, if appropriate.

Note: An error message like the following will be generated if you issue a HOLD FORMAT DB2 ... DROP command but do not have the authority to DROP an existing file.

```
(FOC1400) SQLCODE IS -601 (HEX: FFFFFDA7): [42710] ABC in XYZ type *FILE
already exists.

(FOC1414) EXECUTE IMMEDIATE ERROR.
```

- If you receive this message, make sure that you have DB2 DROP/CREATE authority and reissue the command.
- When connecting from a non-IBM i platform through a local Db2 client, the SET isolation does not work. You can use the following specific platform syntax.

Activating NONBLOCK Mode

The Adapter for Db2 has the ability to issue calls in NONBLOCK mode. The default behavior is BLOCK mode.

This feature allows the adapter to react to a client request to cancel a query while the adapter is waiting on engine processing. This wait state usually occurs during SQL parsing, before the first row of an answer set is ready for delivery to the adapter or while waiting for access to an object that has been locked by another application.

Syntax: How to Activate NONBLOCK Mode

You change this setting by expanding the Domains and Reports folders, then right-clicking a subfolder and choosing *Metadata*. Right-click *Db2* and select *Change Settings*. The Change Settings pane opens.

The available parameters are:

DB2

Indicates the adapter. You can omit this value if you previously issued the SQLENGINE command.

n

Is a positive numeric number. 0 is the default value, which means that the adapter will operate in BLOCK mode. A value of 1 or greater activates the NONBLOCK calling and specifies the time, in seconds, that the adapter will wait between each time it checks to see if the:

- Query has been executed.
- Client application has requested the cancellation of a query.
- Kill Session button on the Reporting Server browser interface is pressed.

Note: A value of 1 or 2 should be sufficient for normal operations.

Controlling Column Names

You can use the NOCOLUMNTITLE command to control the column names in a report when executing a stored procedure.

Syntax: How to Control Column Names

You change this setting by expanding the Domains and Reports folders, then right-clicking a subfolder and choosing *Metadata*. Right-click *Db2* and select *Change Settings*. The Change Settings pane opens.

DB2

Indicates the adapter. You can omit this parameter value if you previously issued the SET SQLENGINE command.

ON

Uses generated column names (for example, E01, E02, and so on) instead of the column names returned by Db2.

OFF

Uses the column names returned by Db2. OFF is the default value.

Obtaining the Number of Rows Updated or Deleted

PASSRECS returns the number of rows affected by a successfully executed SQL Passthru INSERT, UPDATE, or DELETE command.

Syntax: How to Obtain the Number of Rows Updated or Deleted

You change this setting by expanding the Domains and Reports folders, then right-clicking a subfolder and choosing *Metadata*. Right-click *Db2* and select *Change Setting*s. The Change Settings pane opens.

DB2

Indicates the adapter. You can omit this value if you previously issued the SQLENGINE command.

ON

Provides the number of rows affected in the application program SCB count member after the successful execution of an SQL Passthru INSERT, UPDATE, or DELETE command. ON is the default value.

OFF

Provides no information after the successful execution of an SQL Passthru INSERT, UPDATE, or DELETE command.

Note: This behavior applies for all supported versions of Db2 up to Version 9 on all platforms. In Db2 Version 9 on z/OS, the PASSRECS command does not provide a row count for mass DELETE operations (that is, DELETE statements without a WHERE clause). However, you can obtain a row count in this situation by adding a WHERE phrase to your report request. For example, the following request would generate a delete count:

Setting Naming Conventions

Controls the separator character used for interpreting multipart table, view, and RPC names. This option is typically used to allow a request coded with "." as the separator to be run on systems that use "/" as the separator (for example, IBM i).

This setting is only valid for use with a server configured as CLI. (This applies to most platforms, including IBM i if so configured). If the server needs to support both naming conventions, you can configure an additional service block with a service block profile that uses the desired naming setting and sends requests to the desired block.

Syntax: How to Set Naming Conventions

You change this setting by expanding the Domains and Reports folders, then right-clicking a subfolder and choosing *Metadata*. Right-click *Db2* and select *Change Setting*s. The Change Settings pane opens.

The available parameters are:

DB2

Indicates the adapter. You can omit this value if you previously issued the SQLENGINE command.

SQL

Standard "." character is used as separator in multipart table names.

SYS

System "/" character is used as separator in multipart table names.

Controlling HOLD DBMS Creation

An extension of the HOLD AS *app/name* FORMAT DB2 syntax enables you to exercise more precise control over the creation of HOLD DBMS files.

Syntax: How to Control DBMS Creation

```
HOLD AS app/name FORMAT DB2 [TABLENAME dbms_name][CONNECTION conn_name]
  [DROP]
```

where:

dbms_name

Is the DBMS table to create. It may be a one, two, or three part name, using the separator appropriate to the DBMS (typically "." (dot)).

For IBM i, a "/" (slash) is the separator, unless the NAMING SQL command is being used to reset the separator character.

Note: For Db2 running on IBM i, if the TABLENAME parameter and value are omitted, the default location for the resulting table is the user's default login library. This applies to adapters configured for SQL and CLI.

conn_name

Is the DBMS connection name. When multiple DBMS connections have been configured and are in use, conn_name specifies which connection to use.

`DROP`

Drops the table before creation. This option enables you to delete either a known table or one that was created and stored in a temporary space when persistValue=*global_temporary*. Without the DROP option, if a table already exists, an error occurs when the table is created, resulting in failure to load the table.

If no table exists, this option is ignored.

Db2 Optimization Settings

Adapter optimization allows the RDBMS to perform the work for which it is best suited, reducing the volume of RDBMS-to-server communication and improving response time. It also enables the RDBMS to exploit its own internal optimization techniques.

Improving Efficiency With Aggregate Awareness

Aggregate awareness substantially improves the efficiency of queries.

Syntax: **How to Set Aggregate Awareness**

```
SET AGGREGATE_AWARENESS {FRESH_ONLY|OLD_OK|OFF}
```

where:

`FRESH_ONLY`

Sets different values for the parameters associated with each RDBMS.

`OLD_OK`

Sets different values for the parameters associated with each RDBMS.

`OFF`

If no option is selected, the behavior of the target RDBMS is determined by the database configuration options. There is no default for this setting.

Using Db2 Cube Views

The Adapter for Db2 supports Cube Views as an object type. You can specify Cubes as the object type for which you want to create synonyms.

Before you can use the Adapter for Db2 with Cube Views, you must define Db2 connections for the database(s) containing the Db2 cube views.

For details, see *Creating Synonyms* on page 20.

Mapping Metadata for Db2 Cubes Views

Synonyms are generated for Cubes (not Cube Models). These synonyms reflect the structure of the cube and its attributes and measures, rather than the structure of the underlying Db2 tables and their columns.

The Master File synonym component has several segments:

- The root segment corresponds to the Cube Facts.
- The children segments correspond to the Cube Hierarchies.

The fields in the root segment reflect the Cube Measures. Field names and titles are generated based on measure business names, while aliases are generated based on measure names.

- The PROPERTY attribute for these fields contains the word Measure (or Calculated Measure for calculated measures).
- The REFERENCE attribute contains the aggregated expression for measures that are based on simple columns (not expressions) and have aggregations that match those available in the DML.

Example: Sample Db2 Cube View Master File

```
FILENAME=BASEAPP/COMPLEX_MEASURES__C_, SUFFIX=DB2CV   , $
  SEGMENT=SALESCF, SEGTYPE=S0, $
    FIELDNAME=4ADDS, ALIAS=4adds, USAGE=I11, ACTUAL=I4, MISSING=ON,
      TITLE='4adds', REFERENCE=CNT.4ADDS, PROPERTY=MEASURE,  $
    FIELDNAME=POWEROFADDS, ALIAS=powerofadds, USAGE=D21.2, ACTUAL=D8,
      MISSING=ON, TITLE='powerofadds', PROPERTY=MEASURE,  $
    FIELDNAME=PROFIT_CONST, ALIAS=profit*const, USAGE=D21.2, ACTUAL=D8,
      MISSING=ON, TITLE='profit*const', PROPERTY=MEASURE,  $
    FIELDNAME=RANDOM, ALIAS=random, USAGE=I11, ACTUAL=I4, MISSING=ON,
      TITLE='random', REFERENCE=CNT.RANDOM, PROPERTY=MEASURE,  $
    FIELDNAME=3_COLS_WITH_MIXTUREOF_FUNC,
      ALIAS='3 cols with mixtureof FUNC', USAGE=D21.2, ACTUAL=D8,
      MISSING=ON, TITLE='3 cols with mixtureof FUNC',
      PROPERTY=MEASURE,  $
    FIELDNAME=ROUND_OF_ANOTHER_MEAS, ALIAS='round of another meas',
      USAGE=D21.2, ACTUAL=D8, MISSING=ON, TITLE='round of another meas',
      PROPERTY=MEASURE,  $
```

```
SEGMENT=PRODUCT__CH_, SEGTYPE=U, PARENT=SALESCF, $
  FIELDNAME=PRODUCT_GROUP_ID, ALIAS=product_group_ID, USAGE=I11,
    ACTUAL=I4, MISSING=ON, TITLE='product_group_ID',
    WITHIN='*product (CH)', $
  FIELDNAME=PRODUCT_LINE_ID, ALIAS=product_line_ID, USAGE=I11, ACTUAL=I4,
    MISSING=ON, TITLE='product_line_ID', WITHIN=PRODUCT_GROUP_ID, $
  FIELDNAME=PRODUCT_LINE_NAME,
    ALIAS='12  product_line_ID   product_line_name', USAGE=A25V,
    ACTUAL=A25V, MISSING=ON, TITLE='product_line_name',
    REFERENCE=PRODUCT_LINE_ID, PROPERTY=CAPTION,  $

SEGMENT=STORE__CH_, SEGTYPE=U, PARENT=SALESCF, $
  FIELDNAME=STORE_LOCATION_ID, ALIAS='**** store_location_ID_1',
    USAGE=I11, ACTUAL=I4, MISSING=ON, TITLE='store_location_ID',
    WITHIN='*store (CH)', $
  FIELDNAME=STORE_ADDRESS,
    ALIAS='store_location_ID_1   _store_address', USAGE=A25V,
    ACTUAL=A25V, MISSING=ON, TITLE='store address',
    REFERENCE=STORE_LOCATION_ID, PROPERTY=UDA,  $
  FIELDNAME=STORE_CITY, ALIAS='store_location_ID_1   _store_city',
    USAGE=A45, ACTUAL=A45, MISSING=ON, TITLE='store_city',
    REFERENCE=STORE_LOCATION_ID, PROPERTY=UDA,  $
  FIELDNAME=STORE_ID, ALIAS=store_ID, USAGE=I11, ACTUAL=I4, MISSING=ON,
    TITLE='store_ID',WITHIN=STORE_LOCATION_ID, $
```

Calling a Db2 Stored Procedure Using SQL Passthru

Db2 stored procedures are supported using SQL Passthru. These procedures need to be developed within Db2 using the CREATE PROCEDURE command.

The adapter supports stored procedures with IN, OUT, and INOUT parameters.

The output parameter values that are returned by stored procedures are available as result sets. These values form a single-row result set that is transferred to the client after all other result sets are returned by the invoked stored procedure. The names of the output parameters (if available) become the column titles of that result set.

Note that only the output parameters (and the returned value) referenced in the invocation string are returned to the client. As a result, users have full control over which output parameters have their values displayed.

The server supports invocation of stored procedures written according to the rules of the underlying DBMS. Note that the examples shown in this section are SQL-based. See the DBMS documentation for rules, languages, and additional programming examples.

Syntax: How to Invoke a Stored Procedure

```
SQL DB2 EX procname [parameter_specification1]
[,parameter_specification2]...
END
```

where:

`DB2`

Is the ENGINE suffix for Db2.

`procname`

Is the name of the stored procedure. It is the fully or partially qualified name of the stored procedure in the native RDBMS syntax.

You can employ either SQL or SYS naming conventions to control the separator character used for interpreting multipart names, as described in *Setting Naming Conventions* on page 41.

`parameter_specification`

IN, OUT, and INOUT parameters are supported. Use the variation required by the stored procedure:

`IN`

Is a literal (for example, 125, 3.14, 'abcde'). You can use reserved words as input. Unlike character literals, reserved words are not enclosed in quotation marks (for example, NULL). Input is required.

`OUT`

Is represented as a question mark (?). You can control whether output is passed to an application by including or omitting this parameter. If omitted, this entry will be an empty string (containing 0 characters).

`INOUT`

Consists of a question mark (?) for output and a literal for input, separated by a slash: /. (For example: ?/125, ?/3.14, ?/'abcde'.) The out value can be an empty string (containing 0 characters).

Example: Invoking a Stored Procedure

Note that this sample employs the SQL naming convention, where the "." character is used as the separator in multipart table names. If your site uses the SYS[TEM] naming convention (typical in IBM i environments), the "/" character is used as a separator in multipart table names. In this case, adjust the separator character as needed to conform to the SYS naming convention. For details, see *Setting Naming Conventions* on page 41.

In this example, a user invokes a stored procedure, edaqa.test_proc01, supplies input values for parameters 1, 3, 5 and 7, and requests the returned value of the stored procedure, as well as output values for parameters 2 and 3.

Note that parameters 4 and 6 are omitted; the stored procedure will use their default values, as specified at the time of its creation.

```
SQL DB2 EX edaqa.test_proc01 125,?,?/3.14,,'abc',,'xyz'
END
```

Example: Sample Stored Procedure

Note that this sample employs the SQL naming convention, where the "." character is used as the separator in multipart table names. If your site uses the SYS[TEM] naming convention (typical in IBM i environments), the "/" character is used as a separator in multipart table names. In this case, adjust the separator character as needed to conform to the SYS naming convention. For details, see *Setting Naming Conventions* on page 41.

This stored procedure uses out and inout parameters:

```
CREATE PROCEDURE EDAQA.PROCP3 (   OUT chSQLSTATE_OUT   CHAR(5),
                                  OUT intSQLCODE_OUT   INT,
                                  INOUT l_name char(20),
                                  INOUT f_name char(20))
    RESULT SETS 1
    LANGUAGE SQL
-----------------------------------------------------------------------
-- SQL Stored Procedure
-----------------------------------------------------------------------
P1: BEGIN
    -- Declare variable
    DECLARE SQLSTATE CHAR(5) DEFAULT '00000';
    DECLARE SQLCODE INT DEFAULT 0;
    -- Declare cursor
    DECLARE cursor1 CURSOR WITH RETURN FOR
        SELECT
           EDAQA.NF29005.SSN5 AS SSN5,
           EDAQA.NF29005.LAST_NAME5 AS LAST_NAME5,
           EDAQA.NF29005.FIRST_NAME5 AS FIRST_NAME5,
           EDAQA.NF29005.BIRTHDATE5 AS BIRTHDATE5,
           EDAQA.NF29005.SEX5 AS SEX5
        FROM
           EDAQA.NF29005
        WHERE
           (
             ( EDAQA.NF29005.LAST_NAME5 =  l_name )
        AND
             ( EDAQA.NF29005.FIRST_NAME5 = f_name )
           );
     -- Cursor left open for client application
    OPEN cursor1;
    SET chSQLSTATE_OUT = SQLSTATE;
    SET intSQLCODE_OUT = SQLCODE;
    SET l_name = 'this is first name';
    SET f_name = 'this is last name';
END P1   @
```

Chapter 4

Using the Adapter for DB Heritage Files

The Adapter for DB Heritage Files allows applications to access and read native IBM i data sources (also known as Physical, Logical, and Multi-format logical data base files) using the Open Query Facility (OPNQRYF). The adapter converts application requests into native OPNQRYF statements and returns optimized answer sets to the requesting application. All physical, logical, and multi-format formats are supported.

Note: In previous releases, this adapter was called DBFILE or DB File.

In this chapter:

Configuring the Adapter for DB Heritage Files

In the Web Query environment, the adapter is pre-configured. No additional configuration steps are necessary.

Managing DB Heritage Files Metadata

When the server accesses a data source, it needs to know how to interpret the data stored there. For each data source the server will access, you create a synonym that describes the structure of the data source and the server mapping of the DB Heritage Files data types.

Creating Synonyms

Synonyms define unique names (or aliases) for each DB Heritage Files file or view that is accessible from the server. Synonyms are useful because they hide the underlying data source location and identity from client applications. They also provide support for extended metadata features of the server, such as virtual fields and additional security mechanisms.

Using synonyms allows an object to be moved or renamed while allowing client applications to continue functioning without modification. The only modification required is a redefinition of the synonym on the server. The result of creating a synonym is a Master File that represents the server metadata.

Note: Refer to *DB Heritage Standard Master File Attributes* on page 53, if you wish to review general information about Master File attributes, create a synonym based on attributes other than a standard DDS data definition, or make manual adjustments to synonyms generated using graphical tools.

Procedure: How to Create a Synonym From the Web Query Environment

1. Expand the *Domains* folder, then expand a domain.
2. Expand the *Reports* folder, right-click a subfolder, and choose *Metadata* from the menu.
3. In the left-hand Adapter navigation pane, right-click DB Heritage Files and choose *Create Synonym* from the menu. The first of a series of synonym creation pages opens.

   Enter values for the parameters required for the adapter.

   For information about these parameters, see *Synonym Creation Parameters for DB Heritage Files* on page 50.
4. After entering parameter values, click *Create Synonym*.

   The synonym is stored in the baseapp directory. You can use any of the available reporting tools to create a report using the generated synonym.

Reference: Synonym Creation Parameters for DB Heritage Files

The following list describes the parameters for which you will need to supply values, and related tasks you will need to complete in order to create a synonym for the adapter. These options may appear on multiple panes. To advance from pane to pane, click the buttons provided, ending with the *Create Synonym* button, which generates the synonym based on your entries.

Select Files

Choose the type of file you wish to see from the following options: All, Physical, or Logical. (At this level, there is no differentiation between logical and multi-format logical.)

Library Name

Supply the name of the library in which the physical and/or logical files reside. (Wildcards are not permitted.)

Note: When you create a synonym for DB Heritage Files on the IBM i platform, standard IBM i naming conventions apply to the target data source. Therefore, the Adapter for DB Heritage Files supports the use of double-quotation marks around any library name and/or file name that contains lower case or NLS characters.

Filter by Name

If you wish to limit retrieval, enter a full file name or a partial name with a wildcard symbol % in the Filter by Name input box. A full name returns just that entry. A name with a wildcard symbol may return many entries.

Use text description as field name

Click this check box to use descriptive text from title fields as more meaningful field names in the synonym.

Application

The default value is baseapp.

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 the prefix and suffix fields blank.

Overwrite Existing Synonyms

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.

Select objects for synonym creation

To select all member names in the list, select the check box to the left of the *Default Synonym Name* column heading. (If the library contains a large number of files, this check box will not appear, however, synonyms will be created for all files automatically.)

To select specific member names, click the corresponding check boxes.

Not all listed objects are data-related. Be sure to select those that are appropriate for creating synonyms.

Note: Mixed-case names or names with NLS character will appear in double quotation marks. However, the double quotation marks will be replaced by underscore characters in the Default Synonym Name column.

Default Synonym Name

This column displays the name that will be assigned to each synonym. To assign a different name, replace the displayed value.

Note: If you see a synonym name surrounded by underscore characters (as described in the previous note), you can remove the underscores if you wish.

Example: Creating a Synonym for a DB Heritage Files Data Source

In this example, the field attributes from the original file initialization are used in the new metadata. (For related information about file initialization, see *DB Heritage Standard Master File Attributes* on page 53.

An internal field (RECFORM) is also automatically added to reference a particular record's format. The RECFORM ALIAS always has a value of the name of the record format and is always added (even for single format files).

To generate the following synonym from the Create Synonym panes:

1. Specify *SHIPPING* in the Library Name field, then click *Submit*.
2. On the second Create Synonym pane, choose the member name *DBF2901* from the list of objects that comprise the library and click *Create Synonym*.

   The synonym is created and added under the specified application directory (baseapp is the default).

   A status window displays the message: All Synonyms Created Successfully
3. From the message window, click *Applications* on the menu bar.
4. Open the baseapp application folder in the navigation pane and right-click the synonym *DBF2901*.
5. Select *Edit as Text* from the drop-down menu to view the generated Master File. Your selections are reflected in the DATASET field, prefixed with QSYS:

Generated Master File:

```
FILENAME=DBF2901, SUFFIX=DBFILE  ,
 DATASET=QSYS:SHIPPING/DBF2901, $
  SEGMENT=DBF2901, SEGTYPE=S0, $
    FIELDNAME=CCODE1, ALIAS=COUNTRY_COD1, USAGE=A5, ACTUAL=A5,
      TITLE='COUNTRY_COD1', DESCRIPTION='Country Code Number 1', $
    FIELDNAME=CNAME1, ALIAS=COUNTRY_NAM1, USAGE=P16, ACTUAL=P8,
      TITLE='COUNTRY_NAM1', DESCRIPTION='Country Name Number 1', $
    FIELDNAME=RECFORM, ALIAS=DBF2901, USAGE=A10, ACTUAL=A10, $
```

Reference: Managing Synonyms

Once you have created a synonym, you can right-click the synonym name in the navigation pane to access the available options.

DB Heritage Standard Master File Attributes

Traditional IBM i applications, such as RPG, use Data Description Specification (DDS) files to initialize data files (also known as native database files) with header information about fields, data types, sizes and, optionally, aliases (ALIAS), column headings (COLHDG), descriptive comments (TEXT), and other information such as keys.

The features of the DDS have remained essentially stable despite the advent on IBM i of newer languages, such as C and COBOL, and their associated features. For example, while COBOL file descriptions may contain OCCURS statements, DDS does not support OCCURS clauses; for DDS purposes, separate fields must be used to handle this construct.

For DB Heritage Files, the IBM i DSPFFD command is used to extract the DDS information that is initialized into a data file at synonym-creation time, generating a complete set of metadata. However, in some instances, applications can redefine the use of the DDS information drastically, unbeknownst to the DSPFFD tool since the header information remains unchanged. This might occur, for example, in a COBOL application that has a common set of keys, part of which is a rectype that is used to redefine the remainder of the record into any number of layouts. The presence of the common set of keys suggests that the application is writable, when, in fact, DB Heritage Files is a Read-only adapter. While this situation does not generally occur, if it does you can avoid problems by following these guidelines to define proper metadata:

- Keep in mind that the Adapter for DB Heritage Files is *Read only*. Although some of the descriptive information may give the impression that describing keys is required and would provide Write capability, that is, in fact, *not* the case.

- Although the Adapter for DB Heritage Files is not a read/write adapter, you can use it in conjunction with the Adapter for Db2, which is a read/write adapter. Db2 itself provides an automatic feature for reading and writing native database files, with the exception of multi-format logical files, dynamically by using DSPFFD information on the fly. Thus, Web Query can read and write DB Heritage Files data if the Adapter for Db2 is used to access the data. This two-adapter method enables you to take advantage of features of each one:

  - Adapter for Db2 read/write capabilities.
  - Adapter for DB Heritage Files support of Multi-Format Logical files and faster processing capabilities for large quantities of records.

  Tip: You can set up metadata for both methods and switch between them based upon the needs of your application.

- If a COBOL FD is available, you can use it with the Adapter for Flat Files to generate metadata with a SUFFIX=FIX value. You can then change the SUFFIX to DBFILE and add the DATASET= *value*.

For related information about DSPFFD, see *DSPFFD Behavior* on page 56.

Reference: Standard Master File Attributes for DB Heritage Files

Most standard Master File attributes are used with DB Heritage Files data sources in the standard way.

- **FILENAME.** The FILENAME attribute is the logical name of the file. It typically matches the Master File name.
- **SUFFIX.** The SUFFIX attribute in the declaration has the value DBFILE to indicate that access is for DB Heritage Files.
- **DATASET.** The DATASET attribute in the declaration indicates the default file to access. This attribute is optional and may be overridden by a FILEDEF command issued before a TABLE request (for example, in a focexec). The attribute value is preceded by the text *QSYS:*, which indicates that the file is a library (rather than a two-part application name). The syntax is:

  ```
  QSYS:[library/]file[(member)]
  ```

  If the library is not supplied, the user's library path is used to locate the file.

- **SEGNAME.** The SEGNAME attribute of the first or root segment in a Master File for a DB Heritage Files data source must be ROOT. The remaining segments can have any valid segment name.

The only exception involves unrelated RECTYPEs, where the root SEGNAME must be DUMMY.

All non-repeating data goes in the root segment. The remaining segments may have any valid name from one to eight characters.

Any segment except the root is the descendant, or child, of another segment. The PARENT attribute supplies the name of the segment that is the hierarchical parent or owner of the current segment. If no PARENT attribute appears, the default is the immediately preceding segment. The PARENT name may be one to eight characters.

- **SEGTYPE.** The SEGTYPE attribute should be S0 for DB Heritage Files data sources.
- **GROUP.** The keys of a DB Heritage Files data source are defined in the segment declarations as GROUPs consisting of one or more fields.
- **FIELD.** The field names in the DB Heritage Files, along with the descriptive attributes, ALIAS, USAGE, and ACTUAL. For related information, see *USAGE and ACTUAL Values in DB Heritage Files* on page 55.

Reference: USAGE and ACTUAL Values in DB Heritage Files

Synonyms for DB Heritage Files will pick up certain additional data attributes which may have been declared in the DDS information that was used when the original data file was created.

For example, in a monetary declaration such as a dollar sign ($), if the IBM i system value for QCURSYN is the same as the symbol used in the DDS declaration, then the Master File *M* attribute (floating monetary symbol) is used. In this situation, the actual symbol that is displayed depends on the value defined by the SET CURRSYN command (dollar sign, by default), which may be set in an application or in a profile.

However, if the values for QCURSYN and the monetary sign in the DDS information do not match, but the DDS monetary value is defined as dollar, pound, euro, or yen, then !D, !L, !E, or !Y are displayed regardless of the value of CURRSYN on the assumption that the application is intentionally using multiple monetary symbols.

On the other hand, if QCURSYN and the monetary sign in the DDS information do not match and the DDS monetary value is not set to dollar, pound, euro, or yen, then the M symbol is displayed since dollar, pound, euro, and yen are the only explicitly supported characters for reports with mixed monetary symbols.

Among the other data attributes that may be picked up from the DDS information are date and time fields and the following symbols: *c* (suppresses commas), *C* (inserts a comma after every third significant digit, or a period instead of a comma if continental decimal notation is in use), R (places CR after negative numbers).

Reference: DSPFFD Behavior

The DSPFFD field name value is used as the ALIAS=*value,* and the DSPFFD alias value is used as the FIELDNAME=*value*. This ensures that DSPFFD aliases, which can be too long to use as ALIAS values when creating synonyms in a hub/sub server configuration, are within the required character limit for hub/sub aliases.

While you may wish to refresh existing DB Heritage synonyms for other reasons, there is no explicit need to do so unless the underlying DSPFFD information actually changes. Existing DB Heritage related procedures will continue to run, with one possible difference. Report column titles, and possibly associated column widths, will display differently, if the request:

- Uses recreated metadata.
- Includes different values for FIELDNAME and ALIAS. (Frequently, these values are the same in the Master File.)
- Does not specify a TITLE value in the metadata.
- Does not assign explicit column titles.

Describing a Group Field

A single-segment data source may have only one key field, but it must still be described with a GROUP declaration. The group must have ALIAS=KEY.

Groups can also be assigned simply to provide convenient reference names for groups of fields. Suppose that you have a series of three fields for an employee: last name; first name; and middle initial. You use these three fields consistently to identify the employee. You can identify the three fields in your Master File as a GROUP named EMPINFO. Then, you can refer to these three linked fields as a single unit, called EMPINFO. When using the GROUP feature for non-keys, the parameter ALIAS= must still be used, but should not equal KEY.

For group fields, you must supply both the USAGE and ACTUAL formats in alphanumeric format. The length must be exactly the sum of the subordinate field lengths.

The GROUP declaration USAGE attribute specifies how many positions to use to describe the key in the data source. If a Master File does not completely describe the full key at least once, the following warning message appears:

```
(FOC1016) INVALID KEY DESCRIPTION IN MASTER FILE
```

The cluster key definition is compared to the Master File for length and displacement.

When you expand on the key in a RECTYPE data source, describe the key length in full on the last non-OCCURS segment on each data path.

Do not describe a group with ALIAS=KEY for OCCURS segments.

If the fields that make up a group key are not alphanumeric fields, the format of the group key is still alphanumeric, but its length is determined differently. The ACTUAL length is still the sum of the subordinate field lengths. However, the USAGE format is the sum of the internal storage lengths of the subordinate fields. You determine these internal storage lengths as follows:

- Fields of type I have a value of 4.
- Fields of type F have a value of 4.
- Fields of type P that are 8 bytes can have a USAGE of P15 or P16 (sign and decimal for a total of 15 digits). Fields that are 16 bytes have a USAGE of P17 or larger.
- Fields of type D have a value of 8.
- Alphanumeric fields have a value equal to the number of characters they contain as their field length.

Note:

- Since all group fields must be defined in alphanumeric format, those that include numeric component fields should not be used as verb objects in a report request.
- The MISSING attribute is not supported on the group field, but is supported on the individual fields comprising the group.

Syntax: How to Describe a DB Heritage Files Group Field

```
GROUP = keyname, ALIAS = KEY, USAGE = Ann, ACTUAL = Ann ,$
```

where:

keyname

Can have up to 66 characters.

Example: Describing a DB Heritage Files Group Field

In the library data source, the first field, PUBNO, can be described as a group key. The publisher's number consists of three elements: a number that identifies the publisher, one that identifies the author, and one that identifies the title. They can be described as a group key, consisting of a separate field for each element if the data source is a DB Heritage Files data structure. The Master File looks as follows:

```
FILE = LIBRARY5, SUFFIX = DBFILE,$
 SEGMENT = ROOT, SEGTYPE = S0,$
  GROUP = BOOKKEY      ,ALIAS = KEY ,USAGE = A10    ,ACTUAL = A10  ,$
   FIELDNAME = PUBNO   ,ALIAS = PN  ,USAGE = A3     ,ACTUAL = A3   ,$
   FIELDNAME = AUTHNO  ,ALIAS = AN  ,USAGE = A3     ,ACTUAL = A3   ,$
   FIELDNAME = TITLNO  ,ALIAS = TN  ,USAGE = A4     ,ACTUAL = A4   ,$
   FIELDNAME = AUTHOR   ,ALIAS = AT  ,USAGE = A25   ,ACTUAL = A25  ,$
   FIELDNAME = TITLE    ,ALIAS = TL  ,USAGE = A50   ,ACTUAL = A50  ,$
   FIELDNAME = BINDING  ,ALIAS = BI  ,USAGE = A1    ,ACTUAL = A1   ,$
   FIELDNAME = PRICE    ,ALIAS = PR  ,USAGE = D8.2N ,ACTUAL = D8   ,$
   FIELDNAME = SERIAL   ,ALIAS = SN  ,USAGE = A15   ,ACTUAL = A15  ,$
   FIELDNAME = SYNOPSIS ,ALIAS = SY  ,USAGE = A150  ,ACTUAL = A150 ,$
   FIELDNAME = RECTYPE  ,ALIAS = B   ,USAGE = A1    ,ACTUAL = A1   ,$
```

Example: Describing a DB Heritage Files Group Field With Multiple Formats

```
GROUP = A, ALIAS = KEY, USAGE = A14, ACTUAL = A8   ,$
 FIELDNAME = F1, ALIAS = F1, USAGE = P6, ACTUAL=P2 ,$
 FIELDNAME = F2, ALIAS = F2, USAGE = I9, ACTUAL=I4 ,$
 FIELDNAME = F3, ALIAS = F3, USAGE = A2, ACTUAL=A2 ,$
```

The lengths of the ACTUAL attributes for subordinate fields F1, F2, and F3 total 8, which is the length of the ACTUAL attribute of the group key. The display lengths of the USAGE attributes for the subordinate fields total 17. However, the length of the group key USAGE attribute is found by adding their internal storage lengths as specified by their field types: 8 for USAGE=P6, 4 for USAGE=I9, and 2 for USAGE=A2, for a total of 14.

Example: Accessing a Group Field With Multiple Formats

When you use a group field with multiple formats in a query, you must account for each position in the group, including trailing blanks or leading zeros. The following example illustrates how to access a group field with multiple formats in a query.

```
GROUP = GRPB, ALIAS = KEY, USAGE = A8, ACTUAL = A8 ,$
 FIELDNAME = FIELD1, ALIAS = F1, USAGE = A2, ACTUAL = A2 ,$
 FIELDNAME = FIELD2, ALIAS = F2, USAGE = I8, ACTUAL = I4 ,$
 FIELDNAME = FIELD3, ALIAS = F3, USAGE = A2, ACTUAL = A2 ,$
```

The values in fields F1 and F3 may include some trailing blanks, and the values in field F2 may include some leading zeros. When using the group in a query, you must account for each position. Because FIELD2 is a numeric field, you cannot specify the IF criteria as follows:

```
IF GRPB EQ 'A 0334BB'
```

You can eliminate this error by using a slash (/) to separate the components of the group key.

```
IF GRPB EQ 'A/334/BB'
```

Note: Blanks and leading zeros are assumed where needed to fill out the key.

