Sterling Integrator®

Reporting Services

Version 5.1

Sterling Commerce
An IBM Company

Contents

Reporting Services Overview

Reporting Services is used with your application and with MESA Developer Studio to define and see your data in ways that support your business.

Reporting Services uses Eclipse IDE plug-ins, services, maps, and business processes in the application, and MESA Developer Studio to produce the business level information you need from raw data that flows through the application.

Using the Fact Model Editor Eclipse plug-in, you identify which data to capture from the application and how to structure and store the data. XML schemas are created that you can use with the application to view the data as reports, or you can export and use them with third party software.

In the application, you can view automatically generated reports for your data and define new reports. Using two other Reporting Services Eclipse plug-ins-the Report Editor and Report Format Editor-you can edit and copy reports and change some formatting elements.

Reporting Services, MESA Developer Studio, and the application are all purchased and licensed separately. For information about purchasing these, contact Sterling Commerce Customer Support or your Sterling Commerce Sales Representative.

Implementing Reporting Services

Implementing Reporting Services involves the following phases:

- Planning

  During this phase, you decide what information you want to ultimately see. What are the questions that Reporting Services needs to answer? What data do you want to see, and in what format? Do you need sales by year, month, and day? Do you need sales by region, salesperson, or product? Are you interested in seeing revenue vs. expenses by month, by division, and by department? Once you decide on all the questions to be answered, you can start to define the information.

- Defining and Building Data

  In Reporting Services, data is defined and built using the Fact Model Editor. You define the data and relationships that will enable you to view your business results and facts in the most useful manner for you.

- Extracting Data

  Once you define and build a fact model, you are ready to begin pulling the data you need from the datastream. By creating business processes that include the Mapped Extraction service, you are able to extract your data into a Reporting Services database called the *fact repository* that is created for your fact model.

- Viewing Data

  As data is extracted into the fact repository, you can begin reviewing the default reports that are created automatically for your fact model in the application, to edit those reports, or use them as templates for new reports. If you use JPivot and Mondrian, you can export the Mondrian schema that is automatically created for your fact model and view your data as a multi-dimensional cube.

Reporting Services Components

Reporting Services uses the following components:

- Reporting Services Fact Repository – Reporting Services pulls facts from your data stream (systems, databases, business processes) and stores it in a database called the Reporting Services Fact Repository. This database should be separate from the application database.
- Eclipse plug-ins that are accessed through the MESA Developer Studio:
  - Fact Model Editor – Use to identify the data items to be captured by the system and the structure for storing and presenting the data.
  - Report Editor – Use to edit the two reports created by default for each fact model (Detail and Summary) and to create new reports based on your fact models.
  - Format Editor – Use to define the appearance of your reports. A default report format is included with the Format Editor. You can create additional formats as needed.

Reporting Services also uses these application features:

- Schema check-in and check-out – Fact models and its associated components (reports, Mondrian cubes) are checked in to the application using the Schema check in function.
- Reporting – Reports created for your fact models are accessed through the application Reporting function.
- Map Editor – You can use one or more any-to-JDBC or any-to-XML maps to extract data to the fact repository.
- Business Processes and Services – Reporting Services includes two services: the Mapped Extraction service and the Straight Through Extraction service. Each is used in business processes to extract data and store it in the fact repository.
- Events – application events can be used to trigger updates to the fact repository. The Systems and Operations fact model included with Reporting Services makes extensive use of the system and application operational events to gather its data.
- Event Forward Listener –This allows for specific events to be listened for and