Describe these multiply occurring fields by placing them in a separate segment. Fields A and B are placed in the root segment. Fields C1 and C2, which occur multiply in relation to A and B, are placed in a descendant segment. You use an additional segment attribute, the OCCURS attribute, to specify that these segments represent multiply occurring fields. In certain cases, you may also need a second attribute, called the POSITION attribute.

Using the OCCURS Attribute

The OCCURS attribute is an optional segment attribute used to describe records containing repeating fields or groups of fields. Define such records by describing the singly occurring fields in one segment, and the multiply occurring fields in a descendant segment. The OCCURS attribute appears in the declaration for the descendant segment.

You can have several sets of repeating fields in your data structure. Describe each of these sets of fields as a separate segment in your data source description. Sets of repeating fields can be divided into two basic types: parallel and nested.

Syntax: How to Specify a Repeating Field

```
OCCURS = occurstype
```

Possible values for *occurstype* are:

n

Is an integer value showing the number of occurrences (from 1 to 4095).

fieldname

Names a field in the parent segment whose integer value contains the number of occurrences of the descendant segment.

VARIABLE

Indicates that the number of occurrences varies from record to record. The number of occurrences is computed from the record length (for example, if the field lengths for the segment add up to 40, and 120 characters are read in, it means there are three occurrences).

Place the OCCURS attribute in your segment declaration after the PARENT attribute.

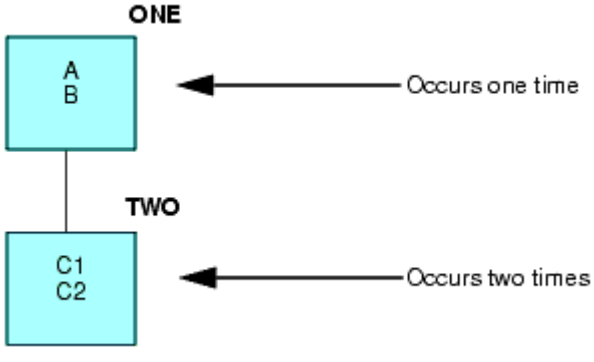
When different types of records are combined in one data source, each record type can contain only one segment defined as OCCURS=VARIABLE. It may have OCCURS descendants (if it contains a nested group), but it may not be followed by any other segment with the same parent—that is, there can be no other segments to its right in the hierarchical data structure. This restriction is necessary to ensure that data in the record is interpreted unambiguously.

Example: **Using the OCCURS Attribute**

Consider the following simple data structure:

A	B	C1	C2	C1	C2

You have two occurrences of fields C1 and C2 for every one occurrence of fields A and B. Thus, to describe this data source, you place fields A and B in the root segment, and fields C1 and C2 in a descendant segment, as shown here.



Describe this data source as follows:

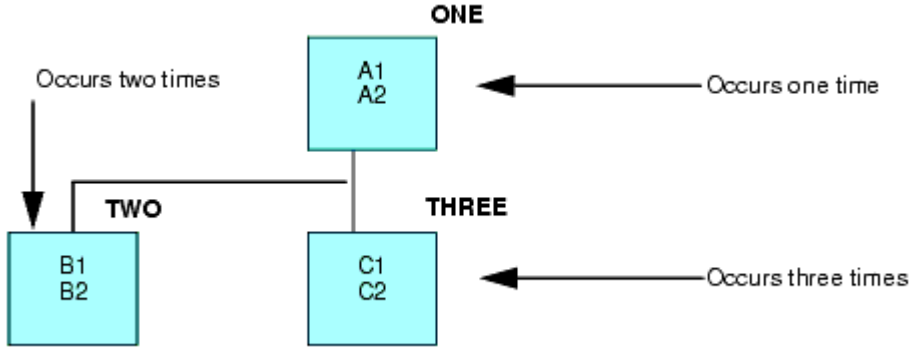
```
FILENAME = EXAMPLE1, SUFFIX = FIX, $
 SEGNAME = ONE, SEGTYPE=S0, $
  FIELDNAME = A,  ALIAS=, USAGE = A2, ACTUAL = A2, $
  FIELDNAME = B,  ALIAS=, USAGE = A1, ACTUAL = A1, $
 SEGNAME = TWO, PARENT = ONE, OCCURS = 2, SEGTYPE=S0, $
  FIELDNAME = C1, ALIAS=, USAGE = I4, ACTUAL = I2, $
  FIELDNAME = C2, ALIAS=, USAGE = I4, ACTUAL = I2, $
```

Describing a Parallel Set of Repeating Fields

Parallel sets of repeating fields are those that have nothing to do with one another (that is, they have no parent-child or logical relationship). Consider the following data structure:

A1	A2	B1	B2	B1	B2	C1	C2	C1	C2	C1	C2

In this example, fields B1 and B2 and fields C1 and C2 repeat within the record. The number of times that fields B1 and B2 occur has nothing to do with the number of times fields C1 and C2 occur. Fields B1 and B2 and fields C1 and C2 are parallel sets of repeating fields. They should be described in the data source description as children of the same parent, the segment that contains fields A1 and A2. The following data structure reflects their relationship.

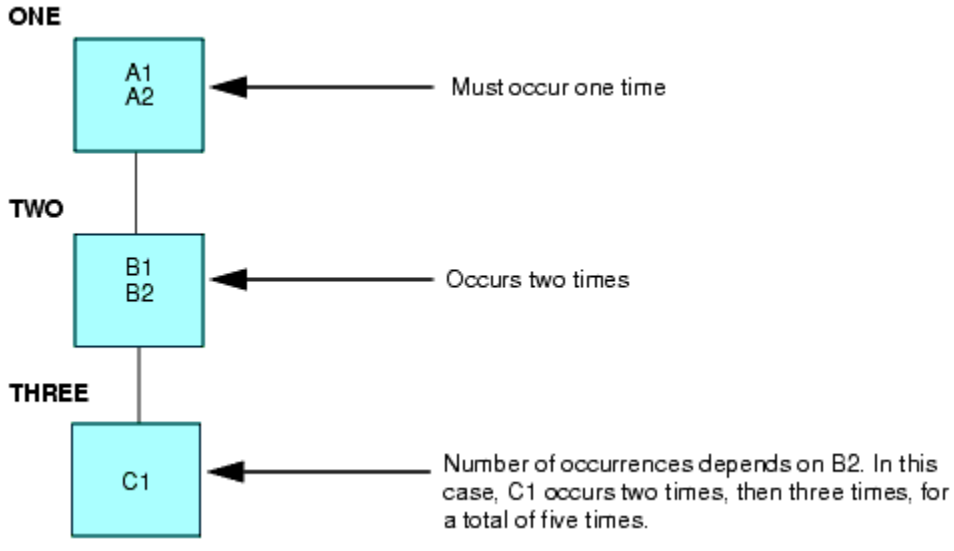


Describing a Nested Set of Repeating Fields

Nested sets of repeating fields are those whose occurrence depends on one another in some way. Consider the following data structure:

A1	A2	B1	B2	C1	C1	B1	B2	C1	C1	C1

In this example, field C1 only occurs after fields B1 and B2 occur once. It occurs varying numbers of times, recorded by a counter field, B2. There is not a set of occurrences of C1 which is not preceded by an occurrence of fields B1 and B2. Fields B1, B2, and C1 are a nested set of repeating fields. They can be represented by the following data structure.



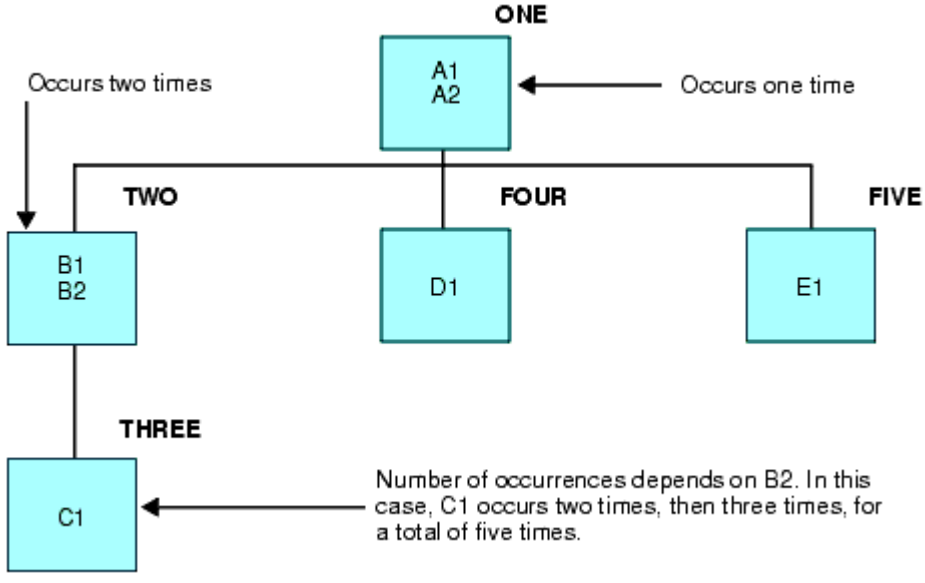
Since field C1 repeats with relation to fields B1 and B2, which repeat in relation to fields A1 and A2, field C1 is described as a separate, descendant segment of Segment TWO, which is in turn a descendant of Segment ONE.

Example: Describing Parallel and Nested Repeating Fields

The following data structure contains both nested and parallel sets of repeating fields.

A 1	A 2	B 1	B 2	C 1	C 1	C 1	B 1	B 2	C 1	C 1	C 1	C 1	D 1	D 1	E 1	E 1	E 1	E 1

It produces the following data structure.



Describe this data source as follows. Notice that the assignment of the PARENT attributes shows you how the occurrences are nested.

```
FILENAME = EXAMPLE3, SUFFIX = FIX,$
 SEGNAME = ONE,    SEGTYPE=S0,$
  FIELDNAME = A1 ,ALIAS= ,ACTUAL = A1  ,USAGE = A1  ,$
  FIELDNAME = A2 ,ALIAS= ,ACTUAL = I1  ,USAGE = I1  ,$
 SEGNAME = TWO,    SEGTYPE=S0, PARENT = ONE, OCCURS = 2  ,$
  FIELDNAME = B1 ,ALIAS= ,ACTUAL = A15 ,USAGE = A15 ,$
  FIELDNAME = B2 ,ALIAS= ,ACTUAL = I1  ,USAGE = I1  ,$
 SEGNAME = THREE, SEGTYPE=S0, PARENT = TWO, OCCURS = B2 ,$
  FIELDNAME = C1 ,ALIAS= ,ACTUAL = A25 ,USAGE = A25 ,$
 SEGNAME = FOUR,  SEGTYPE=S0, PARENT = ONE, OCCURS = A2 ,$
  FIELDNAME = D1 ,ALIAS= ,ACTUAL = A15 ,USAGE = A15 ,$
 SEGNAME = FIVE,  SEGTYPE=S0, PARENT = ONE, OCCURS = VARIABLE,$
  FIELDNAME = E1 ,ALIAS= ,ACTUAL = A5  ,USAGE = A5  ,$
```

Note:

- Segments ONE, TWO, and THREE represent a nested group of repeating segments. Fields B1 and B2 occur a fixed number of times, so OCCURS equals 2. Field C1 occurs a certain number of times within each occurrence of fields B1 and B2. The number of times C1 occurs is determined by the value of field B2, which is a counter. In this case, its value is 3 for the first occurrence of Segment TWO and 4 for the second occurrence.

- Segments FOUR and FIVE consist of fields that repeat independently within the parent segment. They have no relationship to each other or to Segment TWO except their common parent, so they represent a parallel group of repeating segments.
- As in the case of Segment THREE, the number of times Segment FOUR occurs is determined by a counter in its parent, A2. In this case, the value of A2 is two.
- The number of times Segment FIVE occurs is variable. This means that all the rest of the fields in the record (all those to the right of the first occurrence of E1) are read as recurrences of field E1. To ensure that data in the record is interpreted unambiguously, a segment defined as OCCURS=VARIABLE must be at the end of the record. In a data structure diagram, it is the rightmost segment. Note that there can be only one segment defined as OCCURS=VARIABLE for each record type.

Using the POSITION Attribute

The POSITION attribute is an optional attribute used to describe a structure in which multiply occurring fields with an established number of occurrences are located in the middle of the record. You describe the data source as a hierarchical structure, made up of a parent segment and at least one child segment that contains the multiply occurring fields. The parent segment is made up of whatever singly occurring fields are in the record, as well as one or more alphanumeric fields that appear where the multiply occurring fields appear in the record. The alphanumeric field may be a dummy field that is the exact length of the combined multiply occurring fields. For example, if you have four occurrences of an eight-character field, the length of the field in the parent segment is 32 characters.

You can also use the POSITION attribute to re-describe fields with SEGTYPE=U. See *Redefining a Field in a DB Heritage Files Data Source* on page 66.

Syntax: How to Specify the Position of a Repeating Field

The POSITION attribute is described in the child segment. It gives the name of the field in the parent segment that specifies the starting position and overall length of the multiply occurring fields. The syntax of the POSITION attribute is

```
POSITION = fieldname
```

where:

fieldname

Is the name of the field in the parent segment that defines the starting position of the multiple field occurrences.

Example: **Specifying the Position of a Repeating Field**

Consider the following data structure:

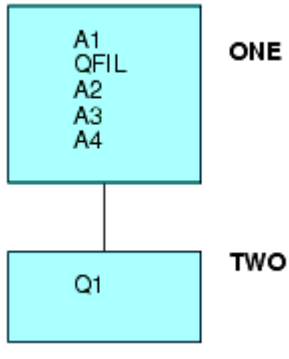
A1	Q1	Q1	Q1	Q1	A2	A3	A4

In this example, field Q1 repeats four times in the middle of the record. When you describe this structure, you specify a field or fields that occupy the position of the four Q1 fields in the record. You then assign the actual Q1 fields to a multiply occurring descendant segment. The POSITION attribute, specified in the descendant segment, gives the name of the field in the parent segment that identifies the starting position and overall length of the Q fields.

Use the following Master File to describe this structure.

```
FILENAME = EXAMPLE3, SUFFIX = FIX,$
 SEGNAME = ONE, SEGTYPE=S0,$
  FIELDNAME = A1    ,ALIAS= ,USAGE = A14 ,ACTUAL = A14 ,$
  FIELDNAME = QFIL  ,ALIAS= ,USAGE = A32 ,ACTUAL = A32 ,$
  FIELDNAME = A2    ,ALIAS= ,USAGE = I2  ,ACTUAL = I2  ,$
  FIELDNAME = A3    ,ALIAS= ,USAGE = A10 ,ACTUAL = A10 ,$
  FIELDNAME = A4    ,ALIAS= ,USAGE = A15 ,ACTUAL = A15 ,$
 SEGNAME = TWO, SEGTYPE=S0, PARENT = ONE, POSITION = QFIL, OCCURS = 4 ,$
  FIELDNAME = Q1    ,ALIAS= ,USAGE = D8  ,ACTUAL = D8  ,$
```

This produces the following structure:



If the total length of the multiply occurring fields is longer than 4095, you can use a filler field after the dummy field to make up the remaining length. This is required, because the format of an alphanumeric field cannot exceed 4095 bytes.

Notice that this structure works only if you have a fixed number of occurrences of the repeating field. This means the OCCURS attribute of the descendant segment must be of the type OCCURS=*n*. OCCURS=*fieldname* or OCCURS=VARIABLE does not work.

Specifying the ORDER Field

In an OCCURS segment, the order of the data may be significant. For example, the values may represent monthly or quarterly data, but the record itself may not explicitly specify the month or quarter to which the data applies.

To associate a sequence number with each occurrence of the field, you may define an internal counter field in any OCCURS segment. A value is automatically supplied that defines the sequence number of each repeating group.

Syntax: How to Specify the Sequence of a Repeating Field

The syntax rules for an ORDER field are:

- It must be the last field described in an OCCURS segment.
- The field name is arbitrary.
- The ALIAS is ORDER.
- The USAGE is I*n*, with any appropriate edit options.
- The ACTUAL is I4.

For example:

```
FIELD = ACT_MONTH, ALIAS = ORDER, USAGE = I2MT, ACTUAL = I4, $
```

Order values are 1, 2, 3, and so on, within each occurrence of the segment. The value is assigned prior to any selection tests that might accept or reject the record, and so it can be used in a selection test.

For example, to obtain data for only the month of June, type:

```
SUM AMOUNT...
WHERE ACT_MONTH IS 6
```

The ORDER field is a virtual field used internally. It does not alter the logical record length (LRECL) of the data source being accessed.

Redefining a Field in a DB Heritage Files Data Source

Redefining record fields in non-FOCUS data sources is supported. This enables you to describe a field with an alternate layout.

Within the Master File, the redefined fields must be described in a separate unique segment (SEGTYPE=U) using the POSITION=*fieldname* and OCCURS=1 attributes.

The redefined fields can have any user-defined name.

Syntax: How to Redefine a Field

```
SEGNAME = segname, SEGTYPE = U, PARENT = parentseg,
OCCURS = 1, POSITION = fieldname,$
```

where:

segname

Is the name of the segment.

parentseg

Is the name of the parent segment.

fieldname

Is the name of the first field being redefined. Using the unique segment with redefined fields helps avoid problems with multipath reporting.

A one-to-one relationship forms between the parent record and the redefined segment.

Example: Redefining a DB Heritage Files Structure

The following example illustrates redefinition of the DB Heritage Files structure described in the COBOL file description, where the COBOL FD is:

```
01 ALLFIELDS
  02 FLD1   PIC X(4)         -  this describes alpha/numeric data
  02 FLD2   PIC X(4)         -  this describes numeric data
  02 RFLD1  PIC 9(5)V99 COMP-3 REDEFINES FLD2
  02 FLD3   PIC X(8)         -  this describes alpha/numeric data

FILE = REDEF, SUFFIX = DBFILE,$
 SEGNAME = ONE, SEGTYPE = S0,$
  GROUP = RKEY, ALIAS = KEY, USAGE = A4 ,ACTUAL = A4 ,$
   FIELDNAME = FLD1,,  USAGE = A4   ,ACTUAL = A4 ,$
   FIELDNAME = FLD2,,  USAGE = A4   ,ACTUAL = A4 ,$
   FIELDNAME = FLD3,,  USAGE = A8   ,ACTUAL = A8 ,$
 SEGNAME = TWO, SEGTYPE = U, POSITION = FLD2, OCCURS = 1, PARENT = ONE ,$
   FIELDNAME = RFLD1,, USAGE = P8.2 ,ACTUAL = Z4 ,$
```

Reference: Special Considerations for Redefining a Field

- Redefinition is a read-only feature and is used only for presenting an alternate view of the data. It is not used for changing the format of the data.

- For non-alphanumeric fields, you must know your data. Attempts to print numeric fields that contain alphanumeric data produce data exceptions or errors converting values. It is recommended that the first definition always be alphanumeric to avoid conversion errors.
- More than one field can be redefined in a segment.
- Redefines are supported only for IDMS, IMS, VSAM, DB Heritage Files, Db2, and FIX data sources.

Extra-Large Record Length Support With DB Heritage Files

If a Master File describes a data source with records longer than 12K, which is typical for OCCURS segments and varchar fields, you must specify a larger record size in advance.

Syntax: How to Define the Maximum Record Length

```
SET MAXLRECL = nnnnn
```

where:

nnnnn

Is up to 32768.

For example, SET MAXLRECL=12000 allows handling of records that are 12000 bytes long. Once you have entered the SET MAXLRECL command, you can obtain the current value of the MAXLRECL parameter by using the ? SET MAXLRECL command.

If the actual record length is longer than specified, retrieval halts and the actual record length appears in hexadecimal notation.

Describing Multiple Record Types in DB Heritage Files

DB Heritage Files data sources can contain more than one type of record. When they do, they can be structured in one of two ways:

- A positional relationship may exist between the various record types, with a record of one type being followed by one or more records containing detailed information about the first record.

  If a positional relationship exists between the various record types, with a parent record of one type followed by one or more child records containing detail information about the parent, you describe the structure by defining the parent as the root, and the detail segments as descendants.

Some DB Heritage Files data sources are structured so that descendant records relate to each other through concatenating key fields. That is, the key fields of a parent record serve as the first part of the key of a child record. In such cases, the segment's key fields must be described using a GROUP declaration. Each segment's GROUP key fields consist of the renamed key fields from the parent segment plus the unique key field from the child record.

- The records have no meaningful positional relationship, and records of varying types exist independently of each other in the data source.

  If the records have no meaningful positional relationship, you have to provide some means for interpreting the type of record that has been read. Do this by creating a dummy root segment for the records.

Key-sequenced DB Heritage Files data sources also use the RECTYPE attribute to distinguish various record types within the data source.

A parent does not always share its RECTYPE with its descendants. It may share some other identifying piece of information, such as the PUBNO in the example. This is the field that should be included in the parent key, as well as all of its descendant keys, to relate them.

When using the RECTYPE attribute in DB Heritage Files data sources with group keys, the RECTYPE field can be part of the segment group key only when it belongs to a segment that has no descendants, or to a segment whose descendants are described with an OCCURS attribute. In *Describing Positionally Related Records* on page 71, the RECTYPE field is added to the group key in the SERIANO segment, the lowest descendant segment in the chain.

Describing a RECTYPE Field

When a data source contains multiple record types, there must be a field in the records themselves that can be used to differentiate between the record types. You can find information on this field in your existing description of the data source (for example, a COBOL FD statement). This field must appear in the same physical location of each record. For example, columns 79 and 80 can contain a different two-digit code for each unique record type. Describe this identifying field with the field name RECTYPE.

Another technique for redefining the parts of records is to use the MAPFIELD and MAPVALUE attributes described in *Describing a Repeating Group Using MAPFIELD* on page 85.

Syntax: How to Specify a Record Type Field

The RECTYPE field must fall in the same physical location of each record in the data source, or the record is ignored. The syntax to describe the RECTYPE field is

```
FIELDNAME = RECTYPE, ALIAS = value, USAGE = format, ACTUAL = format,
ACCEPT = {list|range} ,$
```

where:

value

Is the record type in alphanumeric format, if an ACCEPT list is not specified. If there is an ACCEPT list, this can be any value.

format

Is the data type of the field. In addition to RECTYPE fields in alphanumeric format, RECTYPE fields in packed and integer formats (formats P and I) are supported. Possible values are:

A*n* (where *n* is 1-4095) indicates character data, including letters, digits, and other characters.

I*n* indicates ACTUAL (internal) format binary integers:

- I1 = single-byte binary integer.
- I2 = half-word binary integer (2 bytes).
- I4 = full-word binary integer (4 bytes).

The USAGE format can be I1 through I9, depending on the magnitude of the ACTUAL format.

P*n* (where *n* is 1-16) indicates packed decimal ACTUAL (internal) format. n is the number of bytes, each of which contains two digits, except for the last byte which contains a digit and the sign. For example, P6 means 11 digits plus a sign.

If the field contains an assumed decimal point, represent the field with a USAGE format of P*m.n*, where *m* is the total number of digits, and *n* is the number of decimal places. Thus, P11.1 means an eleven-digit number with one decimal place.

list

Is a list of one or more lines of specific RECTYPE values for records that have the same segment layout. The maximum number of characters allowed in the list is 255. Separate each item in the list with either a blank or the keyword OR. If the list contains embedded blanks or commas, it must be enclosed within single quotation marks. The list may contain a single RECTYPE value. For example:

```
FIELDNAME = RECTYPE, ALIAS = TYPEABC, USAGE = A1,
ACTUAL = A1, ACCEPT = A OR B OR C, $
```

range

Is a range of one or more lines of RECTYPE values for records that have the same segment layout. The maximum number of characters allowed in the range is 255. If the range contains embedded blanks or commas, it must be enclosed within single quotation marks.

To specify a range of values, include the lowest value, the keyword TO, and the highest value, in that order. For example:

```
FIELDNAME = RECTYPE, ALIAS = ACCTREC, USAGE = P3,
ACTUAL = P2, ACCEPT = 100 TO 200, $
```

Example: Specifying the RECTYPE Field

The following field description is of a one-byte packed RECTYPE field containing the value 1:

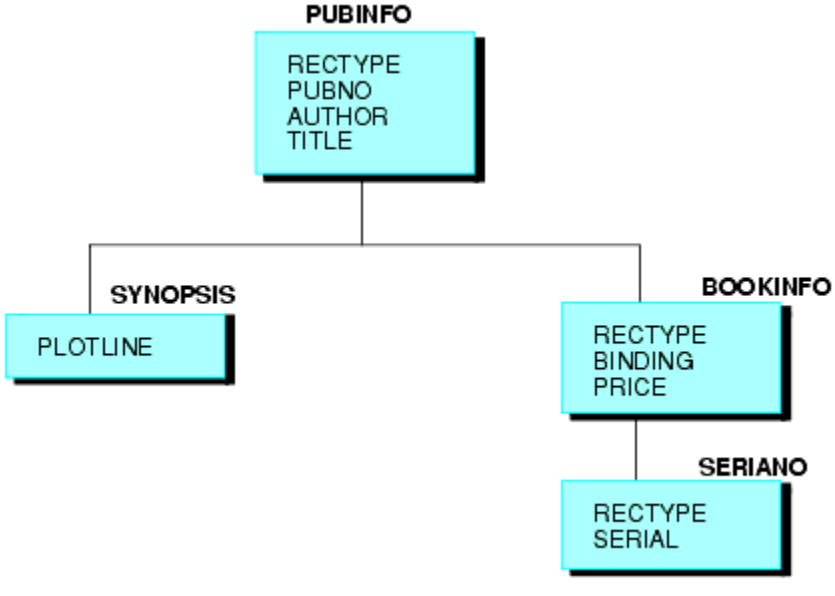
```
FIELD = RECTYPE, ALIAS = 1, USAGE = P1, ACTUAL = P1, $
```

The following field description is of a three-byte alphanumeric RECTYPE field containing the value A34:

```
FIELD = RECTYPE, ALIAS = A34, USAGE = A3, ACTUAL = A3,$
```

Describing Positionally Related Records

The following diagram shows a more complex version of the library data source.



Information that is common to all copies of a given book (the identifying number, the author name, and its title) has the same record type. All of this information is assigned to the root segment in the Master File. The synopsis is common to all copies of a given book, but in this data source it is described as a series of repeating fields of ten characters each, in order to save space.

The synopsis is assigned to its own subordinate segment with an attribute of OCCURS=VARIABLE in the Master File. Although there are segments in the diagram to the right of the OCCURS=VARIABLE segment, OCCURS=VARIABLE is the rightmost segment within its own record type. Only segments with a RECTYPE that is different from the OCCURS=VARIABLE segment can appear to its right in the structure. Note also that the OCCURS=VARIABLE segment does not have a RECTYPE. This is because it is part of the same record as its parent segment.

Binding and price can vary among copies of a given title. For instance, the library may have two different versions of *Pamela*, one a paperback costing $7.95, the other a hardcover costing $15.50. These two fields are of a second record type, and are assigned to a descendant segment in the Master File.

Finally, every copy of the book in the library has its own identifying serial number, which is described in a field of record type S. In the Master File, this information is assigned to a segment that is a child of the segment containing the binding and price information.

Use the following Master File to describe this data source:

```
FILENAME = LIBRARY2, SUFFIX = FIX,$
 SEGNAME = PUBINFO, SEGTYPE = S0,$
  FIELDNAME = RECTYPE  ,ALIAS = P     ,USAGE = A1    ,ACTUAL = A1  ,$
  FIELDNAME = PUBNO    ,ALIAS = PN    ,USAGE = A10   ,ACTUAL = A10 ,$
  FIELDNAME = AUTHOR   ,ALIAS = AT    ,USAGE = A25   ,ACTUAL = A25 ,$
  FIELDNAME = TITLE    ,ALIAS = TL    ,USAGE = A50   ,ACTUAL = A50 ,$
 SEGNAME = SYNOPSIS, PARENT = PUBINFO,  OCCURS = VARIABLE, SEGTYPE = S0,$
  FIELDNAME = PLOTLINE ,ALIAS = PLOTL ,USAGE = A10   ,ACTUAL = A10 ,$
 SEGNAME = BOOKINFO, PARENT = PUBINFO,  SEGTYPE = S0,$
  FIELDNAME = RECTYPE  ,ALIAS = B     ,USAGE = A1    ,ACTUAL = A1  ,$
  FIELDNAME = BINDING  ,ALIAS = BI    ,USAGE = A1    ,ACTUAL = A1  ,$
  FIELDNAME = PRICE    ,ALIAS = PR    ,USAGE = D8.2N ,ACTUAL = D8  ,$
 SEGNAME = SERIANO,  PARENT = BOOKINFO, SEGTYPE = S0,$
  FIELDNAME = RECTYPE  ,ALIAS = S     ,USAGE = A1    ,ACTUAL = A1  ,$
  FIELDNAME = SERIAL   ,ALIAS = SN    ,USAGE = A15   ,ACTUAL = A15 ,$
```

Note that each segment, except OCCURS, contains a field named RECTYPE and that the ALIAS for the field contains a unique value for each segment (P, B, and S). If there is a record in this data source with a RECTYPE other than P, B, or S, the record is ignored. The RECTYPE field must fall in the same physical location in each record.

Ordering of Records in the Data Source

Physical order determines parent/child relationships in sequential records. Every parent record does not need descendants. Specify how you want data in missing segment instances handled in your reports by using the SET command to change the ALL parameter.

In the example *Describing Positionally Related Records* on page 71, if the first record in the data source is not a PUBINFO record, the record is considered to be a child without a parent. Any information allotted to the SYNOPSIS segment appears in the PUBINFO record. The next record may be a BOOKINFO or even another PUBINFO (in which case the first PUBINFO is assumed to have no descendants). Any SERIANO records are assumed to be descendants of the previous BOOKINFO record. If a SERIANO record follows a PUBINFO record with no intervening BOOKINFO, it is treated as if it has no parent.

Example: Describing DB Heritage Files Positionally Related Records

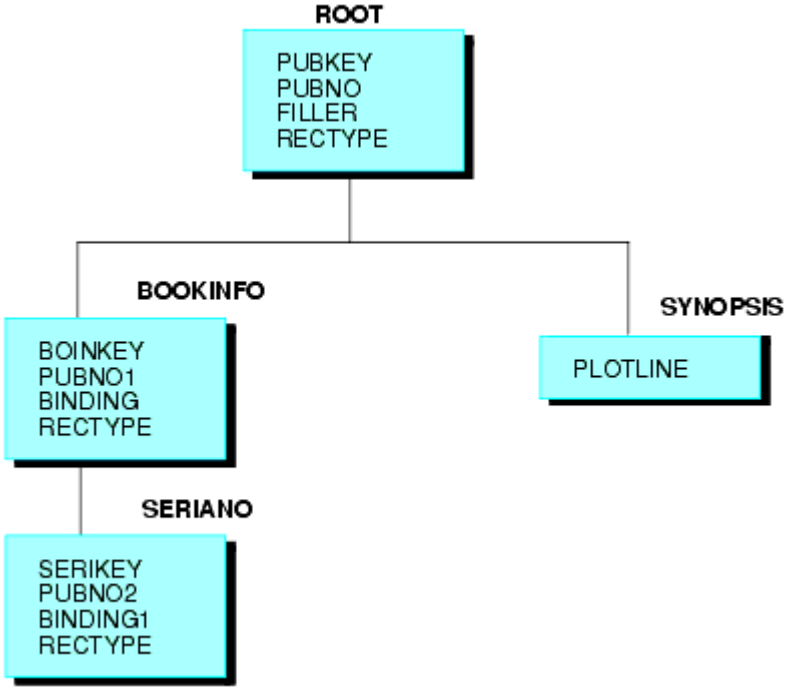
Consider the following DB Heritage Files data source that contains three types of records. The ROOT records have a key that consists of the publisher number, PUBNO. The BOOKINFO segment has a key that consists of that same publisher number, plus a hard- or soft-cover indicator, BINDING. The SERIANO segment key consists of the first two elements, plus a record type field, RECTYPE.

```
FILENAME = LIBRARY6, SUFFIX = DBFILE,$
 SEGNAME = ROOT, SEGTYPE = S0,$
  GROUP=PUBKEY         ,ALIAS=KEY   ,USAGE=A10   ,ACTUAL=A10  ,$
   FIELDNAME=PUBNO     ,ALIAS=PN    ,USAGE=A10   ,ACTUAL=A10  ,$
  FIELDNAME=FILLER     ,ALIAS=      ,USAGE=A1    ,ACTUAL=A1   ,$
  FIELDNAME=RECTYPE    ,ALIAS=1     ,USAGE=A1    ,ACTUAL=A1   ,$
  FIELDNAME=AUTHOR     ,ALIAS=AT    ,USAGE=A25   ,ACTUAL=A25  ,$
  FIELDNAME=TITLE      ,ALIAS=TL    ,USAGE=A50   ,ACTUAL=A50  ,$
 SEGNAME=BOOKINFO,PARENT=ROOT,   SEGTYPE=S0,$
  GROUP=BOINKEY        ,ALIAS=KEY   ,USAGE=A11   ,ACTUAL=A11  ,$
   FIELDNAME=PUBNO1    ,ALIAS=P1    ,USAGE=A10   ,ACTUAL=A10  ,$
   FIELDNAME=BINDING   ,ALIAS=BI    ,USAGE=A1    ,ACTUAL=A1   ,$
  FIELDNAME=RECTYPE    ,ALIAS=2     ,USAGE=A1    ,ACTUAL=A1   ,$
  FIELDNAME=PRICE      ,ALIAS=PR    ,USAGE=D8.2N ,ACTUAL=D8   ,$
 SEGNAME=SERIANO, PARENT=BOOKINFO,SEGTYPE=S0,$
  GROUP=SERIKEY        ,ALIAS=KEY   ,USAGE=A12   ,ACTUAL=A12  ,$
   FIELDNAME=PUBNO2    ,ALIAS=P2    ,USAGE=A10   ,ACTUAL=A10  ,$
   FIELDNAME=BINDING1  ,ALIAS=B1    ,USAGE=A1    ,ACTUAL=A1   ,$
   FIELDNAME=RECTYPE   ,ALIAS=3     ,USAGE=A1    ,ACTUAL=A1   ,$
  FIELDNAME=SERIAL     ,ALIAS=SN    ,USAGE=A15   ,ACTUAL=A15  ,$
 SEGNAME=SYNOPSIS,PARENT=ROOT,    SEGTYPE=S0, OCCURS=VARIABLE,$
  FIELDNAME=PLOTLINE   ,ALIAS=PLOTL ,USAGE=A10   ,ACTUAL=A10  ,$
```

Notice that the length of the key fields specified in the USAGE and ACTUAL attributes of a GROUP declaration is the length of the key fields from the parent segments, plus the length of the added field of the child segment (RECTYPE field). In the example above, the length of the GROUP key SERIKEY equals the length of PUBNO2 and BINDING1, the group key from the parent segment, plus the length of RECTYPE, the field added to the group key in the child segment. The length of the key increases as you traverse the structure.

Note: Each segment key describes as much of the true key as needed to find the next instance of that segment.

In the sample data source, the repetition of the publisher's number as PUBNO1 and PUBNO2 in the descendant segments interrelates the three types of records. The data source can be diagrammed as the following structure:



A typical query may request information on price and call numbers for a specific publisher number:

```
PRINT PRICE AND SERIAL BY PUBNO
IF PUBNO EQ 1234567890 OR 9876054321
```

Since PUBNO is part of the key, retrieval can occur quickly, and the processing continues. To further speed retrieval, add search criteria based on the BINDING field, which is also part of the key.

Describing Unrelated Records

Some data sources do not have records that are related to one another. That is, the key of one record type is independent of the keys of other record types. To describe data sources with unrelated records, define a dummy root segment for the record types. The following rules apply to the dummy root segment:

- The name of the root segment must be DUMMY.
- It must have only one field with a blank name and alias.
- The USAGE and ACTUAL attributes must both be A1.

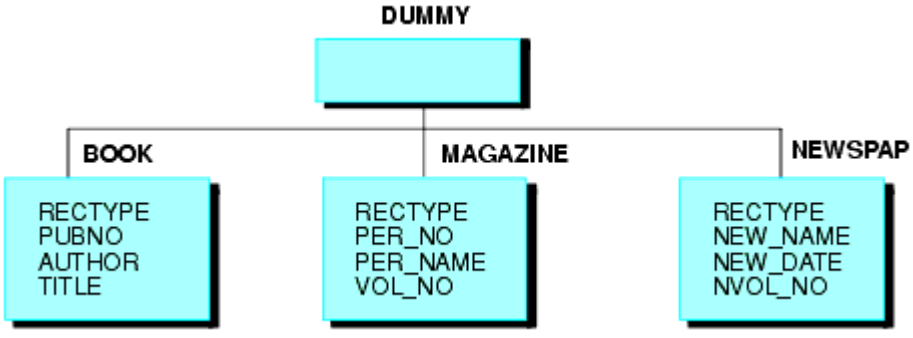
All other non-repeating segments must point to the dummy root as their parent. Except for the root, all non-repeating segments must have a RECTYPE and a PARENT attribute and describe the full key. If the data source does not have a key, the group should not be described. RECTYPEs may be anywhere in the record.

Example: Describing Unrelated Records Using a Dummy Root Segment

The library data source has three types of records: book information, magazine information, and newspaper information. Since these three record types have nothing in common, they cannot be described as parent records followed by detail records.

The data source can look like this:

A structure such as the following can also describe this data source.



The Master File for the structure in this example is:

```
FILENAME = LIBRARY3, SUFFIX = FIX,$
 SEGMENT = DUMMY, SEGTYPE = S0,$
  FIELDNAME=           ,ALIAS=     ,USAGE = A1    ,ACTUAL = A1   ,$
 SEGMENT = BOOK, PARENT = DUMMY, SEGTYPE = S0,$
  FIELDNAME = RECTYPE  ,ALIAS = B  ,USAGE = A1    ,ACTUAL = A1   ,$
  FIELDNAME = PUBNO    ,ALIAS = PN ,USAGE = A10   ,ACTUAL = A10  ,$
  FIELDNAME = AUTHOR   ,ALIAS = AT ,USAGE = A25   ,ACTUAL = A25  ,$
  FIELDNAME = TITLE    ,ALIAS = TL ,USAGE = A50   ,ACTUAL = A50  ,$
  FIELDNAME = BINDING  ,ALIAS = BI ,USAGE = A1    ,ACTUAL = A1   ,$
  FIELDNAME = PRICE    ,ALIAS = PR ,USAGE = D8.2N ,ACTUAL = D8   ,$
  FIELDNAME = SERIAL   ,ALIAS = SN ,USAGE = A15   ,ACTUAL = A15  ,$
  FIELDNAME = SYNOPSIS ,ALIAS = SY ,USAGE = A150  ,ACTUAL = A150 ,$
 SEGMENT = MAGAZINE, PARENT = DUMMY, SEGTYPE = S0,$
  FIELDNAME = RECTYPE  ,ALIAS = M  ,USAGE = A1    ,ACTUAL = A1   ,$
  FIELDNAME = PER_NO   ,ALIAS = PN ,USAGE = A10   ,ACTUAL = A10  ,$
  FIELDNAME = PER_NAME ,ALIAS = NA ,USAGE = A50   ,ACTUAL = A50  ,$
  FIELDNAME = VOL_NO   ,ALIAS = VN ,USAGE = I2    ,ACTUAL = I2   ,$
  FIELDNAME = ISSUE_NO ,ALIAS = IN ,USAGE = I2    ,ACTUAL = I2   ,$
  FIELDNAME = PER_DATE ,ALIAS = DT ,USAGE = I6MDY ,ACTUAL = I6   ,$
 SEGMENT = NEWSPAP, PARENT = DUMMY, SEGTYPE = S0,$
  FIELDNAME = RECTYPE  ,ALIAS = N  ,USAGE = A1    ,ACTUAL = A1   ,$
  FIELDNAME = NEW_NAME ,ALIAS = NN ,USAGE = A50   ,ACTUAL = A50  ,$
  FIELDNAME = NEW_DATE ,ALIAS = ND ,USAGE = I6MDY ,ACTUAL = I6   ,$
  FIELDNAME = NVOL_NO  ,ALIAS = NV ,USAGE = I2    ,ACTUAL = I2   ,$
  FIELDNAME = ISSUE    ,ALIAS = NI ,USAGE = I2    ,ACTUAL = I2   ,$
```

Example: Describing a DB Heritage Files Data Source With Unrelated Records

Consider another data source containing information on the library. This data source has three types of records: book information, magazine information, and newspaper information.

There are two possible structures:

The RECTYPE is the beginning of the key. The key structure is:

RECTYPE B	Book Code
RECTYPE M	Magazine Code
RECTYPE N	Newspaper Code

The sequence of records is:

Book
Book
Magazine
Magazine
Newspaper
Newspaper

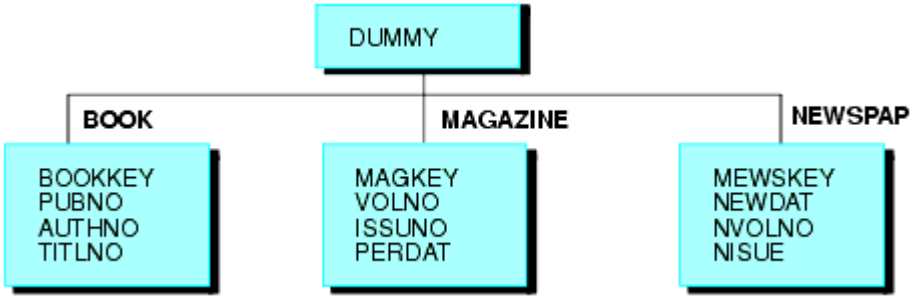
Note the difference between the use of the RECTYPE here and its use when the records are positionally related. In this case, the codes are unrelated and the database designer has chosen to accumulate the records by type first (all the book information together, all the magazine information together, and all the newspaper information together), so the RECTYPE may be the initial part of the key.

The RECTYPE is not in the beginning of the key or is outside of the key. The key structure is:

Book Code
Magazine Code
Newspaper Code

The sequence of record types in the data source can be arbitrary.

Both types of file structure can be represented by the following:



Example: **Describing a Key and a Record Type for a DB Heritage Files Data Source With Unrelated Records**

```
FILE=LIBRARY7, SUFFIX=DBFILE,$
 SEGMENT=DUMMY,$
  FIELDNAME=          ,ALIAS=    ,USAGE=A1     ,ACTUAL=A1   ,$
 SEGMENT=BOOK,      PARENT=DUMMY,SEGTYPE=S0,$
  GROUP=BOOKKEY       ,ALIAS=KEY ,USAGE=A11    ,ACTUAL=A11  ,$
   FIELDNAME=PUBNO    ,ALIAS=PN  ,USAGE=A3     ,ACTUAL=A3   ,$
   FIELDNAME=AUTHNO   ,ALIAS=AN  ,USAGE=A3     ,ACTUAL=A3   ,$
   FIELDNAME=TITLNO   ,ALIAS=TN  ,USAGE=A4     ,ACTUAL=A4   ,$
   FIELDNAME=RECTYPE  ,ALIAS=B   ,USAGE=A1     ,ACTUAL=A1   ,$
  FIELDNAME=AUTHOR    ,ALIAS=AT  ,USAGE=A25    ,ACTUAL=A25  ,$
  FIELDNAME=TITLE     ,ALIAS=TL  ,USAGE=A50    ,ACTUAL=A50  ,$
  FIELDNAME=BINDING   ,ALIAS=BI  ,USAGE=A1     ,ACTUAL=A1   ,$
  FIELDNAME=PRICE     ,ALIAS=PR  ,USAGE=D8.2N  ,ACTUAL=D8   ,$
  FIELDNAME=SERIAL    ,ALIAS=SN  ,USAGE=A15    ,ACTUAL=A15  ,$
  FIELDNAME=SYNOPSIS  ,ALIAS=SY  ,USAGE=A150   ,ACTUAL=A150 ,$
 SEGMENT=MAGAZINE,  PARENT=DUMMY,  SEGTYPE=S0,$
  GROUP=MAGKEY        ,ALIAS=KEY ,USAGE=A11    ,ACTUAL=A11  ,$
   FIELDNAME=VOLNO    ,ALIAS=VN  ,USAGE=A2     ,ACTUAL=A2   ,$
   FIELDNAME=ISSUNO   ,ALIAS=IN  ,USAGE=A2     ,ACTUAL=A2   ,$
   FIELDNAME=PERDAT   ,ALIAS=DT  ,USAGE=A6     ,ACTUAL=A6   ,$
   FIELDNAME=RECTYPE  ,ALIAS=M   ,USAGE=A1     ,ACTUAL=A1   ,$
  FIELDNAME=PER_NAME  ,ALIAS=PRN ,USAGE=A50    ,ACTUAL=A50  ,$
 SEGMENT=NEWSPAP,   PARENT=DUMMY,  SEGTYPE=S0,$
  GROUP=NEWSKEY       ,ALIAS=KEY ,USAGE=A11    ,ACTUAL=A11  ,$
   FIELDNAME=NEWDAT   ,ALIAS=ND  ,USAGE=A6     ,ACTUAL=A6   ,$
   FIELDNAME=NVOLNO   ,ALIAS=NV  ,USAGE=A2     ,ACTUAL=A2   ,$
   FIELDNAME=NISSUE   ,ALIAS=NI  ,USAGE=A2     ,ACTUAL=A2   ,$
   FIELDNAME=RECTYPE  ,ALIAS=N   ,USAGE=A1     ,ACTUAL=A1   ,$
  FIELDNAME=NEWNAME   ,ALIAS=NN  ,USAGE=A50    ,ACTUAL=A50  ,$
```

Using a Generalized Record Type

If your data source has multiple record types that share the same layout, you can specify a single generalized segment that describes all record types that have the common layout. By using a generalized segment, also known as a generalized RECTYPE, instead of one segment per record type, you reduce the number of segments you need to describe in the Master File.

When using a generalized segment, you identify RECTYPE values using the ACCEPT attribute. You can assign any value to the ALIAS attribute.

Syntax: How to Specify a Generalized Record Type

```
FIELDNAME = RECTYPE, ALIAS = alias, USAGE = format, ACTUAL = format,
ACCEPT = {list|range} ,$
```

where:

`RECTYPE`

Is the required field name.

Note: Since the field name, RECTYPE, may not be unique across segments, you should not use it in this way unless you qualify it. An alias is not required; you may leave it blank.

`alias`

Is any valid alias specification. You can specify a unique name as the alias value for the RECTYPE field only if you use the ACCEPT attribute. The alias can then be used in a TABLE request as a display field, a sort field, or in selection tests using either WHERE or IF.

`list`

Is a list of one or more lines of specific RECTYPE values for records that have the same segment layout. The maximum number of characters allowed in the list is 255. Each item in the list must be separated by either a blank or the keyword OR. If the list contains embedded blanks or commas, it must be enclosed within single quotation marks. The list may contain a single RECTYPE value. For example:

```
FIELDNAME = RECTYPE, ALIAS = TYPEABC, USAGE = A1,
ACTUAL = A1, ACCEPT = A OR B OR C, $
```

`range`

Is a range of one or more lines of RECTYPE values for records that have the same segment layout. The maximum number of characters allowed in the range is 255. If the range contains embedded blanks or commas, it must be enclosed within single quotation marks.

To specify a range of values, include the lowest value, the keyword TO, and the highest value, in that order. For example:

```
FIELDNAME = RECTYPE, ALIAS = ACCTREC, USAGE = P3,
ACTUAL = P2, ACCEPT = 100 TO 200, $
```

Example: Using a Generalized Record Type

To illustrate the use of the generalized record type in a Master File, consider the following record layouts in the DOC data source. Record type DN is the root segment and contains the document number and title. Record types M, I, and C contain information about manuals, installation guides, and course guides, respectively. Notice that record types M and I have the same layout.

Record type DN:

```
---KEY---
+-------------------------------------------------------------------------+
DOCID   FILLER         RECTYPE     TITLE
+-------------------------------------------------------------------------+
```

Record type M:

```
--------KEY--------
+-------------------------------------------------------------------------+
MDOCID   MDATE         RECTYPE     MRELEASE          MPAGES     FILLER
+-------------------------------------------------------------------------+
```

Record type I:

```
--------KEY--------
+-------------------------------------------------------------------------+
IDOCID   IDATE         RECTYPE     IRELEASE          IPAGES     FILLER
+-------------------------------------------------------------------------+
```

Record type C:

```
--------KEY--------
+-------------------------------------------------------------------------+
CRSEDOC   CDATE        RECTYPE     COURSENUM  LEVEL  CPAGES     FILLER
+-------------------------------------------------------------------------+
```

Without the ACCEPT attribute, each of the four record types must be described as separate segments in the Master File. A unique set of field names must be provided for record type M and record type I, although they have the same layout.

The generalized RECTYPE capability enables you to code just one set of field names that applies to the record layout for both record type M and I. The ACCEPT attribute can be used for any RECTYPE specification, even when there is only one acceptable value.

```
FILENAME=DOC2, SUFFIX=DBFILE,$
SEGNAME=ROOT, SEGTYPE=SO,$
 GROUP=DOCNUM,     ALIAS=KEY,        A5,  A5,  $
  FIELD=DOCID,     ALIAS=SEQNUM,     A5,  A5,  $
 FIELD=FILLER,     ALIAS,=           A5,  A5,  $
 FIELD=RECTYPE,    ALIAS=DOCRECORD, A3,  A3,  ACCEPT = DN,$
 FIELD=TITLE,      ALIAS=,           A18, A18, $
SEGNAME=MANUALS, PARENT=ROOT, SEGTYPE=SO,$
 GROUP=MDOCNUM,    ALIAS=KEY, A10, A10,$
  FIELD=MDOCID,    ALIAS=MSEQNUM,    A5,  A5,  $
  FIELD=MDATE,     ALIAS=MPUBDATE,   A5,  A5,  $
 FIELD=RECTYPE,    ALIAS=MANUAL,     A3,  A3,  ACCEPT = M OR I,$
 FIELD=MRELEASE,   ALIAS=,           A7,  A7,  $
 FIELD=MPAGES,     ALIAS=,           I5,  A5,  $
 FIELD=FILLER,     ALIAS=,           A6,  A6,  $
SEGNAME=COURSES, PARENT=ROOT, SEGTYPE=SO,$
 GROUP=CRSEDOC,    ALIAS=KEY, A10, A10,$
  FIELD=CDOCID,    ALIAS=CSEQNUM,    A5,  A5,  $
  FIELD=CDATE,     ALIAS=CPUBDATE,   A5,  A5,  $
 FIELD=RECTYPE,    ALIAS=COURSE,     A3,  A3,  ACCEPT = C,$
 FIELD=COURSENUM, ALIAS=CNUM,       A4,  A4,  $
 FIELD=LEVEL,      ALIAS=,           A2,  A2,  $
 FIELD=CPAGES,     ALIAS=,           I5,  A5,  $
 FIELD=FILLER,     ALIAS=,           A7,  A7,  $
```

Combining Multiply-Occurring Fields and Multiple Record Types in DB Heritage Files

You can have two types of descendant segments in a single fixed-format DB Heritage Files data source:

- Descendant segments consisting of multiply occurring fields.
- Additional descendant segments consisting of multiple record types.

Describing a Multiply Occurring Field and Multiple Record Types

In the data structure shown below, the first record, of type 01, contains several different sequences of repeating fields, all of which must be described as descendant segments with an OCCURS attribute. The data source also contains two separate records, of types 02 and 03, which contain information that is related to that in record type 01.

The relationship between the records of various types is expressed as parent-child relationships. The children that contain record types 02 and 03 do not have an OCCURS attribute. They are distinguished from their parent by the field declaration where field=RECTYPE.

01	T1	N1	B1	B2	C1	C1	C1	D1	D1	D1	D1	D1	D1	D1	B1	B2	C1	C1	D1	D1	D1	D1	D1	D1	D1

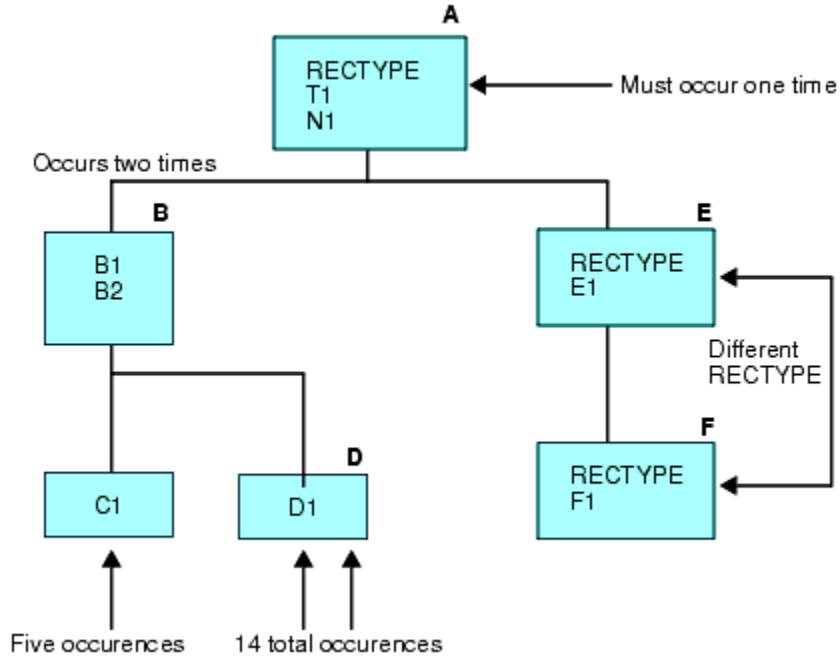
02	E1

03	F1

The description for this data source is:

```
FILENAME = EXAMPLE1, SUFFIX = FIX,$
 SEGNAME = A, SEGTYPE=S0,$
  FIELDNAME = RECTYPE   ,ALIAS = 01   ,USAGE = A2   ,ACTUAL = A2 ,$
  FIELDNAME = T1        ,ALIAS =      ,USAGE = A2   ,ACTUAL = A1 ,$
  FIELDNAME = N1        ,ALIAS =      ,USAGE = A1   ,ACTUAL = A1 ,$
 SEGNAME = B, PARENT = A, OCCURS = VARIABLE, SEGTYPE=S0,$
  FIELDNAME = B1        ,ALIAS =      ,USAGE = I2   ,ACTUAL = I2 ,$
  FIELDNAME = B2        ,ALIAS =      ,USAGE = I2   ,ACTUAL = I2 ,$
 SEGNAME = C, PARENT = B, OCCURS = B1, SEGTYPE=S0,$
  FIELDNAME = C1        ,ALIAS =      ,USAGE = A1   ,ACTUAL = A1 ,$
 SEGNAME = D, PARENT = B, OCCURS = 7,  SEGTYPE=S0,$
  FIELDNAME = D1        ,ALIAS =      ,USAGE = A1   ,ACTUAL = A1 ,$
 SEGNAME = E, PARENT = A, SEGTYPE=S0,$
  FIELDNAME = RECTYPE   ,ALIAS = 02   ,USAGE = A2   ,ACTUAL = A2 ,$
  FIELDNAME = E1        ,ALIAS =      ,USAGE = A1   ,ACTUAL = A1 ,$
 SEGNAME = F, PARENT = E, SEGTYPE=S0,$
  FIELDNAME = RECTYPE   ,ALIAS = 03   ,USAGE = A2   ,ACTUAL = A2 ,$
  FIELDNAME = F1        ,ALIAS =      ,USAGE = A1   ,ACTUAL = A1 ,$
```

It produces the following data structure:



Segments A, B, C, and D all belong to the same record type. Segments E and F each are stored as separate record types.

Note:

- Segments A, E, and F are different records that are related through their record types. The record type attribute consists of certain prescribed values, and is stored in a fixed location in the records. Records are expected to be retrieved in a given order. If the first record does not have a RECTYPE of 01, the record is considered to be a child without a parent. The next record can have a RECTYPE of either 01 (in which case, the first record is considered to have no descendants except the OCCURS descendants) or 02. A record with a RECTYPE of 03 can follow only a record with a RECTYPE of 02 (its parent) or another 03.
- The OCCURS descendants all belong to the record whose RECTYPE is 01. (This is not a necessary condition; records of any type can have OCCURS descendants.) Note that the OCCURS=VARIABLE segment, Segment B, is the rightmost segment within its own record type. If you look at the data structure, the pattern that makes up Segment B and its descendants (the repetition of fields B1, B2, C1, and D1) extends from the first mention of fields B1 and B2 to the end of the record.

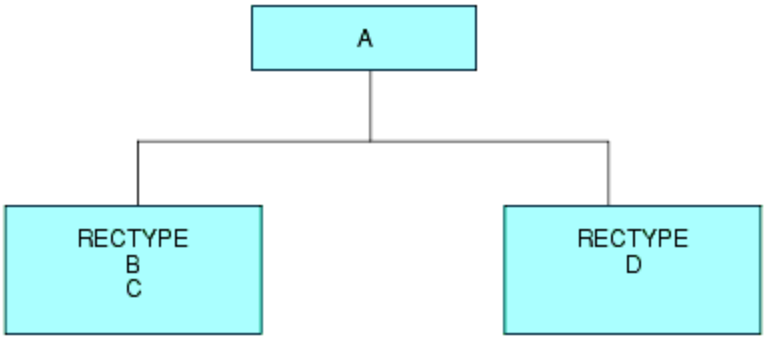
- Although fields C1 and D1 appear in separate segments, they are actually part of the repeating pattern that makes up the OCCURS=VARIABLE segment. Since they occur multiple times within Segment B, they are each assigned to their own descendant segment. The number of times field C1 occurs depends on the value of field B2. In the example, the first value of field B2 is 3; the second, 2. Field D1 occurs a fixed number of times, 7.

Describing a Repeating Group With RECTYPEs

Suppose you want to describe a data source that, schematically, looks like this:

```
A     RECTYPE  B C    RECTYPE   B C
A     RECTYPE  D      RECTYPE   D
```

You need to describe three segments in your Master File, with A as the root segment, and segments for B, C, and D as two descendant OCCURS segments for A.



Each of the two descendant OCCURS segments in this example depends on the RECTYPE indicator that appears for each occurrence.

All the rules of syntax for using RECTYPE fields and OCCURS segments also apply to RECTYPEs within OCCURS segments.

Since each OCCURS segment depends on the RECTYPE indicator for its evaluation, the RECTYPE must appear at the start of the OCCURS segment. This enables you to describe complex data sources, including those with nested and parallel repeating groups that depend on RECTYPEs.

Example: Describing a Repeating Group With RECTYPEs

In this example, B/C, and D represent a nested repeating group, and E represents a parallel repeating group.

A	RECTYPE B C	RECTYPE D	RECTYPE E	RECTYPE E

```
FILENAME=SAMPLE,SUFFIX=DBFILE,$
 SEGNAME=ROOT,SEGTYPE=S0,$
  GROUP=GRPKEY      ,ALIAS=KEY ,USAGE=A8 ,ACTUAL=A8 ,$
   FIELD=FLD000     ,E00       ,A08      ,A08       ,$
   FIELD=A_DATA     ,E01       ,A02      ,A02       ,$
 SEGNAME=SEG001,PARENT=ROOT,OCCURS=VARIABLE,SEGTYPE=S0  ,$
  FIELD=RECTYPE     ,A01       ,A01      ,ACCEPT=B OR C ,$
  FIELD=B_OR_C_DATA ,E02       ,A08      ,A08       ,$
 SEGNAME=SEG002,PARENT=SEG001,OCCURS=VARIABLE,SEGTYPE=S0,$
  FIELD=RECTYPE     ,D         ,A01      ,A01       ,$
  FIELD=D_DATA      ,E03       ,A07      ,A07       ,$
 SEGNAME=SEG003,PARENT=ROOT,OCCURS=VARIABLE,SEGTYPE=S0  ,$
  FIELD=RECTYPE     ,E         ,A01      ,A01       ,$
  FIELD=E_DATA      ,E04       ,A06      ,A06       ,$
```

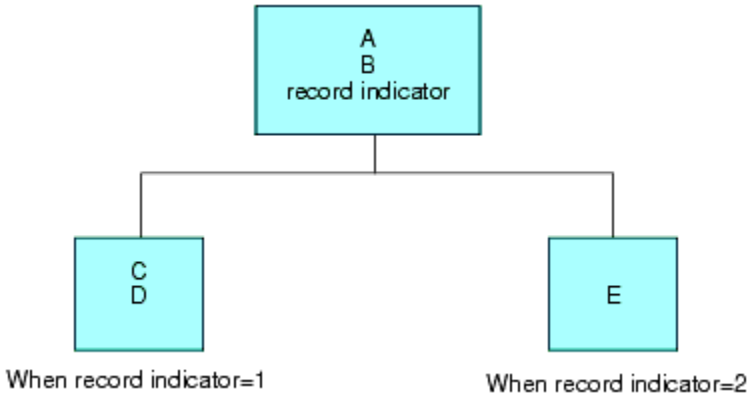
Describing a Repeating Group Using MAPFIELD

In another combination of record indicator and OCCURS, a record contains a record indicator that is followed by a repeating group. In this case, the record indicator is in the fixed portion of the record, not in each occurrence. Schematically, the record appears like this:

A B	record indicator (1)	C D	C D	C D
A B	record indicator (2)	E E		

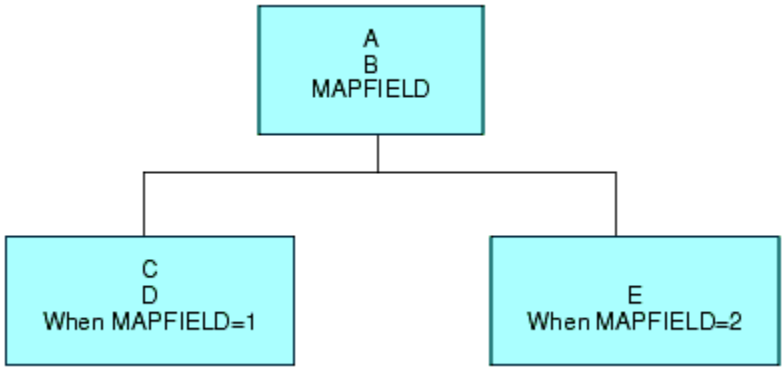
The first record contains header information, values for A and B, followed by an OCCURS segment of C and D that was identified by its preceding record indicator. The second record has a different record indicator and contains a different repeating group, this time for E.

The following diagram illustrates this relationship.



Since the OCCURS segments are identified by the record indicator rather than the parent A/B segment, you must use the keyword MAPFIELD. MAPFIELD identifies a field in the same way as RECTYPE, but since each OCCURS segments has its own value for MAPFIELD, the value of MAPFIELD is associated with each OCCURS segment by means of a complementary field named MAPVALUE.

The following diagram illustrates this relationship.



MAPFIELD is assigned as the ALIAS of the field that is the record indicator. It may have any name.

Syntax: How to Describe a Repeating Group With MAPFIELD

```
FIELD = name, ALIAS = MAPFIELD, USAGE = format, ACTUAL = format,$
```

where:

name

Is the name you choose to provide for this field.

ALIAS

MAPFIELD is assigned as the alias of the field that is the RECTYPE indicator.

USAGE

Follows the usual field format.

ACTUAL

Follows the usual field format.

The descendant segment values depend on the value of the MAPFIELD. They are described as separate segments, one for each possible value of MAPFIELD, and all descending from the segment that has the MAPFIELD. A special field, MAPVALUE, is described as the last field in these descendant segments after the ORDER field, if one has been used. The actual MAPFIELD value is supplied as the ALIAS of MAPVALUE.

Syntax: How to Use MAPFIELD for a Descendant Repeating Segment in a Repeating Group

```
FIELD = MAPVALUE, ALIAS = alias, USAGE = format, ACTUAL = format,
ACCEPT = {list|range} ,$
```

where:

MAPVALUE

Indicates that the segment depends on a MAPFIELD in its parent segment.

alias

Is the primary MAPFIELD value, if an ACCEPT list is not specified. If there is an ACCEPT list, this can be any value.

USAGE

Is the same format as the MAPFIELD format in the parent segment.

ACTUAL

Is the same format as the MAPFIELD format in the parent segment.

list

Is the list of one or more lines of specified MAPFIELD values for records that have the same segment layout. The maximum number of characters allowed in the list is 255. Each item in the list must be separated by either a blank or the keyword OR. If the list contains embedded blanks or commas, it must be enclosed within single quotation marks ('). The list may contain a single MAPFIELD value.

For example:

```
FIELDNAME = MAPFIELD, ALIAS = A, USAGE = A1, ACTUAL = A1,
ACCEPT = A OR B OR C,$
```

range

Is a range of one or more lines of MAPFIELD values for records that have the same segment layout. The maximum number of characters allowed in the range is 255. If the range contains embedded blanks or commas, it must be enclosed in single quotation marks (').

To specify a range of values, include the lowest value, the keyword TO, and the highest value, in that order.

Example: Using MAPFIELD and MAPVALUE

Using the sample data source at the beginning of this section, the Master File for this data source looks like this:

```
FILENAME=EXAMPLE,SUFFIX=FIX,$
 SEGNAME=ROOT,SEGTYPE=S0,$
  FIELD =A,               ,A14   ,A14 ,$
  FIELD =B,               ,A10   ,A10 ,$
  FIELD =FLAG ,MAPFIELD   ,A01   ,A01 ,$
 SEGNAME=SEG001,PARENT=ROOT,OCCURS=VARIABLE,SEGTYPE=S0  ,$
  FIELD =C,               ,A05   ,A05 ,$
  FIELD =D,               ,A07   ,A07 ,$
  FIELD =MAPVALUE ,1      ,A01   ,A01 ,$
 SEGNAME=SEG002,PARENT=ROOT,OCCURS=VARIABLE,SEGTYPE=S0  ,$
  FIELD =E,               ,D12.2 ,D8  ,$
  FIELD =MAPVALUE ,2      ,A01   ,A01 ,$
```

Note: MAPFIELD can only exist on an OCCURS segment that has not been remapped. This means that the segment definition cannot contain POSITION=*fieldname*.

MAPFIELD and MAPVALUE may be used with SUFFIX=FIX and SUFFIX=DBFILE data sources.

Multi-Format Logical Files

Multi-format logical files are an accumulation of different record formats, each one from a separate physical file. Multi-format logical files enable placement of multiple physical files in one logical (hierarchical) file structure. Each of the physical files should have common keys.

Example: Associating a Logical File With Three Physical Files

The following are IBM i Data Description Specifications (DDSs) for three physical files and the logical file that associates them:

```
EMPLOYEE Physical File DDS

0001.00      A                                      UNIQUE
0002.00      A          R EMPINFOR
0003.00      A            EMP_ID         9A
0004.00      A            LAST_NAME     10A
0005.00      A            FIRST_NAME    10A
0006.00      A            ADDRESS       40A
0007.00      A          K EMP_ID

PENSION Physical File DDS

0001.00      A                                      UNIQUE
0002.00      A          R PENSPATH
0003.00      A            EMP_ID         9A
0004.00      A            PLAN_NAME     10A
0005.00      A            VESTED         1A
0006.00      A            AMT_CNTRB      6P 2
0007.00      A          K EMP_ID

VACATION Physical File DDS

0001.00      A                                      UNIQUE
0002.00      A          R VACPATH
0003.00      A            EMP_ID         9A
0004.00      A            DAYS_EARND     2P 1
0005.00      A            DAYS_TAKEN     2P 1
0006.00      A          K EMP_ID

EMP Multi-Format Logical DDS

0001.00      A          R EMPINFOR                   PFILE(EMPLOYEE)
0002.00      A          K EMP_ID
0003.00      A          R PENSPATH                   PFILE(PENSION)
0004.00      A          K EMP_ID
0005.00      A          R VACPATH                    PFILE(VACATION)
0006.00      A          K EMP_ID
```

When you create a synonym for the EMP Multi-format logical file, the following metadata is generated.

```
FILENAME=EMP_MFL, SUFFIX=DBFILE  ,
 DATASET=QSYS:SHIPPING/EMP, $
  SEGMENT=EMPINFOR, SEGTYPE=S0, $
    FIELDNAME=EMP_ID, ALIAS=EMP_ID, USAGE=A9, ACTUAL=A9,
      TITLE='EMP_ID', $
    FIELDNAME=LAST_NAME, ALIAS=LAST_NAME, USAGE=A10, ACTUAL=A10,
      TITLE='LAST_NAME', $
    FIELDNAME=FIRST_NAME, ALIAS=FIRST_NAME, USAGE=A10, ACTUAL=A10,
      TITLE='FIRST_NAME', $
    FIELDNAME=ADDRESS, ALIAS=ADDRESS, USAGE=A40, ACTUAL=A40,
      TITLE='ADDRESS', $
    FIELDNAME=RECFORM, ALIAS=EMPINFOR, USAGE=A10, ACTUAL=A10, $

  SEGMENT=PENSPATH, SEGTYPE=S0, PARENT=EMPINFOR, $
    FIELDNAME=EMP_ID, ALIAS=EMP_ID, USAGE=A9, ACTUAL=A9,
      TITLE='EMP_ID', $
    FIELDNAME=PLAN_NAME, ALIAS=PLAN_NAME, USAGE=A10, ACTUAL=A10,
      TITLE='PLAN_NAME', $
    FIELDNAME=VESTED, ALIAS=VESTED, USAGE=A1, ACTUAL=A1,
      TITLE='VESTED', $
    FIELDNAME=AMT_CNTRB, ALIAS=AMT_CNTRB, USAGE=P8.2, ACTUAL=P4,
      TITLE='AMT_CNTRB', $
    FIELDNAME=RECFORM, ALIAS=PENSPATH, USAGE=A10, ACTUAL=A10, $

  SEGMENT=VACPATH, SEGTYPE=S0, PARENT=PENSPATH, $
    FIELDNAME=EMP_ID, ALIAS=EMP_ID, USAGE=A9, ACTUAL=A9,
      TITLE='EMP_ID', $
    FIELDNAME=DAYS_EARND, ALIAS=DAYS_EARND, USAGE=P4.1, ACTUAL=P2,
      TITLE='DAYS_EARND', $
    FIELDNAME=DAYS_TAKEN, ALIAS=DAYS_TAKEN, USAGE=P4.1, ACTUAL=P2,
      TITLE='DAYS_TAKEN', $
    FIELDNAME=RECFORM, ALIAS=VACPATH, USAGE=A10, ACTUAL=A10, $
```

The internal field, RECFORM, is needed to reference a particular record format. (This is done automatically when you create the synonym.)

The RECFORM ALIAS is required, even for single format files, and must always have a value of the name of the record format.

DB Heritage Files Record Selection Efficiencies

The most efficient way to retrieve selected records from a data source is by applying an IF screening test against the primary key. This results in a direct reading of the data using the data source's index. Only those records that you request are retrieved from the file. The alternative method of retrieval, the sequential read, forces the adapter to retrieve all the records into storage.

Selection criteria that are based on the entire primary key, or on a subset of the primary key, cause direct reads using the index. A partial key is any contiguous part of the primary key beginning with the first byte.

IF selection tests performed against virtual fields can take advantage of these efficiencies as well, if the full or partial key is embedded in the virtual field.

The EQ and IS relations realize the greatest performance improvement over sequential reads. When testing on a partial key, equality logic is used to retrieve only the first segment instance of the screening value. To retrieve subsequent instances, NEXT logic is used.

Screening relations GE, FROM, FROM-TO, GT, EXCEEDS, IS-MORE-THAN, and NOT-FROM-TO all obtain some benefit from direct reads. The following example uses the index to find the record containing primary key value 66:

```
IF keyfield GE 66
```

It then continues to retrieve records by sequential processing, because DB Heritage Files stores records in ascending key sequence. The direct read is not attempted when the IF screening conditions NE, IS-NOT, CONTAINS, OMITS, LT, IS-LESS-THAN, LE, and NOT-FROM are used in the report request.

Reporting From Files With Alternate Indexes

Similar performance improvement is available for files that use alternate indexes. An alternate index provides access to records in a key sequenced data set based on a key other than the primary key.

All benefits and limitations inherent with screening on the primary or partial key are applicable to screening on the alternate index or partial alternate index.

Note: It is not necessary to take an explicit indexed view to use a particular index.

Chapter 5

Using the Adapter for Esri ArcGIS

The ESRI GIS Adapter is used to report against the information residing in the ESRI ArcGIS Online environment or on a locally hosted ArcGIS server. For example, Data Enrichment information for a specific ZIP code can be analyzed.

You can configure the ESRI GIS Adapter using the Web Query Reporting Server Reporting Server browser interface. The adapter requires a connection, which stores the refresh token. A valid ESRI ArcGIS Online refresh token is required to issue ESRI ArcGIS Online API calls. This token is associated with an ESRI ArcGIS Online application and a specific ESRI ArcGIS Online user.

If your site is not using an on-premise ESRI server, the Reporting Server requires access to online resources in order for mapping functionality to work. The security rules at your site may block access, by default, and may require a specific list of URLs. ESRI documents the required URLs at the following location:

https://downloads.esri.com/resources/enterprisegis/AGOL_Domain_Requirements.pdf

Note: To configure a named connection with the Adapter for Esri ArcGIS by clicking *Get Data Connect* on the Web Query Home Page, the application server used by and the Web Query Reporting Server must both be running on the same machine. Otherwise, you cannot retrieve the refresh token for the adapter, and the Refresh Token field displays the text *Refresh Token unavailable*. In this case, configure the adapter from the Server Console, accessed through an HTTP listener, to acquire the token.

In this chapter:

Creating an ESRI ArcGIS Online Application

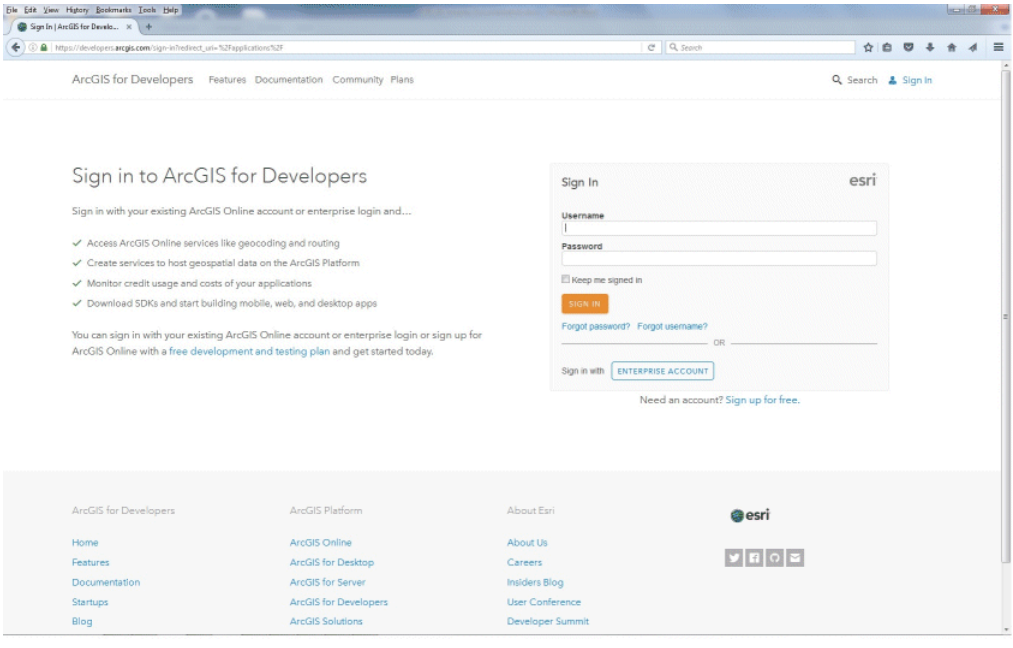
An ESRI ArcGIS Online application must be available before you can configure the ESRI GIS Adapter.

Procedure: How to Create an ESRI ArcGIS Application

1. Enter the following URL in a web browser:

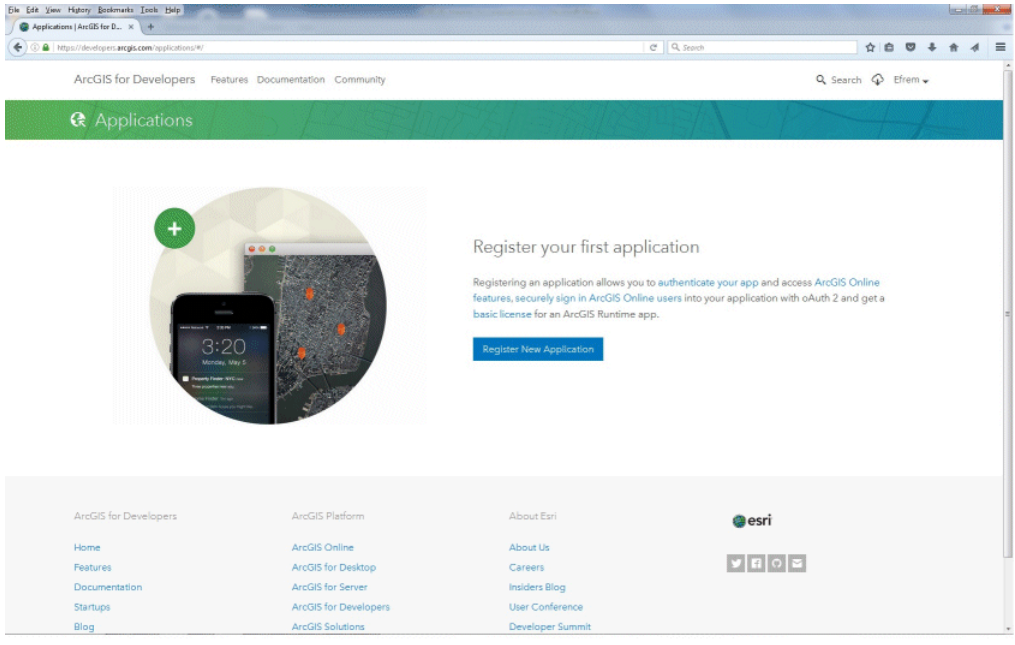
   `https://developers.arcgis.com/applications/#/`

   If you are not already signed into the ESRI, a sign-in dialog for the ArcGIS for Developers site opens, as shown in the following image.



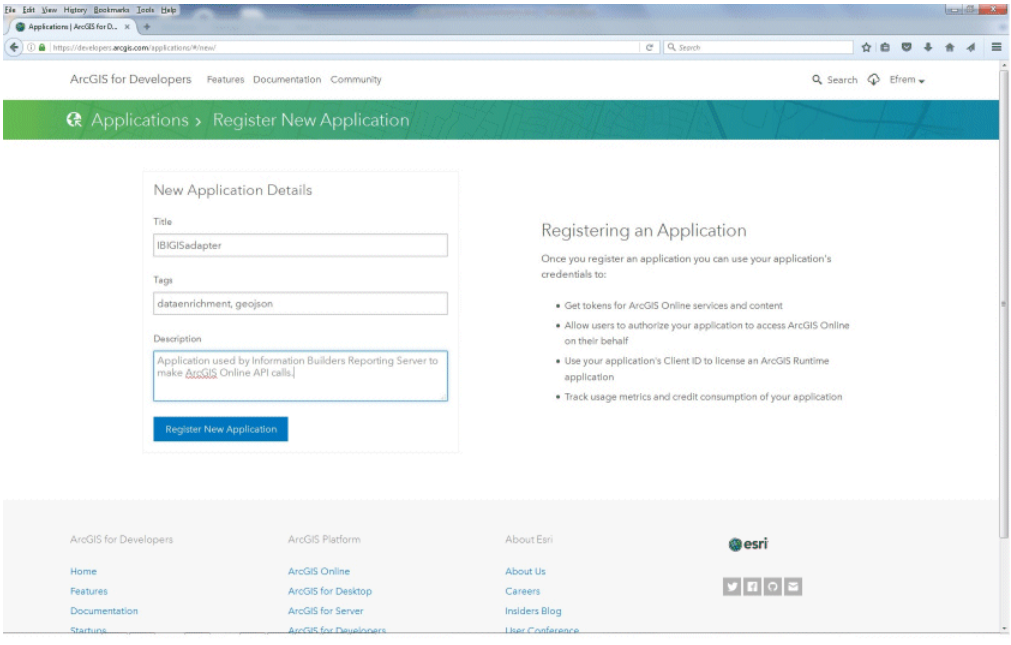
2. Enter valid ESRI ArcGIS Online account credentials that have the permission to create an ESRI ArcGIS Online application, and then click *Sign In*.

The Application screen opens, as shown in the following image.



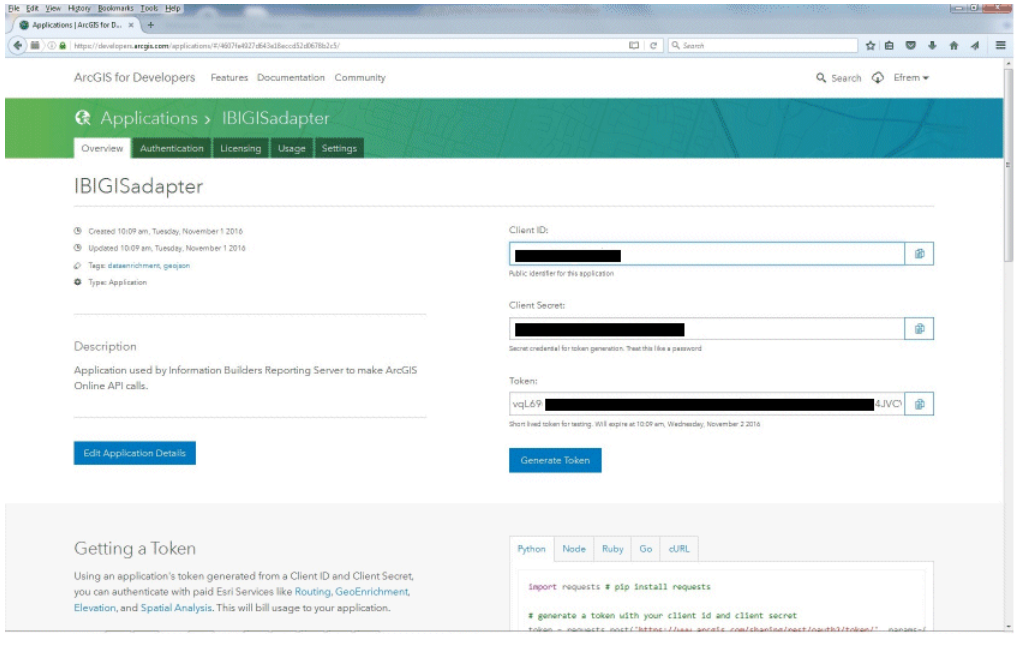
3. Click *Register New Application*.

The Register New Application screen opens, as shown in the following image.



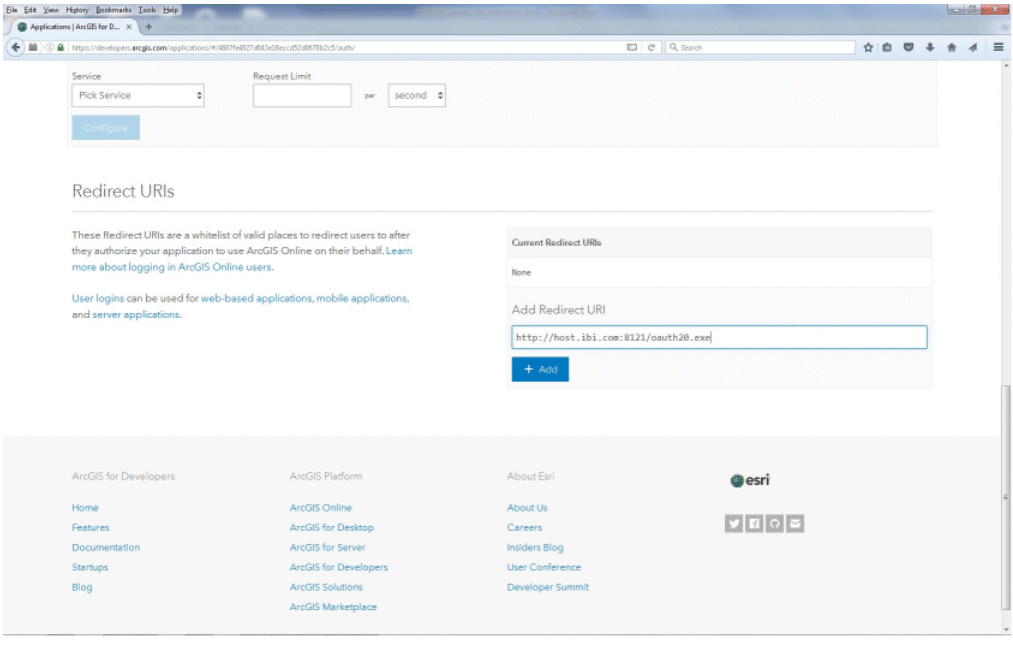
4. Enter a title, search tags, and a description for your new application and click *Register New Application*.

The Application Overview screen opens, which displays your Client ID and Client Secret values, as shown in the following image.



5. Click the *Authentication* tab.

Scroll down to the Redirect URIs section of the page, as shown in the following image.



6. Enter the host name and port used to access the Reporting Server Reporting Server browser interface appended with *oauth20.exe* in the Add Redirect URI field.

   For example:

   ```
   http://host.ibi.com:8121/oauth20.exe
   ```

   If the Reporting Server is installed as a standalone server, the following should be specified as the value in the Add Redirect URI field.

   ```
   http://localhost/oauth20.exe
   ```

7. Click + *Add*.

Configuring the Adapter for ESRI ArcGIS

This content describes how to configure the Adapter for ESRI ArcGIS.

Procedure: How to Configure the Adapter for ESRI ArcGIS for Non-Premium API Calls to ArcGIS Online

1. Clear cookies from the web browser that will be used to start the Reporting Server Reporting Server browser interface.

2. Access the Reporting Server Reporting Server browser interface using the host name and port that you specified in the Redirect URI field of the ESRI ArcGIS Online application.

   For example:

   ```
   http://host.ibi.com:8121
   ```

   For more information, see *Creating an ESRI ArcGIS Online Application* on page 93.

3. From the Reporting Server browser interface Applications page, click *Get Data*

   or

   From the Data Migrator desktop interface, expand the *Adapters* folder.

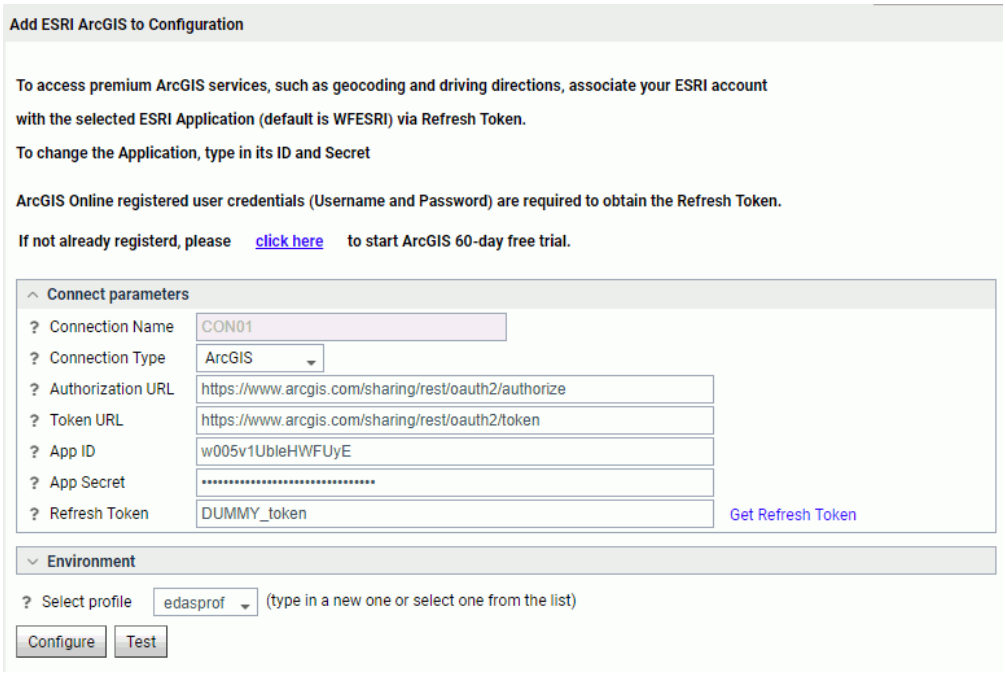
   In the Reporting Server browser interface, the Get Data page opens showing your configured adapters. In the Data Migrator desktop interface, the Adapters folder opens.

4. In the Reporting Server browser interface, click the *(+)* button, and find the adapter on the page or, in the Data Migrator desktop interface, expand the *Available* folder if it is not already expanded.

   In the Reporting Server browser interface, you can select a category of adapter from the drop-down list or use the search option (magnifying glass) to search for specific characters.

5. Right-click the ESRI ArcGIS node and click *Add connection*.

The Add ESRI ArcGIS to Configuration pane opens, as shown in the following image.

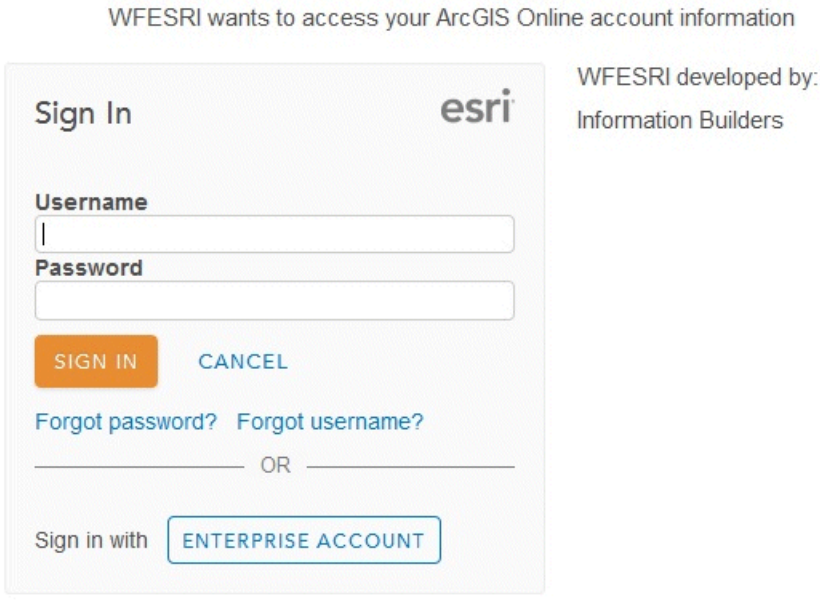


A default App ID and App Secret is pre-populated, to be used for Non-Premium ESRI ArcGIS Online API calls.

Data Enrichment is included in the Premium ESRI ArcGIS Online API calls.

6. Click *Get Refresh Token*.

The ESRI Sign In screen opens, as shown in the following image.



7. Enter the ESRI Sign In credentials and click *Sign In*.

   In order to obtain ESRI credentials, you can sign up at the following URL:

   https://developers.arcgis.com/sign-up/

   You are returned to the Add ESRI ArcGIS to Configuration pane where the Refresh Token is now populated.

8. Click *Configure*.

   The configured ESRI ArcGIS connection is added to the ESRI ArcGIS node in the left pane.

Procedure: How to Configure the Adapter for ESRI ArcGIS for Premium API Calls to ArcGIS Online

1. Clear cookies from the web browser that will be used to start the Reporting Server Reporting Server browser interface.
2. Access the Reporting Server Reporting Server browser interface using the host name and port that you specified in the Redirect URI field of the ESRI ArcGIS Online application.

   For example:

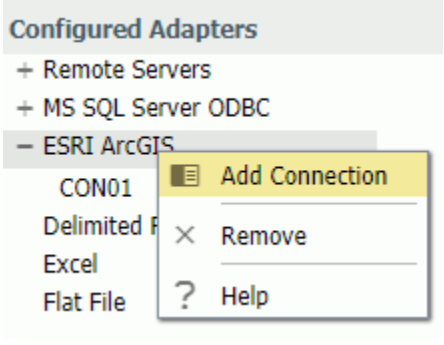
   `http://host.ibi.com:8121`

For more information, see *Creating an ESRI ArcGIS Online Application* on page 93.

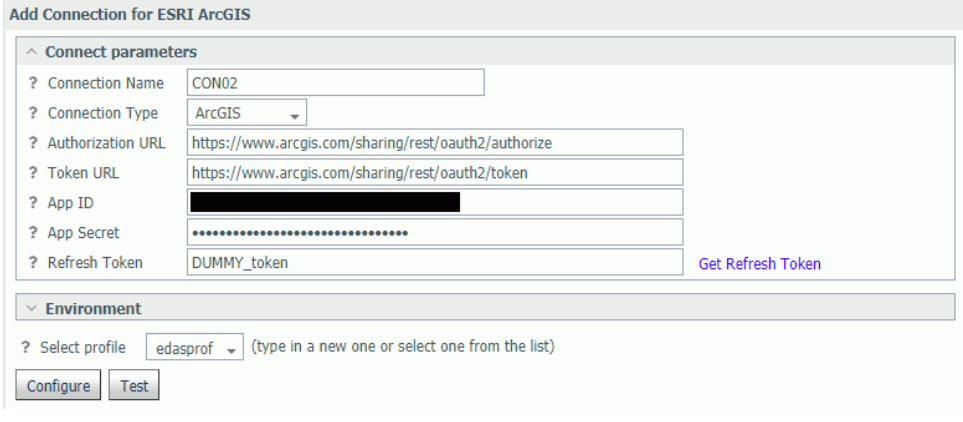
3. From the Reporting Server Reporting Server browser interface menu bar, click *Get Data*.

   The Get Data pane opens.

4. Expand the *Configured* folder, if it is not already expanded.
5. Expand the *GIS* folder.
6. Right-click the ESRI ArcGIS node and click *Add connection*, as shown in the following image.



   The Add Connection for ESRI ArcGIS pane opens, as shown in the following image.



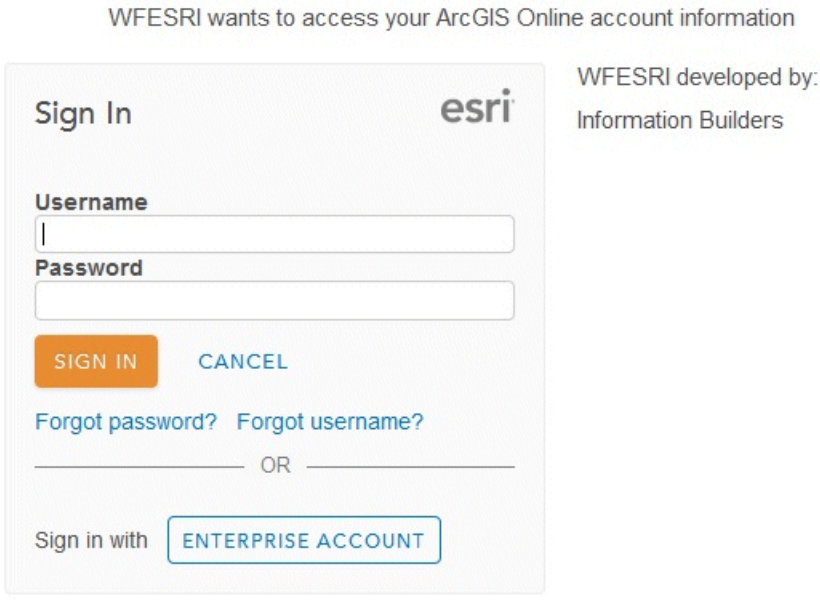
7. Enter the values for the App ID and App Secret defined in the ESRI application you created.

   For more information, see *Creating an ESRI ArcGIS Online Application* on page 93.

   Data Enrichment is included in the Premium ESRI ArcGIS Online API calls.

8. Click *Get Refresh Token*.

The ESRI Sign In screen opens, as shown in the following image.



9. Enter the ESRI Sign In credentials and click *Sign In*.

   In order to obtain ESRI credentials, you can sign up at the following URL:

   https://developers.arcgis.com/sign-up/

   You are returned to the Add ESRI ArcGIS to Configuration pane where the Refresh Token is now populated.

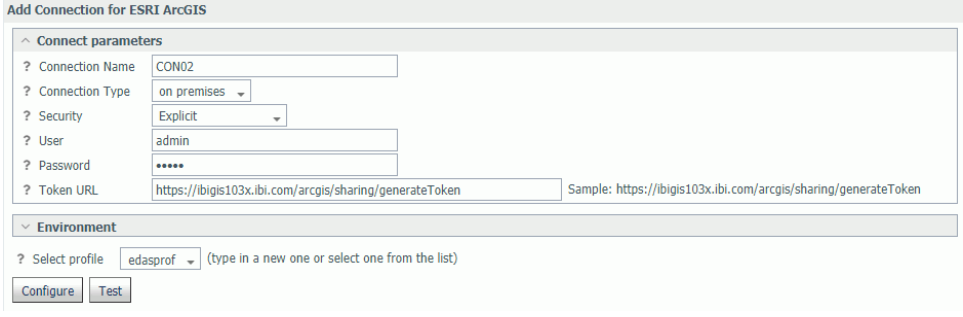
10. Click *Configure*.

    The configured ESRI ArcGIS connection is added to the ESRI ArcGIS node in the left pane.

Procedure: How to Configure a Locally Hosted ArcGIS Server

At some sites, ArcGIS Server is used in conjunction with or instead of ArcGIS Online. This type of configuration is called *Esri on Premise*.

1. When adding a connection for the Adapter for Esri ArcGIS, select *on premises* from the Connection Type drop-down list, as shown in the following image.



2. Select *Explicit* from the Security drop-down list.
3. Enter the user ID for the locally hosted ArcGIS Server in the User text box.
4. Enter the password for the locally hosted ArcGIS Server in the Password text box.
5. Enter the Token URL in the following format in the Token URL text box.

   `https://hostname/arcgis/sharing/generateToken`

   where:

   hostname

   Is the host where the ArcGIS Server is installed. For example:

   `https://ibigis103x.ibi.com/arcgis/sharing/generateToken`

6. Test the connection by clicking *Test*.
7. When the test is successful, click *Configure*.

Note: It is recommended to use Explicit authentication when connecting to a local ArcGIS Server. However, if you wish to use Password Passthru, to pass credentials from the Web Query Server to the ArcGIS Server, complete the following steps. The passed credentials must have administrative privileges on the Web Query Server.

1. On the Security tab of the Web Query Administration Console, set the Web Query Client to use Reporting Server security.
   a. In the Security Configuration folder, click *External*.

b. Select the *Enable External Security* check box.

c. Leave the External Security Type set to *Reporting Server*.

d. Select the Reporting Server Node where you are configuring the adapter for Esri ArcGIS.

e. Provide the username and password of a user with administrator privileges on the Web Query Server.

f. In the Actions panel, click *Save*.

2. On the Configuration tab of the Web Query Administration Console, set the connection to the Web Query Server from the Web Query Client to prompt for credentials.

   a. On the Configuration tab, in the Server Connections folder within the Reporting Servers folder, select the same server node that you selected in step 1d.

   b. Set the Security option to *Prompt for Credentials*.

   c. Click *Save*.

3. Configure the adapter with Esri ArcGIS to use Password Passthru security when connecting to the on-premise ArcGIS Server.

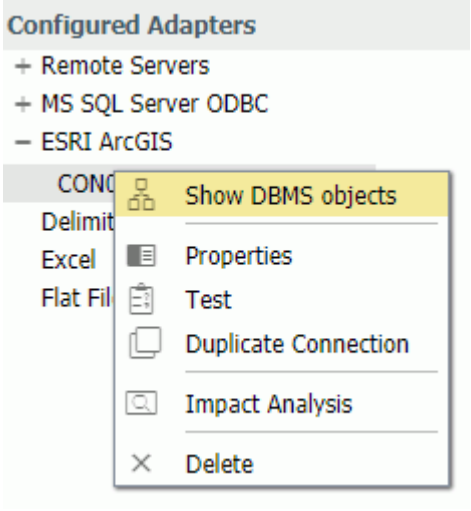
Creating Metadata and Sample Reports for the Adapter for ESRI ArcGIS Using Premium API Calls

Create Synonym for the Adapter for ESRI ArcGIS creates the metadata and sample Web Query reports used for Web Query reporting against data returned from the Premium ESRI ArcGIS Online API calls.

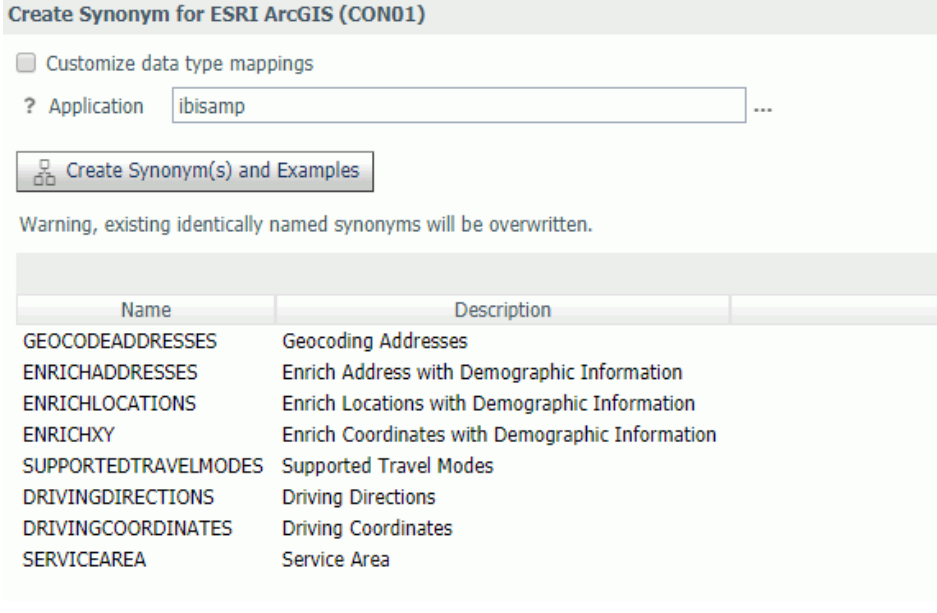
Procedure: How to How to Create Metadata and Sample Reports for Premium API Calls

1. From the DMC, expand the *Adapters* folder, then the *Configured* folder, and then the *ESRI ArcGIS* folder.

2. Right-click the configured connection for the ESRI ArcGIS connection (for example, CON02) that has access to Premium ArcGIS Online API calls, and click *Show DBMS objects* from the context menu, as shown in the following image.



The Create Synonym for ESRI ArcGIS pane opens, as shown in the following image.



3. Enter a specific application in the Application field, or click the ellipsis button to the right of the field to select an application where the metadata and sample reports are to be stored.
4. Click *Create Synonym(s) and Examples*.

   The Create Synonym for ESRI ArcGIS Status pane opens and indicates that the synonym was created successfully.

Sample Metadata and Reports

This section describes the metadata and sample reports for the Adapter for ESRI ArcGIS.

Reference: Adapter for ESRI ArcGIS Metadata

Metadata	Description
drivingdirections	Driving directions between geocoded coordinates
enrichaddresses	Enriches a radius around an address with demographic information
enrichlocations	Enriches locations with demographic information (for example, ZIP codes)
enrichxy	Enriches a radius around a geocoded coordinate with demographic information
geocodeaddresses	Geocodes specific addresses
servicearea	Geocoded coordinates that can be serviced around a specific location
supportedtravelmodes	Valid travel modes

Reference: Adapter for ESRI ArcGIS Sample Reports

The following table lists and describes the sample reports for the Adapter for ESRI ArcGIS.

Sample Report	Description
esrsampl/drivingcoordinates	Returns a series of geocoded coordinates for the driving directions between geocoded coordinates. Uses drivingdirections synonym.
esrsampl/drivingdirections	Returns the driving directions between geocoded coordinates. Uses drivingdirections synonym.
esrsampl/enrich_wfretail_us	Ennriches a ZIP code from the Retail Sales Demo Data with demographic information. Uses enrichlocations synonym.
esrsampl/enrichaddresses	Enriches a radius around an address with demographic information. Uses enrichaddresses synonym.
esrsampl/enrichlocations	Enriches one or more ZIP codes with demographic information. Uses enrichlocations synonym.
esrsampl/enrichxy	Enriches a radius around a geocoded coordinate with demographic information. Uses enrichxy synonym.
esrsampl/geocodeaddresses	Geocodes specific addresses. Uses geocodeaddresses synonym.
esrsampl/servicearea	Returns a list of geocoded coordinates that can be serviced around a specific location. Uses servicearea synonym.

Sample Report	Description
esrsampl/supportedtravelmodes	Returns a list of supported travel modes. Uses supportedtravelmodes synonym.

Chapter 6

Using the Adapter for JDBC

The Adapter for JDBC allows applications to access specific JDBC data sources. The adapter converts application requests into JDBC calls and returns optimized answer sets to the requesting application.

For details about JDBC-supported data sources, see the *Server Release Notes*.

In this chapter:

Preparing the JDBC Environment

In order to use the Adapter for JDBC, you must install the JDBC driver for whichever data source you would like to access, set its CLASSPATH value before server startup, and ensure that the JSCOM3 service is running.

For example, if you want to access a UniData database, you must install a JDBC driver for UniData. You can configure only one generic JDBC adapter per server instance.

Note:

- If an adapter for a specific data source type is available, you should use that adapter to access that type. The Adapter for JDBC does not support data source types for which a specific adapter already exists. For example, the Adapter for Db2 is available to access Db2 databases, so you should not use the Adapter for JDBC to access Db2.

- If a specific adapter for a data source type is not currently available, you may use the Adapter for JDBC with a vendor-provided JDBC driver. Note that the adapter conforms to generic JDBC standards. Vendors, on the other hand, may have interpreted or extended those standards for their specific data source requirements. If this introduces an incompatibility, you may experience inconsistent behavior when using the adapter for that vendor's data source type. If this occurs please contact Customer Support Services for assistance. They will work with your enterprise to try and support your data access needs.

Procedure: How to Set Up the Environment on Windows and UNIX

1. Identify the location of the JDBC Driver files by adding them to the environment variable CLASSPATH before server startup.

   For example, to set a UNIX or IBM i location for some JDBC Driver files to /usr/driver_files/mydriver.jar, issue the commands:

   ```
   CLASSPATH=/usr/driver_files/mydriver.jar;$CLASSPATH
   export CLASSPATH
   ```

   Similarly, on Windows, you would issue the command:

   ```
   set CLASSPATH=c:\usr\driver_files\mydriver.jar:%CLASSPATH%
   ```

   Note that if you run the server workspace as a Windows service, you must set the CLASSPATH as a system-wide environment variable.

   Similarly, on UNIX, to ensure that the variable is set before server startup, add the CLASSPATH setting in your UNIX profile.

2. Start (or restart) the server.

   (Note that you also need to restart the server if the driver has changed.)

3. Verify that the JSCOM3 service is active by issuing the following command in the server workspace

   ```
   edastart -show
   ```

   and checking that the special services section displays "JSCOM3 active".

   If the JSCOM3 service is not active, a "fail to start" message will typically also be in the server EDAPRINT log. To resolve the problem, review the JDBC listener requirements in the chapter for your operating system in the *Server Installation* manual.

Configuring the Adapter for JDBC

Configuring the adapter consists of specifying connection and authentication information for each of the connections you want to establish.

In order to connect to a JDBC data source, the adapter requires connection and authentication information.

You supply this information using the SET CONNECTION_ATTRIBUTES command. You can:

- Enter connection and authentication information in the Reporting Server browser interface or the Data Migrator desktop interface configuration pane. The consoles add the command to the server profile you select: global (edasprof.prf), user (user.prf), or group (group.prf) if supported on your platform.

Procedure: How to Configure an Adapter

1. From the Reporting Server browser interface Applications page, click *Get Data*

   or

   From the Data Migrator desktop interface, expand the *Adapters* folder.

   In the Reporting Server browser interface, the Get Data page opens showing your configured adapters. In the Data Migrator desktop interface, the Adapters folder opens.

2. In the Reporting Server browser interface, click the *(+)* button, and find the adapter on the page or, in the Data Migrator desktop interface, expand the *Available* folder if it is not already expanded.

   In the Reporting Server browser interface, you can select a category of adapter from the drop-down list or use the search option (magnifying glass) to search for specific characters.

3. In the Data Migrator desktop interface, expand the appropriate group folder and the specific adapter folder. The group folder is described in the connection attributes reference.

4. Right-click the adapter name and/or version and select *Configure*.

   The Add Adapter to Configuration pane opens.

5. Enter values for the parameters required by the adapter, as described in the chapter for the specific adapter you want to configure.

6. Click *Configure*. The configured adapter is added to the Configured list in the Reporting Server browser interface or in the Adapters list in the Data Migrator desktop interface resources tree.

   In the Reporting Server browser interface, the adapter remains on the Available Adapters list with an asterisk to indicate that at least one connection has been configured. You can configure additional connections from either the Configured or Available list by right-clicking the adapter and clicking *Add Connection*.

Reference: Connection Attributes for JDBC

The *JDBC* adapter is under the *SQL* group folder.

The following list describes the connection attributes for which you can supply values. To complete the attribute declaration, click the *Configure* button.

Connection name

Logical name used to identify this particular set of connection attributes. The default is CON01.

URL

Location URL for the JDBC data source.

Driver name

Name for the JDBC driver.

See driver documentation for the specific release you are using.

IBI_CLASSPATH

Defines the additional Java Class directories or full-path jar names which will be available for Java Services. Value may be set by editing the communications file or in the Reporting Server browser interface. Using the Reporting Server browser interface, you can enter one reference per line in the input field. When the file is saved, the entries are converted to a single string using colon (:) delimiters for all platforms. OpenVMS platforms must use UNIX-style conventions for values (for example, /mydisk/myhome/myclasses.jar, rather than mydisk:[myhome]myclasses.jar) when setting values. When editing the file manually, you must maintain the colon delimiter.

Security

There are two methods by which a user can be authenticated when connecting to a database server:

- **Explicit.** The user ID and password are explicitly specified for each connection and passed to the database, at connection time, for authentication.
- **Password Passthru.** The user ID and password received from the client application are passed to the database, at connection time, for authentication.

User

Primary authorization ID by which you are known to the data source.

Password

Password associated with the primary authorization ID.

Select profile

Select a profile from the drop-down menu to indicate the level of profile in which to store the CONNECTION_ATTRIBUTES command. The global profile, edasprof.prf, is the default.

If you wish to create a new profile, either a user profile (*user*.prf) or a group profile if available on your platform (using the appropriate naming convention), choose *New Profile* from the drop-down menu and enter a name in the Profile Name field (the extension is added automatically).

Store the connection attributes in the server profile (edasprof).

Syntax: How to Declare Connection Attributes Manually

```
ENGINE SQLJDBC SET CONNECTION_ATTRIBUTES connection
'URL'/userid,password
```

where:

`SQLJDBC`

Indicates the adapter. You can omit this value if you previously issued the SET SQLENGINE command.

`connection`

Is the connection name.

`URL`

Is the URL to the location of the JDBC data source.

`userid`

Is the primary authorization ID by which you are known to the target database.

`password`

Is the password associated with the primary authorization ID.

Example: Declaring Connection Attributes

The following SET CONNECTION_ATTRIBUTES command connects to data source using the JDBC Driver, with an explicit user ID (MYUSER) and password (PASS). To ensure security, specifying connection attributes from the Reporting Server browser interface or DMC will encrypt the password before adding it to the server profile.

```
ENGINE SQLJDBC SET CONNECTION_ATTRIBUTES CON1
  'jdbc:xxxxxxx://hostname:port/datasource
```

Overriding the Default Connection

If multiple connections have been defined, the connection named in the *first* SET CONNECTION_ATTRIBUTES command serves as the default connection.

Syntax: How to Change the Default Connection

You change this setting by expanding the Domains and Reports folders, then right-clicking a subfolder and choosing *Metadata*. Right-click *SQLJDBC* and select *Change Setting*s. The Change Settings pane opens.

`SQLJDBC`

Indicates the adapter. You can omit this value if you previously issued the SQLENGINE command.

`connection`

Is the connection defined in a previously issued CONNECTION_ATTRIBUTES command. If this name was not previously declared, the following message is issued:

```
FOC1671, Command out of sequence
```

Note:

- If you use the DEFAULT_CONNECTION command more than once, the connection name specified in the *last* command serves as the default connection.
- The DEFAULT_CONNECTION command cannot be issued while an uncommitted transaction (LUW) is pending. In that case, the following message is issued:

```
FOC1671, Command out of sequence.
```

Controlling the Connection Scope

The SET AUTODISCONNECT command controls the persistence of connections when using the adapter for each of the connections you want to establish.

Syntax: How to Control the Connection Scope

You change this setting by expanding the Domains and Reports folders, then right-clicking a subfolder and choosing *Metadata*. Right-click *SQLJDBC* and select *Change Setting*s. The Change Settings pane opens.

`SQLJDBC`

Indicates the adapter. You can omit this value if you previously issued the SQLENGINE command.

`FIN`

Disconnects automatically only after the session has been terminated. FIN is the default value.

`COMMIT`

Disconnects automatically only after COMMIT or ROLLBACK is issued as a native SQL command.

Managing JDBC Metadata

When the server accesses a data source, it needs to know how to interpret the data stored there. For each object the server will access, you create a synonym that describes its structure and the server mapping of the data types.

Identifying the Adapter

The SUFFIX attribute in the Master File identifies the adapter needed to interpret a request. Use the SUFFIX value SQLJDBC to identify the Adapter for JDBC.

Syntax: How to Identify the Adapter

```
FILE[NAME]=file, SUFFIX=SQLJDBC [,$]
```

where:

file

Is the file name for the Master File. The file name without the .mas extension can consist of a maximum of eight alphanumeric characters. The file name should start with a letter and be representative of the table or view contents. The actual file must have a .mas extension, but the value for this attribute should not include the extension.

`SQLJDBC`

Is the value for the adapter.

Accessing Database Tables

If you choose to access a remote third-party table using JDBC, you must locally install the RDBMS' JDBC Driver.

The Server can access third-party database tables across the JDBC network. You must specify a URL for the data source and, possibly, a user ID and/or password for the database you are accessing. You can define these parameters in either the server's global profile or in a user profile.

Creating Synonyms

Synonyms define unique names (or aliases) for each object that is accessible from the server. Synonyms are useful because they hide the underlying data source location and identity from client applications. They also provide support for extended metadata features of the server, such as virtual fields and additional security mechanisms.

Using synonyms allows an object to be moved or renamed while allowing client applications to continue functioning without modification. The only modification required is a redefinition of the synonym on the server. The result of creating a synonym is a Master File and an Access File, which represent the server metadata.

Note: If you are creating a synonym for a JDBC data source, you must first add the syntax SET SYNONYM=BASIC to the edasprof.prf file.

Procedure: How to Create a Synonym

1. From the Reporting Server browser interface Application page, click *Get Data*.
2. On the Configured Adapters section of the page, in Simple Mode, right-click an adapter and click *Show Connections*. Right-click a connection.

   Depending on the type of adapter you choose, one of the following options appears on the context menu.

   - **Show DBMS objects.** This option opens the page for selecting synonym objects and properties.
   - **Create metadata objects.** This option opens the page for selecting synonym objects and properties.
   - **Show files.** This option opens a file picker. After you choose a file of the correct type, the page for selecting synonym objects and properties opens.
   - **Show local files.** This option opens a file picker. After you choose a file of the correct type, the page for selecting synonym objects and properties opens.
   - **Show topics.** This option opens the page for selecting synonym objects and properties for topics within the environment.
3. Enter values for the parameters required by the adapter as described in the chapter for your adapter.
4. After entering the parameter values, click *Add*.

   This button may be labeled *Next*, *Create Synonym*, *Create Base Synonyms*, *Create Cluster Synonym*, or *Update Base Synonyms*.

The synonym creation process for most adapters has been consolidated so that you can enter all necessary parameters on one page. However, for some adapters such as LDAP, continue clicking *Next* until you get to a page that has a *Create Synonym* button.

The synonym is created and added under the specified application directory.

Note: When creating a synonym, if you select the Validate check box, where available, the server adjusts special characters and checks for reserved words.

Reference: Synonym Creation Parameters for JDBC

The following list describes the synonym creation parameters for which you can supply values.

Restrict Object Type to

Restrict candidates for synonym creation based on the selected object type(s): Tables, Views, External SQL Scripts, and any other supported objects.

Choosing *External SQL Scripts* from the drop-down list enables you to represent an SQL Query as a synonym for read-only reporting. A Synonym candidate can be any file that contains one (and only one) valid SQL Query and does not contain end-of-statement delimiters (";" or "/") and comments.

Depending on the adapter, you can further restrict your search by choosing

Filter by Owner/Schema and Object name

Selecting this option adds the Owner/Schema and Object Name parameters to the screen.

- **Owner/Schema.** Type a string for filtering the selection, inserting the wildcard character (%) as needed at the beginning and/or end of the string. For example, enter: ABC% to select tables or views whose owner/schema begin with the letters ABC; %ABC to select tables or views whose owner/schema end with the letters ABC; %ABC% to select tables or views whose owner/schema contain the letters ABC at the beginning, middle, or end.

- **Object name.** Type a string for filtering the object names, inserting the wildcard character (%) as needed at the beginning and/or end of the string. For example, enter: ABC% to select all objects whose names begin with the letters ABC; %ABC to select all whose names end with the letters ABC; %ABC% to select all whose names contain the letters ABC at the beginning, middle, or end.

Location of External SQL Scripts

If you specify *External SQL Scripts* in the *Restrict Object type to* field, these additional fields are displayed.

The following standard naming conventions apply for UNIX, IBM i IFS, and z/OS HFS:

- In the *Base Location* field, specify the physical directory location of the file that contains the SQL Query. You can type a directory name or click on the ellipsis. This opens the Select Base Location dialog box.
- In the *Document Name* field, enter the file name with or without wild card characters.
- In the *Document Extension* field, enter the extension of the script files to filter the list of candidates.

On IBM i, you can use alternative IFS naming conventions to access library members. The following entry illustrates this method:

- In the *Base Location* field, enter:

  `/QSYS.LIB/MYLIBRARY.LIB/MYSRC.FILE`

- The *Document Extension* is understood to be MBR. You can enter this value explicitly or leave the input box blank.

During synonym generation, the adapter issues native API calls to obtain a list of elements in the select list and builds the Master File with a field for each element. The generated Access File references the location of the SQL script in the DATASET attribute, which contains the full path, including the name and extension of the file containing the SQL Query. For example,

```
DATASET=/ul/home2/apps/report3.sql
```

When a Web Query report is created, the SQL Query is used to access data.

Cardinality

Select the *Cardinality* check box to reflect the current cardinality (number of rows or tuples) in the table during metadata creation. Cardinality is used for equi-joins. The order of retrieval is based on the size (cardinality) of the table. Smaller tables are read first.

If the cardinality of the tables to be used in the application are dynamic, it may not be beneficial to choose this setting.

Build cluster using foreign keys (deprecated)

You can select the *Build cluster using foreign keys* check box to include within this synonym every table related to the current table by a foreign key. However, this option has been deprecated, as the recommended way to create a cluster is by using the Synonym Editor. The resulting multi-table synonym describes all of the foreign key relationships of this table.

For Subquery

Only available when *External SQL Scripts* is selected from the Restrict objects type to drop-down menu. When selected, a SUBQUERY keyword is added to the Access File of the generated synonym. If the corresponding SQL string has valid syntax that can be used in the FROM statement of the generated SQL (what is known as a Derived Table), then the SQL SCRIPT will be processed as a subquery embedded into a FROM clause. This usage allows for more flexibility. For example, the synonym can be used as a target for a JOIN.

If the SQL SCRIPT has parameter markers, such as ? or :, or the syntax contains constructs that are invalid for a derived table, for example ORDER BY, then this keyword should not be selected. At runtime, if SUBQUERY=Y is present and it is determined that the SQL SCRIPT cannot be used in the FROM statement, the setting will be ignored, and a FOC1782 warning message will be issued. The default is selected (SUBQUERY=Y).

Application

The default value is baseapp.

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 the prefix and suffix fields blank.

Customize data type mappings

To change the data type mappings from their default settings, select this check box. The customizable mappings are displayed.

For information about customizable mappings, see *Data Type Support Report* on page 124.

Update or Create Metadata

Select *Create* to overwrite any existing synonym with the same fully-qualified name, or *Update* to synchronize the metadata with an existing synonym. If you select *Update*, the next screen will show a list of attributes from the DBMS catalog that you can check to allow attributes from the DBMS catalog to override attributes from the existing synonym.

Overwrite Existing Synonyms

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.

Default Synonym Name

This column displays the name that will be assigned to each synonym. To assign a different name, replace the displayed value.

Owner/Schema

The user account that created the object or a collection of objects owned by a user.

Table name

Is the name of the underlying object.

Type

The object type (Table, View, and so on).

Select tables

Select tables for which you wish to create synonyms:

- To select all tables in the list, select the *Select All* check box.
- To select specific tables, select the corresponding check boxes.

Example: Sample Generated Synonym

An Adapter for JDBC synonym comprises a Master File and an Access File. This is a synonym for the table nf29004.

Master File nf29004.mas

```
FILE=DIVISION, SUFFIX=SQLJDBC ,$
SEGNAME=SEG1_4, SEGTYPE=S0 ,$
FIELD=DIVISION4,    DIVISION4, I9, I4, MISSING=OFF ,$
FIELD=DIVISION_NA4, DIVISION4, I9, I4, MISSING=OFF ,$
FIELD=DIVISION_HE4, DIVISION4, I9, I4, MISSING=OFF ,$
```

Access File nf29004.acx

```
SEGNAME=SEG1_4,TABLENAME=EDAQA.NF29004,
CONNECTION=CON1,KEYS=1,WRITE=YES,$
```

Reference: Access File Keywords

This chart describes the keywords in the Access File.

Keyword	Description
SEGNAME	Value must be identical to the SEGNAME value in the Master File.
TABLENAME	Identifies the JDBC table. The value assigned to this attribute can include the name of the owner (also known as schema) and the database link name as follows: `TABLENAME=[owner.]table`
CONNECTION	Indicates a previously declared connection. The syntax is: `CONNECTION=connection` CONNECTION=' ' indicates access to the local data source. Absence of the CONNECTION attribute indicates access to the default database server.
KEYS	Indicates how many columns constitute the primary key for the table. Corresponds to the first *n* fields in the Master File segment. See the KEY attribute below for information about specifying the key fields without having to describe them first in the Master File.
KEY	Specifies the columns that participate in the primary key without having to describe them as the first fields in the Master File. The syntax is: `KEY=fld1/fld2/.../fldn`
WRITE	Specifies whether write operations are allowed against the table.

Keyword	Description
KEYFLD IXFLD	Supply the names of the primary key and foreign key fields that implement the relationships established by the multi-table Master File. Together, KEYFLD and IXFLD identify the field shared by a related table pair. ❑ KEYFLD is the FIELDNAME of the common column from the parent table. ❑ IXFLD is the FIELDNAME of the common column from the related table. KEYFLD and IXFLD must have the same data type. It is recommended, but not required, that their lengths also be the same. **Note:** An RDBMS index on both the KEYFLD and IXFLD columns provides the RDBMS with a greater opportunity to produce efficient joins. The columns must have the same data type. If their length is the same, the RDBMS handles the join more efficiently.

Reference: Managing Synonyms

Once you have created a synonym, you can right-click the synonym name in the navigation pane to access the available options.

Data Type Support Report

SQL Data Type mapping options are available in a report available from the Reporting Server browser interface.

Customizing the JDBC Environment

The Adapter for JDBC provides several parameters for customizing the environment and optimizing performance.

Specifying a Timeout Limit

TIMEOUT specifies the number of seconds the adapter will wait for a response after you issue an SQL request to JDBC.

Syntax: How to Issue the TIMEOUT Command

```
ENGINE SQLJDBC SET TIMEOUT {nn|0}
```

where:

`SQLJDBC`

Indicates the adapter. You can omit this value if you previously issued the SET SQLENGINE command.

nn

Is the number of seconds before a timeout occurs. 30 is the default value.

`0`

Represents an infinite period to wait for a response.

Cancelling Long Requests

You can cancel long running requests from the Reporting Server browser interface. Depending on the capabilities of the native JDBC driver, this action will either cancel the request entirely or break out of the fetch cycle.

Procedure: How to Cancel Long Requests

1. From the Reporting Server browser interface menu bar choose *Workspace*, *Java Services*, DEFAULT. Right-click on *DEFAULT* and select *Agents*.
2. In the Java Services Agents pane, highlight a row with the jscomid you wish to kill, right-click on it and select *Stop.*

Obtaining the Number of Rows Updated or Deleted

PASSRECS returns the number of rows affected by a successfully executed SQL Passthru INSERT, UPDATE, or DELETE command.

Tip: You can change this setting manually or from the Reporting Server browser interface by clicking *Get Data* on the menu bar, clicking a configured adapter, and choosing *Change Settings* from the menu. The Change Settings pane opens.

Syntax: How to Obtain the Number of Rows Updated or Deleted

```
ENGINE SQLJDBC SET PASSRECS {ON|OFF}
```

where:

SQLJDBC

Indicates the adapter. You can omit this value if you previously issued the SET SQLENGINE command.

ON

Provides the number of rows affected in the application program SCB count member after the successful execution of an SQL Passthru INSERT, UPDATE, or DELETE command. ON is the default value.

OFF

Provides no information after the successful execution of an SQL Passthru INSERT, UPDATE, or DELETE command.

JDBC Optimization Settings

Adapter optimization allows the RDBMS to perform the work for which it is best suited, reducing the volume of RDBMS-to-server communication and improving response time. It also enables the RDBMS to exploit its own internal optimization techniques.

Chapter 7

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.

In this chapter:

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 EnterpriseOne 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.

Configuring the Adapter for JD Edwards EnterpriseOne

This process defines connection and authentication information. An application folder, JOWSEC, is created during the configuration process. This is a working directory for the adapter.

Note: Do not modify the contents of the JOWSEC directory as they are required for the operation of the JD Edwards EnterpriseOne adapter. This includes creating new synonyms in this directory.

Overview of the Setup Process

The setup process for the adapter for JD Edwards EnterpriseOne is comprised 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 EnterpriseOne 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.

Procedure: How to Configure the Adapter

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. Always use the same administrator ID to work with adapter configuration.
2. Expand the top level folder.
3. Expand a reports folder.
4. Right-click a reports subfolder, and select *Metadata*. Then, select *New*.
5. In the left-hand Adapter navigation pane, expand the *Available* folder.
6. Expand the *ERP* folder.
7. Double-click *JD Edwards EnterpriseOne*.

8. Select the connection parameters.

   `Server Authentication`

   Check this box if the reporting server is secured. For Db2 Web Query, check this box. 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.

   `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).

   `UDC Direct File Access`

   When you select this check box, you give users access to the User Defined Codes, directly from the JD Edwards EnterpriseOne dictionary files.

   `Select Profile`

   You *must* choose edasprof.prf.

9. Click *Configure*.

   You will receive a confirmation message.

   Note: 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.

10. Click *OK*.

Refreshing the Security Extract

You will have the menu option to Refresh the Security Extract, only if GROUP or ROLE security was chosen during adapter configuration.

This process creates security extract files that are stored in the JOWSEC application that is created when you configure the adapter.

You must repeat this step whenever security extract information needs to be updated.

Procedure: How to Refresh the Security Extract

In order to refresh the security extract, use the menu option available on the JD Edwards EnterpriseOne adapter. You will need to perform this step anytime there are changes to the JD Edwards EnterpriseOne Security rules.

1. Right-click the configured JD Edwards EnterpriseOne adapter and select *Refresh Security Extract*.
2. Enter the Library Name of the library containing the specified objects.
3. Click *Submit* to refresh the security extract.

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 tables change. (At most sites, changes occur infrequently.)

Important: This step must be done before you convert any Master Files (synonyms).

Procedure: How to Refresh the Metadata Repository

From the Adapters list in the navigation pane on the Web Console or the Adapters tab in the Data Management Console:

1. Right-click the configured JD Edwards EnterpriseOne adapter and select *Refresh Metadata Repository*.

   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. For example, UDC and security changes in JDE tables would require a metadata refresh.

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

2. Enter the Library name of the library containing the specified objects. The UDC library can be any arbitrary name, for example, UDCLIB.

   The UDC library parameter is the library name that will contain the extracted information on the User Defined Codes from the JDE dictionary. If you did not check the UDC Direct File Access during adapter configuration, a new library with the name specified will be created on the system. Additionally, a new table will be created in that library which will contain UDC information to be used by DB2 Web Query.

3. Click *Refresh Now* to refresh the metadata repository.

Creating the JD Edwards EnterpriseOne Synonyms

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

Note: If a JDE synonym needs to be refreshed, the synonym will need to be re-created through the JD Edwards EnterpriseOne adapter. Otherwise the JDE dictionary information will not be added to the synonym.

Procedure: How to Create the JD Edwards EnterpriseOne Synonyms

To create the synonyms for reporting with JD Edwards EnterpriseOne:

1. Log on to Db2 Web Query.
2. Expand the top level folder, and then expand a domain.
3. Expand a reports folder.
4. Right-click a reports subfolder, and select *Metadata*. Then, select *New*.
5. Right-click *JD Edwards EnterpriseOne*, and select *Create Synonym*.
6. Click the DB2 CLI connection that points to your JD Edwards EnterpriseOne data tables.
7. Select the restrictions you would like to apply when searching for synonym candidates.

   Restriction options included are restrict object type, further restricting Tables, Views, and Aliases.

   Restrict Object Type to

   Restrict candidates for synonym creation based on the selected object type(s): Tables, Views, External SQL Scripts, and any other supported objects.

   Choosing *External SQL Scripts* from the drop-down list enables you to represent an SQL Query as a synonym for read-only reporting. A Synonym candidate can be any file that contains one (and only one) valid SQL Query and does not contain end-of-statement delimiters (";" or "/") and comments.

   Depending on the adapter, you can further restrict your search by choosing:

   Library

   Type a string for filtering the Library (or Db2 Collection), inserting the wildcard character (%) as needed at the beginning and/or end of the string. For example, enter: ABC% to select tables or views whose owner IDs begin with the letters ABC; %ABC to select tables or views whose owner IDs end with the letters ABC; %ABC% to select tables or views whose owner IDs contain the letters ABC at the beginning, middle, or end.

Object Name

Type a string for filtering the table, view, or object names, inserting the wildcard character (%) as needed at the beginning and/or end of the string. For example, enter: ABC% to select all tables, views, or objects whose names begin with the letters ABC; %ABC to select all whose names end with the letters ABC; %ABC% to select all whose names contain the letters ABC at the beginning, middle, or end.

Note: When you create a synonym for Db2 on the IBM i platform, standard IBM i naming conventions apply to the target data source. Therefore, the Adapter for Db2 supports the use of double-quotation marks around any library name and/or file name that contains lowercase or NLS characters.

8. Click *Next*.

The following list describes the parameters for which you will need to supply values, and related tasks you will need to complete in order to create a synonym for the adapter. These options may appear on multiple panes. To advance from pane to pane, click the buttons provided, ending with the *Create Synonym* button, which generates the synonym based on your entries.

Cardinality

Select the *Cardinality* check box to reflect the current cardinality (number of rows or tuples) in the table during metadata creation. Cardinality is used for equi-joins. The order of retrieval is based on the size (cardinality) of the table. Smaller tables are read first.

If the cardinality of the tables to be used in the application are dynamic, it may not be beneficial to choose this setting.

Build cluster using foreign keys (deprecated)

You can select the *Build cluster using foreign keys* check box to include within this synonym every table related to the current table by a foreign key. However, this option has been deprecated, as the recommended way to create a cluster is by using the Synonym Editor. The resulting multi-table synonym describes all of the foreign key relationships of this table.

Dynamic columns

To specify that the Master File created for the synonym should not contain column information, select the *Dynamic columns* check box.

If this option is selected, column data is retrieved dynamically from the data source at the time of the request.

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 run-time 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 as follows:

```
TABLENAME=MYTABLE
```

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

Application

The default value is baseapp.

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 the prefix and suffix fields blank.

Customize data type mappings

To change the data type mappings from their default settings, select this check box. The customizable mappings are displayed.

Overwrite Existing Synonyms

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.

9. Click *Create synonym*.

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.

Language Code

Enter the appropriate Language Code, which exists in the JDE F9292 file. (Leave the field blank for English.)

UDC

Select the *UDC* check box to ensure that UDC description fields are generated as DEFINEs (virtual fields) in the synonym. The default setting is *ON*.

Combine UDC

Select the *Combine UDC* check box to Combine User Defined Code. The default setting is *OFF*.

Floating Currency Symbol

Use floating or non-floating currency symbol for currency fields.

10. Click *Continue*.

The synonym has been successfully created.

Chapter 8

Using the Adapter for JD Edwards World

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

In this chapter:

Preparing the JD Edwards World Environment

Although no environment preparation steps are required, ensure that your system complies with all software specifications for JD Edwards World on IBM i.

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

Configuring the Adapter for JD Edwards World

This process defines connection and authentication information. An application folder, JDWSEC, is created during the configuration process. This is a working directory for the adapter.

Note: Do not modify the contents of the JDWSEC directory as they are required for the operation of the JD Edwards World adapter. Also do not create new synonyms in this directory.

Overview of the Set Up Process

The setup process is comprised of the following basic steps:

1. **Configure the Adapter for JD Edwards World.** Define what type of JD Edwards World 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 World metadata tables or security information. This occurs infrequently at most sites.
3. **Create the JD Edwards World synonyms.** Synonyms are required for Web Query reporting against a data source.

Procedure: How to Configure the Adapter for JD Edwards World

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. Always use the same administration ID to work with adapter configurations.
2. Expand the top level folder.
3. Expand a reports folder.
4. Right-click the reports subfolder, and select *Metadata*. Then, select *New*.
5. In the left-hand Adapter navigation pane, expand the *Available* folder.
6. Expand the *ERP* folder.
7. Expand the *JD Edwards World* folder.
8. Double-click *A7.x - A9.x*.
9. Select the connection parameters.

   `Business Unit Security`

   Check this box to enable automatic execution of JD Edwards World Business Unit Security. The Server for IBM i automatically restricts user access to data, based on information retrieved from the JDE dictionary tables, and then adds appropriate WHERE conditions to the user's submitted data access request. Unchecked (OFF) is the default setting. If you check this parameter, you cannot turn it OFF until the server is shut down and then restarted (with no parameter settings).

   `Search Type Security`

   Check this box to enable automatic execution of JD Edwards World Search Type Security. The adapter automatically restricts user access to data, based on information retrieved from the JDE dictionary table, and then adds appropriate WHERE conditions to the user's submitted data access request. Unchecked (OFF) is the default setting. If you check this parameter, you cannot turn it OFF until the server is shut down and then restarted (with no parameter settings).

`Business Unit (forPA) Security`

Check this box to revert (if necessary) to an older security model used by this adapter. Unchecked (OFF) is the default setting. If checked, this option overrides standard Business Unit Security (as described above).

`Column Security`

Check this box to enable column security based on information in the JDE dictionary file. Unchecked (OFF) is the default setting. If you check this parameter, you cannot turn it OFF until the server is shut down and then restarted (with no parameter settings).

`UDC Direct File Access`

- Check this box to enable User Defined Code Direct File Access. This will retrieve UDC information directly from the JDE dictionary.

  Note: It will create files, udcdicdb.*, in the JDWSEC application folder directory. Do not delete these files.

- Unchecked (OFF) is the default setting. This will create a UDC extract library which will considerably speed up UDC retrieval.

`Select Profile`

This must be EDASPROF.

10. Click *Configure*.

    You will receive a confirmation message.

11. Click *OK*.

Restarting the Reporting Server disconnects any users currently working in Db2 Web Query, please confirm no Web Query user jobs are running before clicking OK.

After the server restarts, the adapter for JD Edwards World is successfully added to the configuration.

Refreshing the Metadata Repository

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

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

Procedure: How to Refresh Metadata Repository

Important: In order to refresh metadata, you must have first configured the adapter.

1. Right-click *JD Edwards World*.
2. Click *Refresh Metadata Repository*.

   This is only done when you first configure the adapter for JD Edwards World, or when JD Edwards World data dictionary information changes. For example, UDC and security changes in JDE tables would require a metadata refresh.
3. Select the version of JD Edwards World you will be using. Enter the name of the library for each of the specified objects.

   Check the alternate language file if you want to access UDC description in another language.

   The UDC library parameter is the library name that will contain information on the User Defined Codes in the JDE dictionary. If you did not check the UDC Direct File Access during adapter configuration, a new library with the name specified will be created on the system. Additionally, a new table will be created in that library which will contain UDC information to be used by Db2 Web Query.
4. Click *Refresh Now*.

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

Creating the JD Edwards World Synonyms

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

Note: If a JDE synonym needs to be refreshed, the synonym will need to be re-created through the JD Edwards World adapter. Otherwise the JDE dictionary information will not be added to the synonym.

Procedure: How to Create the JD Edwards World Synonyms

To create the synonyms for reporting with JD Edwards World:

1. If not already at the page, access the adapter by performing steps 1 through 4 in *How to Configure the Adapter for JD Edwards World* on page 136.
2. Right-click *JD Edwards World*, and select *Create Synonym*.
3. Click the Db2 cli connection that points to your JD Edwards World data tables.
4. Select the restrictions you would like to apply when searching for synonym candidates.

   Restriction options included are restrict object type, further restricting Tables, Views, and Aliases.

5. Click *Next*.

   The following options will be displayed.

   Cardinality

   Select the *Cardinality* check box to reflect the current cardinality (number of rows or tuples) in the table during metadata creation. Cardinality is used for equi-joins. The order of retrieval is based on the size (cardinality) of the table. Smaller tables are read first.

   If the cardinality of the tables to be used in the application are dynamic, it may not be beneficial to choose this setting.

   Build cluster using foreign keys (deprecated)

   You can select the *Build cluster using foreign keys* check box to include within this synonym every table related to the current table by a foreign key. However, this option has been deprecated, as the recommended way to create a cluster is by using the Synonym Editor. The resulting multi-table synonym describes all of the foreign key relationships of this table.

   Dynamic columns

   To specify that the Master File created for the synonym should not contain column information, select the *Dynamic columns* check box.

   If this option is selected, column data is retrieved dynamically from the data source at the time of the request.

   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 run-time 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 as follows:

   ```
   TABLENAME=MYTABLE
   ```

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

Application

The default value is baseapp.

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 the prefix and suffix fields blank.

Customize data type mappings

To change the data type mappings from their default settings, select this check box. The customizable mappings are displayed.

Overwrite Existing Synonyms

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. Click *Create Synonym*.

The following options will show.

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.

Presumptive Join

Select the *Presumptive Joins* check box to include additional DEFINEs (virtual fields) for presumptive join fields in the synonym.

The default setting is *ON*.

Field Names

Select the*Long Fieldname* radio button (the default) to display the field descriptions as names on reports. Select the *Short Fieldname* radio button to use the JDE aliases as field names on reports. The default setting is *Long Fieldname*.

Language Code

Enter the appropriate Language Code, which exists in the JDE F9292 file. (Leave the field blank for English.)

UDC

Select the *UDC* check box to ensure that UDC description fields are generated as DEFINEs (virtual fields) in the synonym. The default setting is *ON*.

Combine UDC

Select the *Combine UDC* check box to Combine User Defined Code. The default setting is *OFF*.

Floating Currency Symbol

Use floating or non-floating currency symbol for currency fields.

7. Click *Continue* to create synonym and continue to add JD Edwards World dictionary information to your synonym.

Once you have created the synonyms, you can now develop Db2 Web Query reports to access JD Edwards World data.

Enabling JD Edwards World Security

You can also change security options after you configure the adapter from the Web Console.

Procedure: How to Change JD Edwards World Security

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. Expand the top level folder.
3. Expand a reports folder.
4. Right-click the reports subfolder, and select *Metadata*. Then, select *New*.
5. From the list of configured adapters in the navigation pane, right-click *JD Edwards World*, and select *Properties*.

   The Change Connect Parameters pane for JD Edwards World opens.

6. You can accept the defaults (unchecked) or select one or more options by clicking the corresponding check boxes.

Business Unit Security

Check this box to enable automatic execution of JD Edwards World Business Unit Security. The server for IBM i automatically restricts user access to data, based on information retrieved from the JDE dictionary tables, and then adds appropriate WHERE conditions to the users submitted data access request.

Unchecked (OFF) is the default setting.

If you check this parameter, you cannot turn it OFF until the server is shut down and then restarted (with no parameter settings).

Search Type Security

Check this box to enable automatic execution of JD Edwards World Search Type Security. The server for IBM i automatically restricts user access to data, based on information retrieved from the JDE dictionary table, and then adds appropriate WHERE conditions to the submitted data access request of the user.

Unchecked (OFF) is the default setting.

If you check this parameter, you cannot turn it OFF until the server is shut down and then restarted (with no parameter settings).

Business Unit (for PA) Security

Check this box to revert (if necessary) to an older security model used by this adapter.

Unchecked (OFF) is the default setting.

If checked, this option overrides standard Business Unit Security (as described above).

Column Security

Check this box to enable column security based on information in the JDE dictionary file.

Unchecked (OFF) is the default setting.

If you check this parameter, you cannot turn it OFF until the server is shut down and then restarted (with no parameter settings).

Select Profile

In Web Query this must be EDASPROF.

Chapter 9

Using the Db2 Web Query Adapter for Microsoft SQL Server

The Db2 Web Query Adapter for Microsoft® SQL Server® allows applications to access Microsoft SQL Server data sources. The adapter converts data or application requests into native Microsoft SQL Server statements and returns optimized answer sets to the requesting program.

In this chapter:

Preparing the Microsoft SQL Server Environment

In order to take full advantage of the features available through the Db2 Web Query Adapter for Microsoft SQL Server 2012/2014, we strongly recommend using the same or higher version of the Microsoft SQL Server Client. To determine which version of MS SQL Server you are using, please refer to the following article: *http://support.microsoft.com/kb/321185*.

Procedure: How to Set Up the Environment on IBM i

Identify the location of the Microsoft SQL Server JDBC Driver files using the environment variable $CLASSPATH. For example, to set the location of the JDBC Driver files, specify:

```
CLASSPATH=/qas/mss/sqljdbc_4.0/enu/sqljdbc4.jar
export CLASSPATH
```

The driver file sqljdbc4.jar file is within the "Microsoft JDBC Driver 4.0 for SQL Server" downloadable *.tar.gz version of the Microsoft package. Optionally, the CLASSPATH or IBI_CLASSPATH may be set up as part of the adapter configure step or in Java Services property (and exporting skipped, but the directory location must be known).

JVM access is built-in on IBM i and no further environment set up is needed. Java version 1.6 or higher is required.

Configuring the Db2 Web Query Adapter for Microsoft SQL Server

Before you configure the adapter, be sure to install the Microsoft SQL Server 2000 or 2005 JDBC Driver to the Java Extensions directory: /QIBM/UserData/Java400/ext.

Configuring the adapter consists of specifying connection and authentication information for each of the connections you want to establish.

Declaring Connection Attributes

In order to connect to an Microsoft SQL Server database server, the adapter requires connection and authentication information. You supply this information using the Web Query Metadata option. The connection and authentication information is added to the profile you select: the global server profile (edasprof.prf), a user profile (*user*.prf), or a group profile (if supported on your platform).

You can:

You can declare connections to more than one Microsoft SQL Server databases by including multiple CONNECTION_ATTRIBUTES commands. The actual connection to the Microsoft SQL Server takes place when the first query that references the connection is issued. If you issue multiple CONNECTION_ATTRIBUTES commands:

Reference: Connection Attributes for Microsoft SQL Server

The *MS SQL Server* adapter is under the *SQL* group folder.

The following list describes the connection attributes for which you can supply values. To complete the attribute declaration, click the *Configure* button.

Connection name

Logical name used to identify this particular set of connection attributes. The default is CON01.

Server (Windows only)

Name of the machine where Microsoft SQL Server is running. If that machine has more than one instance of Microsoft SQL Server installed, provide the server name and the instance name as follows: server\instance.

The server connection attribute will allow users to choose from the list of MS SQL Servers visible within the local network using the SQL Native Client Enumerator.

Please note that due to limitations of the SQL Native Client Enumerator, local network settings and MS SQL Server settings, it is possible that not all operational servers will be visible. If the name of the server you wish to target does not appear, you can enter it manually.

URL (UNIX, IBM i, and z/OS only)

Enter the location URL for the Microsoft SQL Server data source.

To use the adapter with TLS, you need to add the encrypt and trustServerCertificate properties, as follows:

```
jdbc:sqlserver://host:port;encrypt=true;trustServerCertificate=true
```

Note: The URL properties can vary, depending on how the SQL server is configured or the release level of the JDBC driver. For more information, refer to *https://learn.microsoft.com/en-us/sql/connect/jdbc/setting-the-connection-properties?view=sql-server-ver16#properties*.

Security

There are three methods by which a user can be authenticated when connecting to a Microsoft SQL Server:

- **Explicit.** The user ID and password are explicitly specified for each connection and passed to Microsoft SQL Server, at connection time, for authentication as a standard login.

  This option requires that SQL Server security be set to SQL Server and Windows (for Windows), or else to SQL Server and UNIX.

- **Password Passthru.** (*Windows only*) The user ID and password received from the client application are passed to Microsoft SQL Server, at connection time, for authentication as a standard login.

  This option requires that SQL Server security be set to: SQL Server and Windows.

- **Trusted.** The adapter connects to Microsoft SQL Server as an operating system login using the credentials of the operating system user impersonated by the server data access agent.

  This option works with either of the SQL Server security settings.

User

Primary authorization ID by which you are known to the data source.

Password

Password associated with the primary authorization ID.

Default Database (Windows only)

Name of the default database for the connection. This value is used when a data object is not qualified with the database name.

This parameter is optional. If not specified, it defaults to the database associated with the authorization ID.

Driver name (UNIX, IBM i, and z/OS only)

Name for the Microsoft JDBC driver.

Select profile

Select a profile from the drop-down menu to indicate the level of profile in which to store the CONNECTION_ATTRIBUTES command. The global profile, edasprof.prf, is the default.

If you wish to create a new profile, either a user profile (*user*.prf) or a group profile if available on your platform (using the appropriate naming convention), choose *New Profile* from the drop-down menu and enter a name in the Profile Name field (the extension is added automatically).

Store the connection attributes in the server profile (edasprof).

Controlling the Connection Scope

The AUTODISCONNECT command controls the persistence of connections when using the adapter for each of the connections you want to establish.

Syntax: How to Control the Connection Scope

You change this setting by expanding the Domains and Reports folders, then right-clicking a subfolder and choosing *Metadata*. Right-click MS SQL Server 200*x* and select *Change Settings*. The Change Settings pane opens.

`SQLMSS`

Indicates the adapter. You can omit this value if you previously issued the SQLENGINE command.

`FIN`

Disconnects automatically only after the session has been terminated. FIN is the default value.

COMMAND

Disconnects automatically after each request. Depending on how often the event occurs, the AUTODISCONNECT command may result in considerable overhead. Almost all of this overhead is not related to the server. It is related to the operating system and the data source.

Managing Microsoft SQL Server Metadata

When the server accesses a data source, it needs to know how to interpret the data stored there. For each data source the server will access, you create a synonym that describes the structure of the data source and the server mapping of the Microsoft SQL Server data types.

Creating Synonyms

Synonyms define unique names (or aliases) for each Microsoft SQL Server table or view that is accessible from the server. Synonyms are useful because they hide the underlying data source location and identity from client applications. They also provide support for extended metadata features of the server, such as virtual fields and additional security mechanisms.

Using synonyms allows an object to be moved or renamed while allowing client applications to continue functioning without modification. The only modification required is a redefinition of the synonym on the server. The result of creating a synonym is a Master File and an Access File, which represent the server metadata.

Note that creating a synonym for a stored procedure is described with reporting against a stored procedure, in *Generating a Synonym for a Stored Procedure* on page 157.

Procedure: How to Create a Synonym

1. From the Reporting Server browser interface Application page, click *Get Data*.
2. On the Configured Adapters section of the page, in Simple Mode, right-click an adapter and click *Show Connections*. Right-click a connection.

   Depending on the type of adapter you choose, one of the following options appears on the context menu.

   - **Show DBMS objects.** This option opens the page for selecting synonym objects and properties.
   - **Create metadata objects.** This option opens the page for selecting synonym objects and properties.
   - **Show files.** This option opens a file picker. After you choose a file of the correct type, the page for selecting synonym objects and properties opens.

- **Show local files.** This option opens a file picker. After you choose a file of the correct type, the page for selecting synonym objects and properties opens.
- **Show topics.** This option opens the page for selecting synonym objects and properties for topics within the environment.

3. Enter values for the parameters required by the adapter as described in the chapter for your adapter.
4. After entering the parameter values, click *Add*.

   This button may be labeled *Next*, *Create Synonym*, *Create Base Synonyms*, *Create Cluster Synonym*, or *Update Base Synonyms*.

   The synonym creation process for most adapters has been consolidated so that you can enter all necessary parameters on one page. However, for some adapters such as LDAP, continue clicking *Next* until you get to a page that has a *Create Synonym* button.

The synonym is created and added under the specified application directory.

Note: When creating a synonym, if you select the Validate check box, where available, the server adjusts special characters and checks for reserved words.

Procedure: How to Create a Synonym From the Web Query Environment

1. Expand the *Domains* folder, then expand a domain.
2. Expand the *Reports* folder, right-click a request subfolder, and choose *Metadata* from the menu.
3. In the left-hand Adapter navigation pane, click the database connection (MS SQL Server 200*x*) and choose *Create Synonym* from the menu. The first of a series of synonym creation pages opens.
4. Enter values for the parameters required by the adapter.

   For information about these parameters, see *Synonym Creation Parameters for Microsoft SQL Server* on page 149.
5. After entering parameter values, click *Create Synonym*.

   Synonyms are created and added under the specified application directory.

   The Status pane indicates that the synonyms were created successfully.
6. You can click *Go to Metadata page* where you can manage synonyms from the navigation pane.

Reference: Synonym Creation Parameters for Microsoft SQL Server

The following list describes the synonym creation parameters for which you can supply values.

Restrict Object Type to

Restrict candidates for synonym creation based on the selected object type(s): Tables, Views, External SQL Scripts, and any other supported objects.

Choosing *External SQL Scripts* from the drop-down list enables you to represent an SQL Query as a synonym for read-only reporting. A Synonym candidate can be any file that contains one (and only one) valid SQL Query and does not contain end-of-statement delimiters (";" or "/") and comments.

Depending on the adapter, you can further restrict your search by choosing

Important: If you select Stored Procedures as your object type, the input parameters will be a little different from those described here. For details, refer to *Creating a Report Against a Stored Procedure* on page 161.

Database selection

To specify a database from which you can select a table or other object, do one of the following:

- Check Use current database to use the database that has been set as the default database.
- Select a database from the Select database drop-down list, which lists all databases in the current DBMS instance.

  Before selecting a database, if Use current database is checked, uncheck it.

To specify the intended database, choose from the Select database drop-down menu, which shows all databases on the targeted instance of Microsoft SQL Server. Selecting Default Database will retain the database set during connection configuration. If Default Database was not set during configuration, the database assigned to the active login on the SQL Server will be used as the default.

Filter by Owner/Schema and Object name

Selecting this option adds the Owner/Schema and Object Name parameters to the screen.

- **Owner/Schema.** Type a string for filtering the selection, inserting the wildcard character (%) as needed at the beginning and/or end of the string. For example, enter: ABC% to select tables or views whose owner/schema begin with the letters ABC; %ABC to select tables or views whose owner/schema end with the letters ABC; %ABC% to select tables or views whose owner/schema contain the letters ABC at the beginning, middle, or end.

- **Object name.** Type a string for filtering the object names, inserting the wildcard character (%) as needed at the beginning and/or end of the string. For example, enter: ABC% to select all objects whose names begin with the letters ABC; %ABC to select all whose names end with the letters ABC; %ABC% to select all whose names contain the letters ABC at the beginning, middle, or end.

Location of External SQL Scripts

If you specify *External SQL Scripts* in the *Restrict Object type to* field, these additional fields are displayed.

The following standard naming conventions apply for UNIX, IBM i IFS, and z/OS HFS:

- In the *Base Location* field, specify the physical directory location of the file that contains the SQL Query. You can type a directory name or click on the ellipsis. This opens the Select Base Location dialog box.

- In the *Document Name* field, enter the file name with or without wild card characters.

- In the *Document Extension* field, enter the extension of the script files to filter the list of candidates.

On IBM i, you can use alternative IFS naming conventions to access library members. The following entry illustrates this method:

- In the *Base Location* field, enter:

  `/QSYS.LIB/MYLIBRARY.LIB/MYSRC.FILE`

- The *Document Extension* is understood to be MBR. You can enter this value explicitly or leave the input box blank.

During synonym generation, the adapter issues native API calls to obtain a list of elements in the select list and builds the Master File with a field for each element. The generated Access File references the location of the SQL script in the DATASET attribute, which contains the full path, including the name and extension of the file containing the SQL Query. For example,

```
DATASET=/ul/home2/apps/report3.sql
```

When a Web Query report is created, the SQL Query is used to access data.

Cardinality

Select the *Cardinality* check box to reflect the current cardinality (number of rows or tuples) in the table during metadata creation. Cardinality is used for equi-joins. The order of retrieval is based on the size (cardinality) of the table. Smaller tables are read first.

If the cardinality of the tables to be used in the application are dynamic, it may not be beneficial to choose this setting.

Build cluster using foreign keys (deprecated)

You can select the *Build cluster using foreign keys* check box to include within this synonym every table related to the current table by a foreign key. However, this option has been deprecated, as the recommended way to create a cluster is by using the Synonym Editor. The resulting multi-table synonym describes all of the foreign key relationships of this table.

For Subquery

Only available when *External SQL Scripts* is selected from the Restrict objects type to drop-down menu. When selected, a SUBQUERY keyword is added to the Access File of the generated synonym. If the corresponding SQL string has valid syntax that can be used in the FROM statement of the generated SQL (what is known as a Derived Table), then the SQL SCRIPT will be processed as a subquery embedded into a FROM clause. This usage allows for more flexibility. For example, the synonym can be used as a target for a JOIN.

If the SQL SCRIPT has parameter markers, such as ? or :, or the syntax contains constructs that are invalid for a derived table, for example ORDER BY, then this keyword should not be selected. At runtime, if SUBQUERY=Y is present and it is determined that the SQL SCRIPT cannot be used in the FROM statement, the setting will be ignored, and a FOC1782 warning message will be issued. The default is selected (SUBQUERY=Y).

Application

The default value is baseapp.

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 the prefix and suffix fields blank.

Customize data type mappings

To change the data type mappings from their default settings, select this check box. The customizable mappings are displayed.

For information about customizable mappings, see *Microsoft SQL Server Data Type Support* on page 156.

Update or Create Metadata

Select *Create* to overwrite any existing synonym with the same fully-qualified name, or *Update* to synchronize the metadata with an existing synonym. If you select *Update*, the next screen will show a list of attributes from the DBMS catalog that you can check to allow attributes from the DBMS catalog to override attributes from the existing synonym.

Overwrite Existing Synonyms

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.

Default Synonym Name

This column displays the name that will be assigned to each synonym. To assign a different name, replace the displayed value.

Table name

Is the name of the underlying object.

Select tables

Select tables for which you wish to create synonyms:

- To select all tables in the list, select the *Select All* check box.
- To select specific tables, select the corresponding check boxes.

Example: **Sample Generated Synonym**

DB2 Web Query Adapter for Microsoft SQL Server synonym comprises a Master File and an Access File. This is a synonym for the table nf29004.

Generated Master File nf29004.mas

```
FILE=DIVISION, SUFFIX=SQLMSS ,$
SEGNAME=SEG1_4, SEGTYPE=S0 ,$
FIELD=DIVISION4,    DIVISION4,    I9,  I4,  MISSING=OFF ,$
FIELD=DIVISION_NA4, DIVISION_NA4, A25, A25, MISSING=ON  ,$
FIELD=DIVISION_HE4, DIVISION_HE4, I9,  I4,  MISSING=ON  ,$
```

Generated Access File nf29004.acx

```
SEGNAME=SEG1_4,TABLENAME=edaqa.nf29004,
CONNECTION=connmss, KEYS=1, WRITE=YES,$
```

Reference: **Mapping Microsoft SQL Table Comments Into a Synonym**

When you generate a synonym for a Microsoft SQL table or view, the adapter maps comments as follows:

- MS SQL Server table/view comments (if present) are mapped to the REMARKS attribute in the Master File synonym.
- MS SQL Server column comments (if present) are mapped to the DESCRIPTION attribute in the Master File synonym.

Both Unicode and non-Unicode comments are supported.

Also, MS SQL Server column title (if present) is mapped to the TITLE attribute in the Master File synonym.

Reference: **Access File Keywords**

This chart describes the keywords in the Access File.

Keyword	Description
SEGNAME	Value must be identical to the SEGNAME value in the Master File.
TABLENAME	Identifies the Microsoft SQL Server table. The table name can be fully qualified as follows: TABLENAME=[[*database*.]*owner*.]*table*

Keyword	Description
CONNECTION	Indicates a previously declared connection. The syntax is: CONNECTION=*connection* CONNECTION=' ' indicates access to the local database server. Absence of the CONNECTION attribute indicates access to the default database server.
KEYS	Indicates how many columns constitute the primary key for the table. Corresponds to the first *n* fields in the Master File segment. This attribute requires the columns that constitute the key to be described first in the Master File. See the KEY attribute below for information about specifying the key fields without having to describe them first in the Master File.
KEY	Specifies the columns that participate in the primary key without having to describe them first in the Master File. The syntax is: KEY=*fld1/fld2/.../fldn*
WRITE	Specifies whether write operations are allowed against the table.

Keyword	Description
KEYFLD IXFLD	Supply the names of the primary key and foreign key fields that implement the relationships established by the multi-table Master File. Together, KEYFLD and IXFLD identify the field shared by a related table pair. ❏ KEYFLD is the FIELDNAME of the common column from the parent table. ❏ IXFLD is the FIELDNAME of the common column from the related table. KEYFLD and IXFLD must have the same data type. It is recommended, but not required, that their lengths also be the same. **Note:** An RDBMS index on both the KEYFLD and IXFLD columns provides the RDBMS with a greater opportunity to produce efficient joins. The columns must have the same data type. If their length is the same, the RDBMS handles the join more efficiently.
AUTO INCREMENT	Set to *Yes* to allow the auto increment feature.
START	Initial value in incrementing sequence.
INCREMENT	Increment interval.
INDEX_NAME INDEX_UNIQUE INDEX_COLUMNS INDEX_ORDER	Indicate a name of the index in a database, uniqueness, name, and order of the indexed column(s).

Reference: **Managing Synonyms**

Once you have created a synonym, you can right-click the synonym name in the navigation pane to access the available options.

Microsoft SQL Server Data Type Support

SQL Data Type mapping options are available in a report available from the Reporting Server browser interface.

Enabling National Language Support

The parameter NCHAR indicates whether the character set is single-byte, double-byte, or triple-byte. The NCHAR setting affects the mapping of NCHAR and NVARCHAR data types.

The following chart lists data type mappings based on the value of NCHAR.

Microsoft SQL Server Data Type	Remarks	NCHAR SBCS		NCHAR DBCS		NCHAR TBCS	
NCHAR (*n*)	*n* is an integer between 1 and 4000 *d* = 2 * *n* *t* = 3 * *n*	A*n*	A*n*	A*d*	A*d*	A*t*	A*t*
NVARCHAR (*n*)	*n* is an integer between 1 and 4000 *d* = 2 * *n* *t* = 3 * *n*	A*n*	A*n*	A*d*	A*d*	A*t*	A*t*

Syntax: How to Enable National Language Support

You change this setting by expanding the Domains and Reports folders, then right-clicking a subfolder and choosing *Metadata*. Right-click MS SQL Server 200*x* and select *Change Settings*. The Change Settings pane opens.

SQLMSS

Indicates the adapter. You can omit this value if you previously issued the SQLENGINE command.

SBCS

Indicates a single-byte character set. SBCS is the default value.

DBCS

Indicates a double-byte character set.

TBCS

Indicates a triple-byte character set.

Support of Read-Only Fields

CREATE SYNONYM creates a field description with FIELDTYPE=R for Microsoft SQL Server columns created as TIMESTAMP or columns with the IDENTITY attribute. These fields are read-only.

Reporting Against a Microsoft SQL Server Stored Procedure

You can use a reporting tool, such as a SELECT statement or TABLE command, to execute Microsoft SQL Server stored procedures and report against the procedure output parameters and answer set. Among the benefits of this method of executing a stored procedure are:

- The retrieval of output parameters: OUT parameters, and INOUT parameters in OUT mode, as well as the answer set. (Other methods of invocation retrieve the answer set only.)
- The ease with which you can process, format, and display output parameters and the answer set, using TABLE and other reporting tools.

To report against a stored procedure:

1. **Generate a synonym** for the stored procedure answer set, as described in *Generating a Synonym for a Stored Procedure* on page 157.
2. **Create a report procedure,** as described in *Creating a Report Against a Stored Procedure* on page 161.
3. **Run the report.** This executes the stored procedure and reports against any output parameters (OUT and INOUT in OUT mode), and any answer set fields, specified in the report.

Generating a Synonym for a Stored Procedure

A synonym describes the stored procedure parameters and answer set.

An answer set structure may vary depending on the input parameter values that are provided when the procedure is executed. Therefore, you need to generate a separate synonym for each set of input parameter values that will be provided when the procedure is executed at run time. For example, if users may execute the stored procedure using three different sets of input parameter values, you need to generate three synonyms, one for each set of values. (Unless noted otherwise, *input parameters* refers to IN parameters and to INOUT parameters in IN mode.)

There is an exception. If you know the internal logic of the procedure, and are certain which range of input parameter values will generate each answer set structure returned by the procedure, you can create one synonym for each answer set structure, and for each synonym simply provide a representative set of the input parameter values necessary to return that answer set structure.

A synonym includes the following segments:

- INPUT, which describes any IN parameters and INOUT parameters in IN mode.

  If there are no IN parameters or INOUT parameters in IN mode, the segment describes a single dummy field.

- OUTPUT, which describes any OUT parameters and INOUT parameters in OUT mode.

  If there are no OUT parameters or INOUT parameters in OUT mode, the segment is omitted.

- ANSWERSET*n*, one for each answer set.

  If there is no answer set, the segment is omitted.

Example: Synonym for Microsoft SQL Server Stored Procedure CustOrders

The following synonym describes a Microsoft SQL Server stored procedure with one input parameter, one output parameter, and one answer set containing four variables.

The Master File synonym is:

```
FILENAME=CUSTORDERS, SUFFIX=SQLMSS  , $
   SEGMENT=INPUT, SEGTYPE=S0, $
      FIELDNAME=@CUSTOMERID, ALIAS=P0001, USAGE=A5, ACTUAL=A5,
         MISSING=ON, ACCESS_PROPERTY=(NEED_VALUE), $
   SEGMENT=OUTPUT, SEGTYPE=S0, PARENT=INPUT, $
      FIELDNAME=@RETURN_VALUE, ALIAS=P0000, USAGE=I11, ACTUAL=I4, $
   SEGMENT=ANSWERSET1, SEGTYPE=S0, PARENT=INPUT, $
      FIELDNAME=ORDERID, ALIAS=OrderID, USAGE=I11, ACTUAL=I4, $
      FIELDNAME=ORDERDATE, ALIAS=OrderDate, USAGE=HYYMDs, ACTUAL=HYYMDs,
         MISSING=ON, $
      FIELDNAME=REQUIREDDATE, ALIAS=RequiredDate, USAGE=HYYMDs,
         ACTUAL=HYYMDs, MISSING=ON, $
      FIELDNAME=SHIPPEDDATE, ALIAS=ShippedDate, USAGE=HYYMDs,
         ACTUAL=HYYMDs, MISSING=ON, $
```

The Access File synonym is:

```
SEGNAME=INPUT, CONNECTION=ITarget, STPNAME=Northwind.dbo.CustOrders, $
SEGNAME=OUTPUT, STPRESORDER=0, $
SEGNAME=ANSWERSET1, STPRESORDER=1, $
```

Reference: Synonym Creation Parameters for Stored Procedures

The following list describes the synonym creation parameters for which you can supply values.

Restrict Object Type to

Select *Stored Procedures*.

Database selection

To specify a database from which you can select a table or other object, do one of the following:

- Check Use current database to use the database that has been set as the default database.
- Select a database from the Select database drop-down list, which lists all databases in the current DBMS instance.

Before selecting a database, if Use current database is checked, uncheck it.

Filter by Owner/Schema and Object name

Selecting this option adds the Owner/Schema and Object Name parameters to the screen.

- **Owner/Schema.** Type a string for filtering the selection, inserting the wildcard character (%) as needed at the beginning and/or end of the string. For example, enter: ABC% to select tables or views whose owner/schema begin with the letters ABC; %ABC to select tables or views whose owner/schema end with the letters ABC; %ABC% to select tables or views whose owner/schema contain the letters ABC at the beginning, middle, or end.
- **Object name.** Type a string for filtering the object names, inserting the wildcard character (%) as needed at the beginning and/or end of the string. For example, enter: ABC% to select all objects whose names begin with the letters ABC; %ABC to select all whose names end with the letters ABC; %ABC% to select all whose names contain the letters ABC at the beginning, middle, or end.

Note: For Db2, this applies to all platforms except IBM i.

Select

Select a procedure. You may only select one procedure at a time since each procedure will require unique input in the Values box on the next synonym creation pane.

Name

The name of the synonym, which defaults to the stored procedure name.

Application

The default value is baseapp.

Prefix/Suffix

If you have stored procedures with identical names, assign a prefix or a suffix to distinguish their corresponding synonyms. Note that the resulting synonym name cannot exceed 64 characters.

If all procedures have unique names, leave the prefix and suffix fields blank.

Overwrite Existing Synonyms

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.

Customize data type mappings

To change the data type mappings from their default settings, select this check box. The customizable mappings are displayed.

For information about customizable mappings, see *Microsoft SQL Server Data Type Support* on page 156.

Values

Select the check box for *every* parameter displayed for the specified procedure.

Note the following before you enter parameter values: if the procedure you selected has input parameters (IN parameters and/or INOUT parameters in IN mode), you will be prompted to enter values for them. However, the need for an explicit Value entry depends on the logic of the procedure and the data structures it produces. Therefore, while you must check the parameter box, you may not need to enter a value. Follow these guidelines:

- Explicit input values (and separate synonyms) are required when input parameter values cause answer sets with different data structures, which vary depending on the input parameters provided.
- Explicit input values are not required when you know the procedure's internal logic and are certain that it always produces the same data structure. In this situation, only one synonym needs to be created and you can leave the Value input blank for synonym creation purposes.

If a Value is required, enter it without quotes. Any date, date-time, and timestamp parameters must have values entered in an ISO format. Specify the same input parameters that will be provided when the procedure is executed at run time if it is a procedure that requires explicit values.

Creating a Report Against a Stored Procedure

You can report against a stored procedure answer set using the same facilities you use to report against a database table:

- **SQL SELECT statement.** For syntax, see *How to Report Against a Stored Procedure Using SELECT* on page 162.
- **TABLE and GRAPH commands.** For syntax, see *How to Report Against a Stored Procedure Using the TABLE Command* on page 161.

When joining from or to a stored procedure answer set, you can:

- **Join from** only OUTPUT and ANSWERSET segments in a host file.
- **Join to** only INPUT segments in a cross-referenced file.

Syntax: How to Report Against a Stored Procedure Using the TABLE Command

To execute a stored procedure using the TABLE command, use the following syntax

```
TABLE FILE synonym
PRINT [parameter [parameter] ... | *]
[IF in-parameter EQ value]
  .
  .
  .
END
```

where:

synonym

Is the synonym of the stored procedure you want to execute.

parameter

Is the name of a parameter whose values you want to display in the report. You can specify input parameters, output parameters, or input and output parameters.

If the stored procedure does not require parameters, specify an asterisk (*). This displays a dummy segment, created when the synonym is generated, to satisfy the structure of the SELECT statement.

*

Indicates that you want to display all indicated parameters, or that there are no required parameters.

IF

Is an IF or WHERE keyword. Use this to pass a value to an IN parameter or an INOUT parameter in IN mode.

in-parameter

Is the name of an IN parameter, or INOUT parameter in IN mode, to which you want to pass a value.

Note: The length of in-parameters cannot exceed 1000 characters if the adapter is configured for Unicode support.

value

Is the value you are passing to a parameter.

Syntax: How to Report Against a Stored Procedure Using SELECT

```
SQL
SELECT [parameter [,parameter] ... | *] FROM synonym
[WHERE in-parameter = value]
  .
  .
  .
END
```

where:

synonym

Is the synonym of the stored procedure that you want to execute.

parameter

Is the name of a parameter whose values you want to display in the report. You can specify input parameters, output parameters, or input and output parameters.

If the stored procedure does not require parameters, enter an asterisk (*) in the syntax. This displays a dummy segment, created during synonym generation, to satisfy the structure of the SELECT statement.

*

Indicates that you want to display all indicated parameters, or that there are no required parameters.

`WHERE`

Is used to pass a value to an IN parameter or an INOUT parameter in IN mode.

You must specify the value of each parameter on a separate line.

`in-parameter`

Is the name of an IN parameter, or INOUT parameter in IN mode, to which you want to pass a value.

`value`

Is the value you are passing to a parameter.

Customizing the Microsoft SQL Server Environment

The Db2 Web Query Adapter for Microsoft SQL Server provides several parameters for customizing the environment and optimizing performance. This topic provides an overview of customization options.

Specifying the Cursor Type

You can use the CURSORS command to specify the type of cursors for retrieval.

Syntax: How to Specify the Cursor Type

You change this setting by expanding the Domains and Reports folders, then right-clicking a subfolder and choosing *Metadata*. Right-click MS SQL Server 200*x* and select *Change Settings*. The Change Settings pane opens.

`CLIENT`

Uses Microsoft SQL Server client-side cursors for retrieving data. Client-side cursors normally demonstrate the best performance for data retrieval and benefit the Microsoft SQL Server process. However, except in TRANSACTIONS AUTOCOMMITTED mode, using client-side cursors prevents a server agent from simultaneously reading more than one answer set from the same instance of Microsoft SQL Server.

`SERVER`

Uses Microsoft SQL Server server-side cursors for retrieving data. Server-side cursors demonstrate lower performance than client cursors. However, setting a high FETCHSIZE factor (100 is the adapter default) improves performance dramatically making them almost as fast as client-side cursors. Client-side cursors are recommended wherever possible to take the load off the Microsoft SQL Server process.

`blank`

Uses client-side cursors in TRANSACTIONS AUTOCOMMITTED mode and server-side cursors otherwise. This value is the default.

Activating NONBLOCK Mode

The Adapter for Microsoft SQL Server has the ability to issue calls in NONBLOCK mode. The default behavior is BLOCK mode.

This feature allows the adapter to react to a client request to cancel a query while the adapter is waiting on engine processing. This wait state usually occurs during SQL parsing, before the first row of an answer set is ready for delivery to the adapter or while waiting for access to an object that has been locked by another application.

Syntax: How to Activate NONBLOCK Mode

You change this setting by expanding the Domains and Reports folders, then right-clicking a subfolder and choosing *Metadata*. Right-click MS SQL Server 200*x* and select *Change Settings*. The Change Settings pane opens.

`SQLMSS`

Indicates the adapter. You can omit this value if you previously issued the SQLENGINE command.

n

Is a positive numeric number. 0 is the default value, which means that the adapter will operate in BLOCK mode. A value of 1 or greater activates the NONBLOCK calling and specifies the time, in seconds, that the adapter will wait between each time it checks to see if the:

- Query has been executed.
- Client application has requested the cancellation of a query.
- Kill Session button on the Reporting Server browser interface is pressed.

Note: A value of 1 or 2 should be sufficient for normal operations.

Obtaining the Number of Rows Updated or Deleted

PASSRECS returns the number of rows affected by a successfully executed SQL Passthru INSERT, UPDATE, or DELETE command.

Syntax: How to Obtain the Number of Rows Updated or Deleted

You change this setting by expanding the Domains and Reports folders, then right-clicking a subfolder and choosing *Metadata*. Right-click MS SQL Server 200*x* and select *Change Settings*. The Change Settings pane opens.

SQLMSS

Indicates the adapter. You can omit this value if you previously issued the SQLENGINE command.

ON

Provides the number of rows affected in the application program SCB count member after the successful execution of an SQL Passthru INSERT, UPDATE, or DELETE command. ON is the default value.

OFF

Provides no information after the successful execution of an SQL Passthru INSERT, UPDATE, or DELETE command.

Controlling Transactions

The TRANSACTIONS command to controls how the adapter handles transactions.

Syntax: How to Control Transactions

You change this setting by expanding the Domains and Reports folders, then right-clicking a subfolder and choosing *Metadata*. Right-click MS SQL Server 200*x* and select *Change Settings*. The Change Settings pane opens.

SQLMSS

Indicates the adapter. You can omit this value if you previously issued the SQLENGINE command.

LOCAL

Indicates that the adapter implicitly starts a local transaction on each of the connections where any work is performed. At the time of COMMIT or ROLLBACK, or at the end of the server session, the adapter commits or aborts the work on each connection consecutively. LOCAL is the default value.

DISTRIBUTED

Indicates that the adapter implicitly invokes Microsoft Distributed Transactions Coordinator (DTC) to create a single distributed transaction within which to perform all work on all the connections. At the time of COMMIT or ROLLBACK, or at the end of the server session, the adapter invokes DTC to execute the two-phase commit or rollback protocol. For this purpose, the DTC service must be started on the machine where the server is running and also on all the machines where involved instances of Microsoft SQL Server reside.

This mode is recommended for read-write applications that perform updates on multiple connections simultaneously.

AUTOCOMMITTED

Indicates that each individual operation with Microsoft SQL Server is immediately committed (if successful) or rolled back (in case of errors) by the SQL Server. This is recommended for read-only applications for performance considerations. It is not recommended for read-write applications because in this mode it is impossible to roll back a logical unit of work that consists of several operations.

Specifying the Transaction Isolation Level

You can specify the transaction isolation level from the Reporting Server browser interface.

Syntax: How to Specify Transaction Isolation Level

You change this setting by expanding the Domains and Reports folders, then right-clicking a subfolder and choosing *Metadata*. Right-click MS SQL Server 200*x* and select *Change Settings*. The Change Settings pane opens.

The available parameters are:

RU

Sets the transaction isolation level to Read Uncommitted.

RC

Sets the transaction isolation level to Read Committed.

RR

Sets the transaction isolation level to Repeatable Read.

SE

Sets the transaction isolation level to Serializable Read.

`CH`

Sets the transaction isolation level to Chaos.

`CS`

Sets the transaction isolation level to Cursor Stability, which is a synonym for Read Committed.

Microsoft SQL Server Optimization Settings

Adapter optimization allows the RDBMS to perform the work for which it is best suited, reducing the volume of RDBMS-to-server communication and improving response time. It also enables the RDBMS to exploit its own internal optimization techniques.

Optimizing Requests if a Virtual Field Contains Null Values

The OPTNOAGGR command provides finely-tuned control of adapter behavior for optimization. Users who for any reason wish to prevent passing aggregation to the RDBMS can use this command. An example of such a reason might be where NULL values occur in aggregated data with calculations. The OPTNOAGGR command causes the adapter to generate SQL without passing aggregation to the DBMS. Aggregation is instead performed internally by the server while JOIN and SORT operations are handled by the RDBMS.

If any DEFINE field contains calculations with NULL fields then such operations cannot be translated to SQL and pass to DBMS because always return NULL. It has to be processed by Web Query.

This can be achieved by setting OPTIMIZATION to OFF.

However, in some cases it is preferable to use the off-load JOIN and SORT operation to DBMS for better performance while leaving AGGREGATION to Web Query.

Reference: SQL Limitations on Optimization of DEFINE Expressions

Since the Web Query reporting language is more extensive than native SQL, the data adapter cannot pass certain DEFINE expressions to the RDBMS for processing. The data adapter does not offload DEFINE-based aggregation and record selection if the DEFINE includes:

- User-written subroutines.
- Self-referential expressions, such as:

  ```
  X=X+1;
  ```

- EDIT functions for numeric-to-alpha or alpha-to-numeric field conversions.

- DECODE functions for field value conversions.
- Relational operators INCLUDES and EXCLUDES.
- Web Query subroutines ABS, INT, MAX, MIN, LOG, and SQRT.

  Note: Do not confuse the Web Query user-written subroutines MAX and MIN with the MAX. and MIN. prefix operators. DEFINE fields cannot include prefix operators.
- Expressions involving fields with ACTUAL=DATE, except for the subtraction of one DATE field from another and all logical expressions on DATE fields.
- Date-time manipulation handled by the Web Query date-time functions is not converted to SQL.

In addition, IF-THEN-ELSE optimization does not support the following features:

- Any type of DECODE expression.
- STATIC SQL.
- IF/WHERE DDNAME.
- Partial date selection.

Improving Optimizer Efficiency with Hints

DBMS Optimizer hints can be used to alter an execution plan. The adapter provides a setting which enable the TABLE command to place the hints at the end of the generated query for Microsoft SQL Server.

This occurs when the adapter constructs a single SELECT statement. It does not occur in the case of a FOCUS-managed Join when multiple SELECTs are generated.

To reverse the setting, use SET HINT without a hint_text parameter.

Syntax: How to Set Specific Hints

Use the following syntax to set specific hints:

```
SQL SQLMSS SET HINT OPTION (hint_text)
```

where

SQLMSS

Is the target RDBMS. You can omit this value if you previously issued the SET SQLENGINE command.

```
OPTION (hint_text)
```

Is the text of the hint or hints combination. The end user is responsible for the syntax. Omitting OPTION (*hint_text*) resets the hint to none.

Example: **Setting an Microsoft SQL Hint**

```
SQL SQLMSS SET HINT OPTION (FAST 2)
TABLE FILE STXT31M
 PRINT *
 BY F01INT
 END
```

The Web Query request is translated into a SELECT statement that incorporates the specified hint.

```
SELECT
   T1."F01INT",
   T1."F02CHAR_10",
   T1."F03VARCHAR_10"
FROM
   D999AIXPPC71XX_TTXT31M T1
ORDER BY
   T1."F01INT"
OPTION (FAST 2);
```

Calling a Microsoft SQL Server Stored Procedure Using SQL Passthru

Microsoft SQL Server stored procedures are supported using SQL Passthru. These procedures need to be developed within Microsoft SQL Server using the CREATE PROCEDURE command.

The adapter supports stored procedures with input, output, and in-out parameters.

The output parameter values that are returned by stored procedures are available as result sets. These values form a single-row result set that is transferred to the client after all other result sets are returned by the invoked stored procedure. The names of the output parameters (if available) become the column titles of that result set.

Note that only the output parameters (and the returned value) referenced in the invocation string are returned to the client. As a result, users have full control over which output parameters have their values displayed.

The server supports invocation of stored procedures written according to the rules of the underlying DBMS. Note that the examples shown in this section are SQL-based. See the DBMS documentation for rules, languages, and additional programming examples.

Syntax: How to Invoke a Stored Procedure

```
SQL SQLMSS EX procname [parameter_specification1]
[,parameter_specification2]...
END
```

where:

SQLMSS

Is the ENGINE suffix for Microsoft SQL Server.

procname

Is the name of the stored procedure. It is the fully or partially qualified name of the stored procedure in the native RDBMS syntax.

You can employ either SQL or SYS naming conventions to control the separator character used for interpreting multipart names, as described in Setting Naming Conventions.

parameter_specification

IN, OUT, and INOUT parameters are supported. Use the variation required by the stored procedure:

IN

Is a literal (for example, 125, 3.14, 'abcde'). You can use reserved words as input. Unlike character literals, reserved words are not enclosed in quotation marks (for example, NULL). Input is required.

OUT

Is represented as a question mark (?). You can control whether output is passed to an application by including or omitting this parameter. If omitted, this entry will be an empty string (containing 0 characters).

INOUT

Consists of a question mark (?) for output and a literal for input, separated by a slash: /. (For example: ?/125, ?/3.14, ?/'abcde'.) The out value can be an empty string (containing 0 characters).

Example: Invoking a Stored Procedure

In this example, a user invokes a stored procedure, edaqa.test_proc01, supplies input values for parameters 1, 3, 5 and 7, and requests the returned value of the stored procedure, as well as output values for parameters 2 and 3.

Note that parameters 4 and 6 are omitted; the stored procedure will use their default values, as specified at the time of its creation.

```
SQL SQLMSS EX edaqa.test_proc01 125,?,?/3.14,,'abc',,'xyz'
END
```

Example: Sample Stored Procedure

This stored procedure uses out and inout parameters:

```
CREATE PROCEDURE EDAQA.PROCP3 (   OUT chSQLSTATE_OUT   CHAR(5),
                                  OUT intSQLCODE_OUT   INT,
                                  INOUT l_name char(20),
                                  INOUT f_name char(20))
    RESULT SETS 1
    LANGUAGE SQL
------------------------------------------------------------------------
-- SQL Stored Procedure
------------------------------------------------------------------------
P1: BEGIN
    -- Declare variable
    DECLARE SQLSTATE CHAR(5) DEFAULT '00000';
    DECLARE SQLCODE INT DEFAULT 0;
    -- Declare cursor
    DECLARE cursor1 CURSOR WITH RETURN FOR
        SELECT
           EDAQA.NF29005.SSN5 AS SSN5,
           EDAQA.NF29005.LAST_NAME5 AS LAST_NAME5,
           EDAQA.NF29005.FIRST_NAME5 AS FIRST_NAME5,
           EDAQA.NF29005.BIRTHDATE5 AS BIRTHDATE5,
           EDAQA.NF29005.SEX5 AS SEX5
        FROM
           EDAQA.NF29005
        WHERE
           (
             ( EDAQA.NF29005.LAST_NAME5 =  l_name )
        AND
             ( EDAQA.NF29005.FIRST_NAME5 = f_name )
           );
     -- Cursor left open for client application
    OPEN cursor1;
    SET chSQLSTATE_OUT = SQLSTATE;
    SET intSQLCODE_OUT = SQLCODE;
    SET l_name = 'this is first name';
    SET f_name = 'this is last name';
END P1   @
```

Reference: Capturing Application Errors in Stored Procedures

You can capture application errors using the RAISERROR method. Any application error that is issued by the stored procedure is available in the server variable &MSSMSGTXT.

Chapter 10

Using the Adapter for MySQL

The Adapter for MySQL allows applications to access MySQL data sources. The adapter converts application requests into native MySQL statements and returns optimized answer sets to the requesting application.

In this chapter:

Preparing the MySQL Environment

The Adapter for MySQL requires the MySQL Connector/J JDBC driver. For Release 5.x, Version 5.0.4 or higher is required. For Release 3.x, Version 3.1.14 or higher is required.

In order to report against Unicode data in MySQL, you must enable Unicode support in the reporting server. For more information, see *MySQL and Unicode* on page 175.

Procedure: How to Prepare the MySQL Environment on Windows

1. Specify the location of the MySQL Connector/JDBC driver files in the CLASSPATH environment variable. You must specify the full location and name of the jar file. For example, if the jar file is located in C:\Program Files\MySQL\JDBC Driver, then specify:

   ```
   CLASSPATH= C:\Program Files\MySQL\JDBC Driver\mysql-connector-
   java-5.1.17-bin.jar
   ```

2. Specify the location of the Javarun time environment or development kit in the JAVA_HOME or JDK_HOME environment variable. Either is acceptable.

   You must specify the location where the run time environment is installed. You should see a sub-directory bin in that directory. For example, if you have the run time environment in C:\Program Files\java\jre6 then specify:

   ```
   JAVA_HOME=C:\Program Files\java\jre6
   ```

Alternately, if you have the full development kit installed in C:\Program Files\java \jdk1.6.0_17, then specify:

```
JDK_HOME= C:\Program Files\java\jdk1.6.0_17
```

3. Start (or restart) the server.

Note: You also need to restart the server if the driver has changed.

Procedure: How to Prepare the MySQL Environment on UNIX

Specify the location of several files:

1. Specify the location of the MySQL Connector/JDBC driver files in the $CLASSPATH environment variable.

   For example, if the files are located in /usr/driver_files, you would issue the following statements:

   ```
   CLASSPATH=/usr/driver_files/mydriver.jar;$CLASSPATH
   export CLASSPATH
   ```

   To ensure that the variable is set before server startup, add the CLASSPATH setting in your UNIX profile.

   Alternatively, you could use the Reporting Server browser interface to specify the location:

   a. From the Tools menu, select *Workspace*.
   b. Expand the *Java Services* folder, right-click *DEFAULT*, and select *Properties*.

      The Java Services Configuration pane opens.
   c. Select the *Class path* section.
   d. Specify the full path to the MySQL Connector/J jar file in the IBI_CLASSPATH field.
   e. Click *Save and Restart Java Services*.

2. Specify the location of the Java Development Kit's installation directory in the $JDK_HOME environment variable.

   For example, if you want to set the location of the Java Development Kit to /usr/java, you would issue the following statements:

   ```
   JDK_HOME=/usr/java
   export JDK_HOME
   ```

3. Specify the location of the Java Virtual Machine's installation directory in the $LD_LIBRARY_PATH environment variable.

   For example, if you want to set the location of the JVM to /usr/j2sdk1.4.2_01/jre/lib/i386/server, you would issue the following statements:

```
LD_LIBRARY_PATH=/usr/j2sdk1.4.2_01/jre/lib/i386/server
export LD_LIBRARY_PATH
```

Note that if the server is running with security on, the LD_LIBRARY_PATH variable is ignored. In this case, you must use IBI_LIBPATH.

4. Start (or restart) the server.

Note: You also need to restart the server if the driver has changed.

MySQL and Unicode

The Adapter for MySQL is implemented using JDBC. This implementation supports Unicode data stored in character fields with the CHARACTER SET set to UTF-8.

You must set the LANG environment variable in the edastart file or in a separate shell file before you start the server. For example, for American English you would export the following variable:

```
export LANG=EN_US.UTF-8
```

For details, see *Unicode Support* in the *Server Administration* manual.

Configuring the Adapter for MySQL

Configuring the adapter consists of specifying connection and authentication information for each of the connections you want to establish.

Declaring Connection Attributes

The SET CONNECTION_ATTRIBUTES command allows you to declare a connection to one MySQL database server and to supply authentication attributes necessary to connect to the server.

You can declare connections to more than one MySQL database server by issuing multiple SET CONNECTION_ATTRIBUTES commands. The actual connection takes place when the first query referencing that connection is issued (see *Overriding the Default Connection* on page 179). You can include SET CONNECTION_ATTRIBUTES commands in an RPC or a server profile. The profile can be encrypted.

If you issue multiple SET CONNECTION_ATTRIBUTES commands:

- The *first* SET CONNECTION_ATTRIBUTES command sets the default MySQL database server to be used.

- If more than one SET CONNECTION_ATTRIBUTES command declares the same MySQL database server, the authentication information is taken from the *last* SET CONNECTION_ATTRIBUTES command.

Reference: Connection Attributes for MySQL

The MySQL adapter is under the *SQL* group folder.

The following list describes the connection attributes for which you can supply values. To complete the attribute declaration, click the *Configure* button.

Connection name

Logical name used to identify this particular set of connection attributes. The default is CON01.

URL

Enter the location URL for the MySQL data source. The basic syntax is

```
jdbc:mysql://host/database
```

where:

host

Is the computer name or IP address on which the MySQL database is located.

database

Is the name of the database.

These are two examples:

```
jdbc:mysql://localhost/qatst
```

```
jdbc:mysql://edaaix52/qatst
```

Beginning with MySQL Release 4.1, you can reference additional MySQL connection properties in the URL. If you wish to do so, follow these guidelines: in the URL the first property must be preceded by the ? character, and the second and subsequent properties referenced in the URL must be preceded by the & character followed immediately by an | character, as illustrated in the following example.

Suppose that you wish to add the following connection properties:

```
sessionVariables=sql_mode=PIPES_AS_CONCAT
 zeroDateTimeBehavior=convertToNull
```

Enter the URL as follows:

```
jdbc:mysql://host/database?sessionVariables=sql_mode=P
IPES_AS_CONCAT&|
 zeroDateTimeBehavior=convertToNull
```

To use the adapter with SSL, you need to add the verifyServerCertificate, requireSSL, and useSSL properties, with useSSL last, as follows:

```
jdbc:mysql://host:port/server?verifyServerCertificate=false&|
requireSSL=true&|useSSL=true
```

Note: The URL must be entered as a single line, without a space after the | character.

Driver name

Name of the MySQL JDBC driver.

For Connector/J 8.0 and higher, use the following driver:

```
com.mysql.cj.jdbc.Driver
```

For Connector/J versions lower than 8.0, use the following driver:

```
com.mysql.jdbc.Driver
```

Security

There are two methods by which a user can be authenticated when connecting to a database server:

- **Explicit.** The user ID and password are explicitly specified for each connection and passed to the database, at connection time, for authentication.
- **Password Passthru.** The user ID and password received from the client application are passed to the database, at connection time, for authentication.

User

Primary authorization ID by which you are known to the data source.

Password

Password associated with the primary authorization ID.

Select profile

Select a profile from the drop-down menu to indicate the level of profile in which to store the CONNECTION_ATTRIBUTES command. The global profile, edasprof.prf, is the default.

If you wish to create a new profile, either a user profile (*user*.prf) or a group profile if available on your platform (using the appropriate naming convention), choose *New Profile* from the drop-down menu and enter a name in the Profile Name field (the extension is added automatically).

Store the connection attributes in the server profile (edasprof).

Syntax: How to Declare Connection Attributes Manually

For explicit authentication:

```
ENGINE MYSQL SET CONNECTION_ATTRIBUTES [connection]/userid,password
```

For password passthru authentication:

```
ENGINE MYSQL SET CONNECTION_ATTRIBUTES connection/
```

where:

`MYSQL`

Indicates the adapter. You can omit this value if you previously issued the SET SQLENGINE command.

`connection`

The name of the connection. You can give any name to the connection that you want.

`userid`

Is the primary authorization ID by which you are known to MySQL.

`password`

Is the password associated with the primary authorization ID.

Example: Declaring Connection Attributes

The following SET CONNECTION_ATTRIBUTES command connects to the MySQL database server named TEST with an explicit user ID and password:

```
ENGINE MYSQL SET CONNECTION_ATTRIBUTES TEST/USERA,PWDA
```

The following SET CONNECTION_ATTRIBUTES command connects to the MySQL database server named TEST using password passthru authentication:

```
ENGINE MYSQL SET CONNECTION_ATTRIBUTES TEST/
```

Authenticating a User

There are two methods by which a user can be authenticated when connecting to a MySQL database server:

- **Explicit.** User IDs and passwords are explicitly stated in SET CONNECTION_ATTRIBUTES commands. You can include these commands in the server global profile, edasprof.prf, for all users.
- **Database or Password Passthru.** User ID and password received from the client application are passed to the MySQL database server for authentication.

When a client connects to the server, the user ID and password are passed to MySQL for authentication and are not authenticated by the server. To implement this type of authentication, start the server with security turned off. The server allows the client connection, and then stores an encrypted form of the client connection message to be used for connection to a MySQL database server at anytime during the lifetime of the server agent.

Overriding the Default Connection

Once all MySQL connections to be accessed have been declared using the SET CONNECTION_ATTRIBUTES command, there are two ways to select a specific MySQL connection from the list of declared connections:

- You can select a default connection using the SET DEFAULT_CONNECTION command. If you do not issue this command, the connection name value specified in the *first* SET CONNECTION_ATTRIBUTES command is used.
- You can include the CONNECTION= attribute in the Access File of the table specified in the current SQL query. When you include a connection name in the CREATE SYNONYM command from the Reporting Server browser interface, the CONNECTION= attribute is automatically included in the Access File. This attribute supersedes the default connection.

Syntax: How to Select a Connection to Access

```
ENGINE MYSQL SET DEFAULT_CONNECTION [connection]
```

where:

`MYSQL`

Indicates the adapter. You can omit this value if you previously issued the SET SQLENGINE command.

connection

Is the connection name specified in a previously issued SET CONNECTION_ATTRIBUTES command. If omitted, then the local database server will be set as the default. If this connection name has not been previously declared, a FOC1671 message is issued.

Note:

- If you use the DEFAULT_CONNECTION command more than once, the connection name specified in the last command will be the active connection name.
- The DEFAULT_CONNECTION command cannot be issued while an uncommitted transaction (LUW) is pending. In that case, a FOC1671 message is issued.

Controlling Connection Scope

This topic explains how to set the scope of logical units of work using adapters. This is accomplished by the SET AUTODISCONNECT command.

A connection occurs at the first interaction with the declared database server.

Syntax: How to Control the Connection Scope

You change this setting by expanding the Domains and Reports folders, then right-clicking a subfolder and choosing *Metadata*. Right-click MS SQL Server 200*x* and select *Change Settings*. The Change Settings pane opens.

`MYSQL`

Indicates the adapter. You can omit this value if you previously issued the SQLENGINE command.

`FIN`

Disconnects automatically only after the session has been terminated. FIN is the default value.

`COMMAND`

Disconnects automatically after each request. Depending on how often the event occurs, the AUTODISCONNECT command may result in considerable overhead. Almost all of this overhead is not related to the server. It is related to the operating system and the data source.

Managing MySQL Metadata

This topic describes how to use CREATE SYNONYM for MySQL data sources. It also describes MySQL data type support.

Creating Synonyms

Synonyms define unique names (or aliases) for each MySQL table or view that is accessible from the server. Synonyms are useful because they hide location information and the identity of the underlying data source from client applications. They also provide support for extended metadata features of the server such as virtual fields and additional security mechanisms.

Using synonyms allows an object to be moved or renamed while allowing client applications to continue functioning without modification. The only modification required is a redefinition of the synonym on the server. The result of creating a synonym is a Master File and Access File.

Procedure: How to Create a Synonym

1. From the Reporting Server browser interface Application page, click *Get Data*.
2. On the Configured Adapters section of the page, in Simple Mode, right-click an adapter and click *Show Connections*. Right-click a connection.

   Depending on the type of adapter you choose, one of the following options appears on the context menu.

   - **Show DBMS objects.** This option opens the page for selecting synonym objects and properties.
   - **Create metadata objects.** This option opens the page for selecting synonym objects and properties.
   - **Show files.** This option opens a file picker. After you choose a file of the correct type, the page for selecting synonym objects and properties opens.
   - **Show local files.** This option opens a file picker. After you choose a file of the correct type, the page for selecting synonym objects and properties opens.
   - **Show topics.** This option opens the page for selecting synonym objects and properties for topics within the environment.
3. Enter values for the parameters required by the adapter as described in the chapter for your adapter.
4. After entering the parameter values, click *Add*.

   This button may be labeled *Next*, *Create Synonym*, *Create Base Synonyms*, *Create Cluster Synonym*, or *Update Base Synonyms*.

The synonym creation process for most adapters has been consolidated so that you can enter all necessary parameters on one page. However, for some adapters such as LDAP, continue clicking *Next* until you get to a page that has a *Create Synonym* button.

The synonym is created and added under the specified application directory.

Note: When creating a synonym, if you select the Validate check box, where available, the server adjusts special characters and checks for reserved words.

Reference: Synonym Creation Parameters for MySQL

The following list describes the synonym creation parameters for which you can supply values.

Restrict Object Type to

Restrict candidates for synonym creation based on the selected object type(s): Tables, Views, External SQL Scripts, and any other supported objects.

Choosing *External SQL Scripts* from the drop-down list enables you to represent an SQL Query as a synonym for read-only reporting. A Synonym candidate can be any file that contains one (and only one) valid SQL Query and does not contain end-of-statement delimiters (";" or "/") and comments.

Depending on the adapter, you can further restrict your search by choosing

Object name

Selecting this option adds the Object Name parameters to the screen.

Type a string for filtering the object names, inserting the wildcard character (%) as needed at the beginning and/or end of the string. For example, enter: ABC% to select all objects whose names begin with the letters ABC; %ABC to select all whose names end with the letters ABC; %ABC% to select all whose names contain the letters ABC at the beginning, middle, or end.

Location of External SQL Scripts

If you specify *External SQL Scripts* in the *Restrict Object type to* field, these additional fields are displayed.

The following standard naming conventions apply for UNIX, IBM i IFS, and z/OS HFS:

- In the *Base Location* field, specify the physical directory location of the file that contains the SQL Query. You can type a directory name or click on the ellipsis. This opens the Select Base Location dialog box.
- In the *Document Name* field, enter the file name with or without wild card characters.

- In the *Document Extension* field, enter the extension of the script files to filter the list of candidates.

On IBM i, you can use alternative IFS naming conventions to access library members. The following entry illustrates this method:

- In the *Base Location* field, enter:

  `/QSYS.LIB/MYLIBRARY.LIB/MYSRC.FILE`

- The *Document Extension* is understood to be MBR. You can enter this value explicitly or leave the input box blank.

During synonym generation, the adapter issues native API calls to obtain a list of elements in the select list and builds the Master File with a field for each element. The generated Access File references the location of the SQL script in the DATASET attribute, which contains the full path, including the name and extension of the file containing the SQL Query. For example,

```
DATASET=/ul/home2/apps/report3.sql
```

When a Web Query report is created, the SQL Query is used to access data.

Cardinality

Select the *Cardinality* check box to reflect the current cardinality (number of rows or tuples) in the table during metadata creation. Cardinality is used for equi-joins. The order of retrieval is based on the size (cardinality) of the table. Smaller tables are read first.

If the cardinality of the tables to be used in the application are dynamic, it may not be beneficial to choose this setting.

For Subquery

Only available when *External SQL Scripts* is selected from the Restrict objects type to drop-down menu. When selected, a SUBQUERY keyword is added to the Access File of the generated synonym. If the corresponding SQL string has valid syntax that can be used in the FROM statement of the generated SQL (what is known as a Derived Table), then the SQL SCRIPT will be processed as a subquery embedded into a FROM clause. This usage allows for more flexibility. For example, the synonym can be used as a target for a JOIN.

If the SQL SCRIPT has parameter markers, such as ? or :, or the syntax contains constructs that are invalid for a derived table, for example ORDER BY, then this keyword should not be selected. At runtime, if SUBQUERY=Y is present and it is determined that the SQL SCRIPT cannot be used in the FROM statement, the setting will be ignored, and a FOC1782 warning message will be issued. The default is selected (SUBQUERY=Y).

Application

The default value is baseapp.

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 the prefix and suffix fields blank.

Customize data type mappings

To change the data type mappings from their default settings, select this check box. The customizable mappings are displayed.

For information about customizable mappings, see *MySQL Data Type Support* on page 188.

Update or Create Metadata

Select *Create* to overwrite any existing synonym with the same fully-qualified name, or *Update* to synchronize the metadata with an existing synonym. If you select *Update*, the next screen will show a list of attributes from the DBMS catalog that you can check to allow attributes from the DBMS catalog to override attributes from the existing synonym.

Overwrite Existing Synonyms

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.

Default Synonym Name

This column displays the name that will be assigned to each synonym. To assign a different name, replace the displayed value.

Table name

Is the name of the underlying object.

Type

The object type (Table, View, and so on).

Select tables

Select tables for which you wish to create synonyms:

- To select all tables in the list, select the *Select All* check box.
- To select specific tables, select the corresponding check boxes.

Example: Sample Generated Synonym

An Adapter for MySQL synonym comprises a Master File and an Access File. This is a synonym for the table nf29004.

Generated Master File nf29004.mas

```
FILE=DIVISION, SUFFIX=MYSQL ,$
SEGNAME=SEG1_4, SEGTYPE=S0 ,$
FIELD=DIVISION4,    DIVISION4,    I9,  I4,  MISSING=OFF ,$
FIELD=DIVISION_NA4, DIVISION_NA4, A25, A25, MISSING=ON  ,$
FIELD=DIVISION_HE4, DIVISION_HE4, I9,  I4,  MISSING=ON  ,$
```

Generated Access File nf29004.acx

```
SEGNAME=SEG1_4,TABLENAME=EDAQA.NF29004,
CONNECTION=TEST,KEYS=1, WRITE=YES,$
```

Reference: Access File Keywords

This chart describes keywords in the Access File.

Keyword	Description
SEGNAME	Value must be identical to the SEGNAME value in the Master File.
TABLENAME	Identifies the MySQL tablename. The tablename may include owner (schema) name. For example, TABLENAME=[*owner.*]*tablename*

Keyword	Description
CONNECTION	Indicates a previously declared connection. The syntax is: CONNECTION=*connection* CONNECTION=' ' indicates access to the local database server. Absence of the CONNECTION attribute indicates access to the default database server.
KEYS	Indicates how many columns constitute the primary key for the table. Corresponds to the first *n* fields in the Master File segment. This attribute requires the columns that constitute the key to be described first in the Master File. See the KEY attribute below for information about specifying the key fields without having to describe them first in the Master File.
KEY	Specifies the columns that participate in the primary key without having to describe them first in the Master File. The syntax is: KEY=*fld1/fld2/.../fldn*
WRITE	Specifies whether write operations are allowed against the table.

Keyword	Description
KEYFLD IXFLD	Supply the names of the primary key and foreign key fields that implement the relationships established by the multi-table Master File. Together, KEYFLD and IXFLD identify the field shared by a related table pair. ❏ KEYFLD is the FIELDNAME of the common column from the parent table. ❏ IXFLD is the FIELDNAME of the common column from the related table. KEYFLD and IXFLD must have the same data type. It is recommended, but not required, that their lengths also be the same. **Note:** An RDBMS index on both the KEYFLD and IXFLD columns provides the RDBMS with a greater opportunity to produce efficient joins. The columns must have the same data type. If their length is the same, the RDBMS handles the join more efficiently.
AUTO INCREMENT	Set to Yes to enable autoincrementing.
START	Initial value in incrementing sequence
INCREMENT	Increment interval.
INDEX_NAME INDEX_UNIQUE INDEX_COLUMNS INDEX_ORDER	Indicate a name of the index in a database, uniqueness, name, and order of the indexed column(s).

Reference: **Managing Synonyms**

Once you have created a synonym, you can right-click the synonym name in the navigation pane to access the available options.

MySQL Data Type Support

SQL Data Type mapping options are available in a report available from the Reporting Server browser interface.

Customizing the Adapter for the MySQL Environment

This topic describes how to set the adapter for the MySQL environment.

PASSRECS

PASSRECS returns the number of rows affected by a successfully executed SQL Passthru INSERT, UPDATE, or DELETE command.

Tip: You can change this setting manually or from the Reporting Server browser interface by clicking *Get Data* on the menu bar, clicking a configured adapter, and choosing *Change Settings* from the menu. The Change Settings pane opens.

Syntax: How to Set PASSRECS

```
ENGINE MYSQL SET PASSRECS {ON|OFF}
```

where:

`MYSQL`

Indicates the adapter. You can omit this value if you previously issued the SET SQLENGINE command.

`ON`

Provides the number of rows affected in the application program SCB count member after the successful execution of an SQL Passthru INSERT, UPDATE, or DELETE command. ON is the default value.

`OFF`

Provides no information after the successful execution of an SQL Passthru INSERT, UPDATE, or DELETE command.

Specifying the Transaction Isolation Level

You can specify the transaction isolation level from the Reporting Server browser interface or using the SET ISOLATION command.

Syntax: How to Specify Transaction Isolation Level From a SET Command

You can specify transaction isolation level by issuing the following command

```
ENGINE MYSQL SET ISOLATION {RU|RC|RR|SE}
```

where:

`RU`

Sets the transaction isolation level to Read Uncommitted.

`RC`

Sets the transaction isolation level to Read Committed.

`RR`

Sets the transaction isolation level to Repeatable Read.

`SE`

Sets the transaction isolation level to Serializable Read.

Cancelling Long Requests

You can cancel long running requests from the Reporting Server browser interface. Depending on the capabilities of the native JDBC driver, this action will either cancel the request entirely or break out of the fetch cycle.

Procedure: How to Cancel Long Requests

1. From the Reporting Server browser interface menu bar choose *Workspace*, *Java Services*, DEFAULT. Right-click on *DEFAULT* and select *Agents*.
2. In the Java Services Agents pane, highlight a row with the jscomid you wish to kill, right-click on it and select *Stop.*

MySQL Optimization Settings

Adapter optimization allows the RDBMS to perform the work for which it is best suited, reducing the volume of RDBMS-to-server communication and improving response time. It also enables the RDBMS to exploit its own internal optimization techniques.

Chapter 11

Using the Adapter for PostgreSQL

The Adapter for PostgreSQL allows applications to access PostgreSQL data sources. The adapter converts application requests into PostgreSQL calls and returns optimized answer sets to the requesting application.

In this chapter:

Preparing the PostgreSQL Environment

In order to use the Adapter for PostgreSQL, you must install the PostgreSQL driver, set its CLASSPATH value before server startup, and ensure that the JSCOM3 service is running.

Procedure: How to Set Up the Environment on Windows and UNIX

1. Identify the location of the PostgreSQL Driver files by adding them to the environment variable CLASSPATH before server startup.

   For example, to set a UNIX or IBM i location for some JDBC Driver files to /usr/driver_files/mydriver.jar, issue the commands:

   ```
   CLASSPATH=/usr/driver_files/mydriver.jar;$CLASSPATH
   export CLASSPATH
   ```

   Similarly, on Windows, you would issue the command:

   ```
   set CLASSPATH=c:\usr\driver_files\mydriver.jar:%CLASSPATH%
   ```

   Note that if you run the server workspace as a Windows service, you must set the CLASSPATH as a system-wide environment variable.

   Similarly, on UNIX, to ensure that the variable is set before server startup, add the CLASSPATH setting in your UNIX profile.

2. Start (or restart) the server.

   (Note that you also need to restart the server if the driver has changed.)

3. Verify that the JSCOM3 service is active by issuing the following command in the server workspace

   ```
   edastart -show
   ```

   and check the special services section displays "JSCOM3 active".

   If the JSCOM3 service is not active, a "fail to start" message will typically also be in the server EDAPRINT log. To resolve the problem, review the JDBC listener requirements in the chapter for your operating system in the *Server Installation* manual.

Configuring the Adapter for PostgreSQL

Configuring the adapter consists of specifying connection and authentication information for each of the connections you want to establish.

Declaring Connection Attributes

In order to connect to a PostgreSQL data source, the adapter requires connection and authentication information.

You supply this information using the SET CONNECTION_ATTRIBUTES command. You can enter connection and authentication information in the Reporting Server browser interface or the Data Migrator desktop interface configuration pane. The consoles add the command to the server profile you select: global (edasprof.prf), user (user.prf), or group (group.prf), if supported on your platform.

You can declare connections to more than one PostgreSQL data source using the SET CONNECTION_ATTRIBUTES commands. The actual connection takes place when the first query is issued. If you issue multiple SET CONNECTION_ATTRIBUTES commands:

Reference: Connection Attributes for PostgreSQL

The *PostgreSQL* adapter is under the *SQL* group folder.

The following list describes the connection attributes for which you can supply values. To complete the attribute declaration, click the *Configure* button.

Connection name

Logical name used to identify this particular set of connection attributes. The default is CON01.

URL

Location URL for the PostgreSQL data source.

Driver name

Name for the PostgreSQL JDBC driver.

For example: org.postgresql.Driver

See PostgreSQL documentation for the specific driver release you are using.

IBI_CLASSPATH

Defines the additional Java Class directories or full-path jar names which will be available for Java Services. Value may be set by editing the communications file or in the Reporting Server browser interface. Using the Reporting Server browser interface, you can enter one reference per line in the input field. When the file is saved, the entries are converted to a single string using colon (:) delimiters for all platforms. OpenVMS platforms must use UNIX-style conventions for values (for example, /mydisk/myhome/myclasses.jar, rather than mydisk:[myhome]myclasses.jar) when setting values. When editing the file manually, you must maintain the colon delimiter.

Security

There are two methods by which a user can be authenticated when connecting to a database server:

- **Explicit.** The user ID and password are explicitly specified for each connection and passed to the database, at connection time, for authentication.
- **Password Passthru.** The user ID and password received from the client application are passed to the database, at connection time, for authentication.

User

Primary authorization ID by which you are known to the data source.

Password

Password associated with the primary authorization ID.

Select profile

Select a profile from the drop-down menu to indicate the level of profile in which to store the CONNECTION_ATTRIBUTES command. The global profile, edasprof.prf, is the default.

If you wish to create a new profile, either a user profile (*user*.prf) or a group profile if available on your platform (using the appropriate naming convention), choose *New Profile* from the drop-down menu and enter a name in the Profile Name field (the extension is added automatically).

Store the connection attributes in the server profile (edasprof).

Syntax: How to Declare Connection Attributes Manually

```
ENGINE SQLPSTGR SET CONNECTION_ATTRIBUTES connection
'URL'/userid,password
```

where:

`SQLPSTGR`

Indicates the adapter. You can omit this value if you previously issued the SET SQLENGINE command.

`connection`

Is the connection name.

`URL`

Is the URL to the location of the PostgreSQL data source.

`userid`

Is the primary authorization ID by which you are known to the target database.

`password`

Is the password associated with the primary authorization ID.

Example: Declaring Connection Attributes

The following SET CONNECTION_ATTRIBUTES command connects to a data source using the PostgreSQL Driver, with an explicit user ID (MYUSER) and password (PASS). To ensure security, specifying connection attributes from the Reporting Server browser interface or Data Migrator desktop interface will encrypt the password before adding it to the server profile.

Note: Consult vendor documentation for the exact name, port, and path.

```
ENGINE SQLPSTGR SET CONNECTION_ATTRIBUTES CON1
  'jdbc:postgresql://hostname:5432/qatst'
```

Overriding the Default Connection

If multiple connections have been defined, the connection named in the *first* SET CONNECTION_ATTRIBUTES command serves as the default connection.

Syntax: How to Change the Default Connection

You change this setting by expanding the Domains and Reports folders, then right-clicking a subfolder and choosing *Metadata*. Right-click *SQLPSTGR* and select *Change Settings*. The Change Settings pane opens.

`SQLPSTGR`

Indicates the adapter. You can omit this value if you previously issued the SQLENGINE command.

`connection`

Is the connection defined in a previously issued CONNECTION_ATTRIBUTES command. If this name was not previously declared, the following message is issued:

```
FOC1671, Command out of sequence
```

Note:

- If you use the DEFAULT_CONNECTION command more than once, the connection name specified in the *last* command serves as the default connection.
- The DEFAULT_CONNECTION command cannot be issued while an uncommitted transaction (LUW) is pending. In that case, the following message is issued:

```
FOC1671, Command out of sequence.
```

Controlling the Connection Scope

The SET AUTODISCONNECT command controls the persistence of connections when using the adapter for each of the connections you want to establish.

Syntax: How to Control the Connection Scope

You change this setting by expanding the Domains and Reports folders, then right-clicking a subfolder and choosing *Metadata*. Right-click *SQLPSTGR* and select *Change Settings*. The Change Settings pane opens.

`SQLPSTGR`

Indicates the adapter. You can omit this value if you previously issued the SQLENGINE command.

`FIN`

Disconnects automatically only after the session has been terminated. FIN is the default value.

COMMIT

Disconnects automatically only after COMMIT or ROLLBACK is issued as a native SQL command.

Managing PostgreSQL Metadata

When the server accesses a data source, it needs to know how to interpret the data stored there. For each object the server will access, you create a synonym that describes its structure and the server mapping of the data types.

Identifying the Adapter

The SUFFIX attribute in the Master File identifies the adapter needed to interpret a request. Use the SUFFIX value SQLPSTGR to identify the Adapter for PostgreSQL.

Syntax: How to Identify the Adapter

```
FILE[NAME]=file, SUFFIX=SQLPSTGR [,$]
```

where:

file

Is the file name for the Master File. The file name should start with a letter and be representative of the table or view contents. The actual file must have a .mas extension, but the value for this attribute should not include the extension.

SQLPSTGR

Is the value for the adapter.

Accessing Database Tables

If you choose to access a remote third-party table using PostgreSQL, you must locally install the RDBMS PostgreSQL Driver.

The Server can access third-party database tables across the PostgreSQL network. You must specify a URL for the data source and, possibly, a user ID and/or password for the database you are accessing. You can define these parameters in either the server global profile or in a user profile.

Creating Synonyms

Synonyms define unique names (or aliases) for each object that is accessible from the server. Synonyms are useful because they hide the underlying data source's location and identity from client applications. They also provide support for extended metadata features of the server, such as virtual fields and additional security mechanisms.

Using synonyms allows an object to be moved or renamed while allowing client applications to continue functioning without modification. The only modification required is a redefinition of the synonym on the server. The result of creating a synonym is a Master File and an Access File, which represent the server's metadata.

Procedure: How to Create a Synonym

1. From the Reporting Server browser interface Application page, click *Get Data*.
2. On the Configured Adapters section of the page, in Simple Mode, right-click an adapter and click *Show Connections*. Right-click a connection.

   Depending on the type of adapter you choose, one of the following options appears on the context menu.

   - **Show DBMS objects.** This option opens the page for selecting synonym objects and properties.
   - **Create metadata objects.** This option opens the page for selecting synonym objects and properties.
   - **Show files.** This option opens a file picker. After you choose a file of the correct type, the page for selecting synonym objects and properties opens.
   - **Show local files.** This option opens a file picker. After you choose a file of the correct type, the page for selecting synonym objects and properties opens.
   - **Show topics.** This option opens the page for selecting synonym objects and properties for topics within the environment.
3. Enter values for the parameters required by the adapter as described in the chapter for your adapter.
4. After entering the parameter values, click *Add*.

   This button may be labeled *Next*, *Create Synonym*, *Create Base Synonyms*, *Create Cluster Synonym*, or *Update Base Synonyms*.

   The synonym creation process for most adapters has been consolidated so that you can enter all necessary parameters on one page. However, for some adapters such as LDAP, continue clicking *Next* until you get to a page that has a *Create Synonym* button.

The synonym is created and added under the specified application directory.

Note: When creating a synonym, if you select the Validate check box, where available, the server adjusts special characters and checks for reserved words.

Reference: Synonym Creation Parameters for PostgreSQL

The following list describes the synonym creation parameters for which you can supply values.

Restrict Object Type to

Restrict candidates for synonym creation based on the selected object type(s): Tables, Views, External SQL Scripts, and any other supported objects.

Choosing *External SQL Scripts* from the drop-down list enables you to represent an SQL Query as a synonym for read-only reporting. A Synonym candidate can be any file that contains one (and only one) valid SQL Query and does not contain end-of-statement delimiters (";" or "/") and comments.

Depending on the adapter, you can further restrict your search by choosing

Filter by Owner/Schema and Object name

Selecting this option adds the Owner/Schema and Object Name parameters to the screen.

- **Owner/Schema.** Type a string for filtering the selection, inserting the wildcard character (%) as needed at the beginning and/or end of the string. For example, enter: ABC% to select tables or views whose owner/schema begin with the letters ABC; %ABC to select tables or views whose owner/schema end with the letters ABC; %ABC% to select tables or views whose owner/schema contain the letters ABC at the beginning, middle, or end.
- **Object name.** Type a string for filtering the object names, inserting the wildcard character (%) as needed at the beginning and/or end of the string. For example, enter: ABC% to select all objects whose names begin with the letters ABC; %ABC to select all whose names end with the letters ABC; %ABC% to select all whose names contain the letters ABC at the beginning, middle, or end.

Location of External SQL Scripts

If you specify *External SQL Scripts* in the *Restrict Object type to* field, these additional fields are displayed.

The following standard naming conventions apply for UNIX, IBM i IFS, and z/OS HFS:

- In the *Base Location* field, specify the physical directory location of the file that contains the SQL Query. You can type a directory name or click on the ellipsis. This opens the Select Base Location dialog box.
- In the *Document Name* field, enter the file name with or without wild card characters.
- In the *Document Extension* field, enter the extension of the script files to filter the list of candidates.

On IBM i, you can use alternative IFS naming conventions to access library members. The following entry illustrates this method:

- In the *Base Location* field, enter:

  ```
  /QSYS.LIB/MYLIBRARY.LIB/MYSRC.FILE
  ```

- The *Document Extension* is understood to be MBR. You can enter this value explicitly or leave the input box blank.

During synonym generation, the adapter issues native API calls to obtain a list of elements in the select list and builds the Master File with a field for each element. The generated Access File references the location of the SQL script in the DATASET attribute, which contains the full path, including the name and extension of the file containing the SQL Query. For example,

```
DATASET=/ul/home2/apps/report3.sql
```

When a Web Query report is created, the SQL Query is used to access data.

Cardinality

Select the *Cardinality* check box to reflect the current cardinality (number of rows or tuples) in the table during metadata creation. Cardinality is used for equi-joins. The order of retrieval is based on the size (cardinality) of the table. Smaller tables are read first.

If the cardinality of the tables to be used in the application are dynamic, it may not be beneficial to choose this setting.

Build cluster using foreign keys (deprecated)

You can select the *Build cluster using foreign keys* check box to include within this synonym every table related to the current table by a foreign key. However, this option has been deprecated, as the recommended way to create a cluster is by using the Synonym Editor. The resulting multi-table synonym describes all of the foreign key relationships of this table.

For Subquery

Only available when *External SQL Scripts* is selected from the Restrict objects type to drop-down menu. When selected, a SUBQUERY keyword is added to the Access File of the generated synonym. If the corresponding SQL string has valid syntax that can be used in the FROM statement of the generated SQL (what is known as a Derived Table), then the SQL SCRIPT will be processed as a subquery embedded into a FROM clause. This usage allows for more flexibility. For example, the synonym can be used as a target for a JOIN.

If the SQL SCRIPT has parameter markers, such as ? or :, or the syntax contains constructs that are invalid for a derived table, for example ORDER BY, then this keyword should not be selected. At runtime, if SUBQUERY=Y is present and it is determined that the SQL SCRIPT cannot be used in the FROM statement, the setting will be ignored, and a FOC1782 warning message will be issued. The default is selected (SUBQUERY=Y).

Application

The default value is baseapp.

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 the prefix and suffix fields blank.

Customize data type mappings

To change the data type mappings from their default settings, select this check box. The customizable mappings are displayed.

For information about customizable mappings, see *PostgreSQL Data Type Support* on page 203.

Update or Create Metadata

Select *Create* to overwrite any existing synonym with the same fully-qualified name, or *Update* to synchronize the metadata with an existing synonym. If you select *Update*, the next screen will show a list of attributes from the DBMS catalog that you can check to allow attributes from the DBMS catalog to override attributes from the existing synonym.

Overwrite Existing Synonyms

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.

Default Synonym Name

This column displays the name that will be assigned to each synonym. To assign a different name, replace the displayed value.

Owner/Schema

The user account that created the object or a collection of objects owned by a user.

Table name

Is the name of the underlying object.

Type

The object type (Table, View, and so on).

Select tables

Select tables for which you wish to create synonyms:

- To select all tables in the list, select the *Select All* check box.
- To select specific tables, select the corresponding check boxes.

Example: Sample Generated Synonym

An Adapter for PostgreSQL synonym comprises a Master File and an Access File. This is a synonym for the table nf29004.

Master File nf29004.mas

```
FILE=DIVISION, SUFFIX=SQLPSTGR ,$
SEGNAME=SEG1_4, SEGTYPE=S0 ,$
FIELD=DIVISION4,    DIVISION4, I9, I4, MISSING=OFF ,$
FIELD=DIVISION_NA4, DIVISION4, I9, I4, MISSING=OFF ,$
FIELD=DIVISION_HE4, DIVISION4, I9, I4, MISSING=OFF ,$
```

Access File nf29004.acx

```
SEGNAME=SEG1_4,TABLENAME=EDAQA.NF29004,
CONNECTION=CON1,KEYS=1,WRITE=YES,$
```

Reference: Access File Keywords

This chart describes the keywords in the Access File.

Keyword	Description
SEGNAME	Value must be identical to the SEGNAME value in the Master File.
TABLENAME	Identifies the PostgreSQL table. The value assigned to this attribute can include the name of the owner (also known as schema) and the database link name as follows: TABLENAME=[*owner.*]*table*

Keyword	Description
CONNECTION	Indicates a previously declared connection. The syntax is: CONNECTION=*connection* CONNECTION=' ' indicates access to the local data source. Absence of the CONNECTION attribute indicates access to the default database server.
KEYS	Indicates how many columns constitute the primary key for the table. Corresponds to the first *n* fields in the Master File segment. See the KEY attribute below for information about specifying the key fields without having to describe them first in the Master File.
KEY	Specifies the columns that participate in the primary key without having to describe them as the first fields in the Master File. The syntax is: KEY=*fld1/fld2/.../fldn*
WRITE	Specifies whether write operations are allowed against the table.
KEYFLD IXFLD	Supply the names of the primary key and foreign key fields that implement the relationships established by the multi-table Master File. Together, KEYFLD and IXFLD identify the field shared by a related table pair. ❑ KEYFLD is the FIELDNAME of the common column from the parent table. ❑ IXFLD is the FIELDNAME of the common column from the related table. KEYFLD and IXFLD must have the same data type. It is recommended, but not required, that their lengths also be the same. **Note:** An RDBMS index on both the KEYFLD and IXFLD columns provides the RDBMS with a greater opportunity to produce efficient joins. The columns must have the same data type. If their length is the same, the RDBMS handles the join more efficiently.

Reference: **Managing Synonyms**

Once you have created a synonym, you can right-click the synonym name in the navigation pane to access the available options.

PostgreSQL Data Type Support

SQL Data Type mapping options are available in a report available from the Reporting Server browser interface.

Data Type Mapping for A256V

The Adapter for PostgreSQL supports data type mapping using the following setting:

```
ENGINE SQLPSTGR SET CONVERSION LONGCHAR ALPHA n
```

The setting affects the mapping of all PostgreSQL native fields that have a data type of [VAR]CHAR(32767) or TEXT. The value of *n* ranges from 1 to 32767. The default value is 256. The TEXT data type (no *n* length) can be set instead of ALPHA when necessary.

Customizing the PostgreSQL Environment

The Adapter for PostgreSQL provides several parameters for customizing the environment and optimizing performance.

Specifying a Timeout Limit

TIMEOUT specifies the number of seconds the adapter will wait for a response after you issue an SQL request to PostgreSQL.

Syntax: **How to Issue the TIMEOUT Command**

```
ENGINE SQLPSTGR SET TIMEOUT {nn|0}
```

where:

`SQLPSTGR`

Indicates the adapter. You can omit this value if you previously issued the SET SQLENGINE command.

nn

Is the number of seconds before a time-out occurs. 30 is the default value.

`0`

Represents an infinite period to wait for a response.

Cancelling Long Requests

You can cancel long running requests from the Reporting Server browser interface. Depending on the capabilities of the native JDBC driver, this action will either cancel the request entirely or break out of the fetch cycle.

Procedure: How to Cancel Long Requests

1. From the Reporting Server browser interface menu bar choose *Workspace*, *Java Services*, DEFAULT. Right-click on *DEFAULT* and select *Agents*.
2. In the Java Services Agents pane, highlight a row with the jscomid you wish to kill, right-click on it and select *Stop.*

Obtaining the Number of Rows Updated or Deleted

PASSRECS returns the number of rows affected by a successfully executed SQL Passthru INSERT, UPDATE, or DELETE command.

Tip: You can change this setting manually or from the Reporting Server browser interface by clicking *Get Data* on the menu bar, right-clicking a configured adapter, and choosing *Change Setting*s from the shortcut menu. The Change Settings pane opens.

Syntax: How to Obtain the Number of Rows Updated or Deleted

```
ENGINE SQLPSTGR SET PASSRECS {ON|OFF}
```

where:

`SQLPSTGR`

Indicates the adapter. You can omit this value if you previously issued the SET SQLENGINE command.

`ON`

Provides the number of rows affected in the application program SCB count member after the successful execution of an SQL Passthru INSERT, UPDATE, or DELETE command. ON is the default value.

`OFF`

Provides no information after the successful execution of an SQL Passthru INSERT, UPDATE, or DELETE command.

PostgreSQL Optimization Settings

Adapter optimization allows the RDBMS to perform the work for which it is best suited, reducing the volume of RDBMS-to-server communication and improving response time. It also enables the RDBMS to exploit its own internal optimization techniques.

Chapter 12

Using the Adapter for Query/400

The Adapter for Query/400 enables you to create synonyms that Web Query can use for reporting against a Query/400 query definition.

The adapter generates metadata from an existing Query/400 object in such a way that only the required fields and ancillary information is retained for use as a basis for requests or queries.

This Adapter for Query/400 is intended to be used in conjunction with the Web Query client product.

In this chapter:

Configuring the Adapter for Query/400

In the Web Query environment, the adapter is pre-configured. No additional configuration steps are necessary.

Managing Query/400 Metadata

When the server accesses a data source, it needs to know how to interpret the data stored there. For each data source the server will access, you create a synonym that describes the structure of the data source and the server mapping of an existing Query/400 request and its underlying tables, columns, and associated data types.

Creating Synonyms

Synonyms define unique names (or aliases) for each Query/400 request that is accessible from the server. Synonyms are useful because they hide the underlying data source location and identity from client applications. They also provide support for extended metadata features of the server, such as virtual fields and additional security mechanisms.

Using synonyms allows an object to be moved or renamed while allowing client applications to continue functioning without modification. The only modification required is a redefinition of the synonym on the server. The result of creating a synonym is a Master File and an Access File, which represent the server metadata.

Procedure: How to Create a Synonym From the Web Query Environment

1. Expand the *Domains* folder, then expand a domain.
2. Expand the *Reports* folder, right-click a subfolder and choose *Metadata* from the menu.
3. In the left-hand Adapter navigation pane, right-click *Query/400* and choose *Create Synonym* from the menu. The first of a series of synonym creation pages opens.
4. Enter values for the parameters required for the adapter.

   For information about these parameters, see *Synonym Creation Parameters for Query/400* on page 209.
5. Click *Create Synonym*.

   The synonym is stored in the baseapp directory. You can use any of the available reporting tools to create a report using the generated synonym.

Procedure: How to Create a Synonym

1. From the Reporting Server browser interface Application page, click *Get Data*.
2. On the Configured Adapters section of the page, in Simple Mode, right-click an adapter and click *Show Connections*. Right-click a connection.

   Depending on the type of adapter you choose, one of the following options appears on the context menu.

   - **Show DBMS objects.** This option opens the page for selecting synonym objects and properties.
   - **Create metadata objects.** This option opens the page for selecting synonym objects and properties.
   - **Show files.** This option opens a file picker. After you choose a file of the correct type, the page for selecting synonym objects and properties opens.
   - **Show local files.** This option opens a file picker. After you choose a file of the correct type, the page for selecting synonym objects and properties opens.
   - **Show topics.** This option opens the page for selecting synonym objects and properties for topics within the environment.
3. Enter values for the parameters required by the adapter as described in the chapter for your adapter.
4. After entering the parameter values, click *Add*.

   This button may be labeled *Next*, *Create Synonym*, *Create Base Synonyms*, *Create Cluster Synonym*, or *Update Base Synonyms*.

The synonym creation process for most adapters has been consolidated so that you can enter all necessary parameters on one page. However, for some adapters such as LDAP, continue clicking *Next* until you get to a page that has a *Create Synonym* button.

The synonym is created and added under the specified application directory.

Note: When creating a synonym, if you select the Validate check box, where available, the server adjusts special characters and checks for reserved words.

Reference: Synonym Creation Parameters for Query/400

The following list describes the parameters for which you will need to supply values, and related tasks you will need to complete in order to create a synonym for the adapter. These options may appear on multiple panes. To advance from pane to pane, click the buttons provided, ending with the *Create Synonym* button, which generates the synonym based on your entries.

Library Name

Supply the name of the library in which the query definitions reside. (Wildcards are not permitted.)

Note: When you create a synonym for Query/400 on the IBM i platform, standard IBM i naming conventions apply to the target data source. Therefore, the Adapter for Query/400 supports the use of double-quotation marks around any library name and/or file name that contains lower case or NLS characters.

Click *Submit* to continue.

Application

The default value is baseapp.

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 the prefix and suffix fields blank.

Overwrite Existing Synonyms

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.

Do not save library information

Select this check box if you do not wish to save the library of the stored query in the metadata.

This selection ensures that the specific execution is based on the current library path of the connected user at run time.

Note that the library referred to here is *not* the library specification of the files used within the query, but rather where the query itself is located.

Select objects for synonym creation

To select all names in the list, select the check box to the left of the *Default Synonym Name* column heading. (If the library contains a large number of query definitions, this check box will not appear, however, synonyms will be created for all definitions automatically.)

To select specific names, click the corresponding check boxes.

Not all listed objects are data-related. Be sure to select those that are appropriate for creating synonyms.

Note: Mixed case names or names with NLS character will appear in double quotation marks. However, the double quotation marks will be replaced by underscore characters in the Default Synonym Name column (see below).

Default Synonym Name

This column displays the name that will be assigned to each synonym. To assign a different name, replace the displayed value.

Note: If you see a synonym name surrounded by underscore characters (as described in the previous note), you can remove the underscores if you wish.

Reference: Managing Synonyms

Once you have created a synonym, you can right-click the synonym name in the Adapter navigation pane to access the available options.

SOME TIBCO SOFTWARE EMBEDS OR BUNDLES OTHER TIBCO SOFTWARE. USE OF SUCH EMBEDDED OR BUNDLED TIBCO SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED TIBCO SOFTWARE. THE EMBEDDED OR BUNDLED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER TIBCO SOFTWARE OR FOR ANY OTHER PURPOSE.

USE OF TIBCO SOFTWARE AND THIS DOCUMENT IS SUBJECT TO THE TERMS AND CONDITIONS OF A LICENSE AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED SOFTWARE LICENSE AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER LICENSE AGREEMENT WHICH IS DISPLAYED DURING DOWNLOAD OR INSTALLATION OF THE SOFTWARE (AND WHICH IS DUPLICATED IN THE LICENSE FILE) OR IF THERE IS NO SUCH SOFTWARE LICENSE AGREEMENT OR CLICKWRAP END USER LICENSE AGREEMENT, THE LICENSE(S) LOCATED IN THE "LICENSE" FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

TIBCO, the TIBCO logo, the TIBCO O logo, ibi, ibi logo, ActiveMatrix BusinessWorks, TIBCO Administrator, BusinessConnect, TIBCO Designer, Enterprise Message Service, Hawk, and Maporama are either registered trademarks or trademarks of Cloud Software Group, Inc. in the United States and/or other countries.

Java and all Java based trademarks and logos are trademarks or registered trademarks of Oracle Corporation and/or its affiliates.

This document includes fonts that are licensed under the SIL Open Font License, Version 1.1, which is available at: https://scripts.sil.org/OFL

Copyright (c) Paul D. Hunt, with Reserved Font Name Source Sans Pro and Source Code Pro.

All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only.

This software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. See the readme file for the availability of this software version on a specific operating system platform.

THIS DOCUMENT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.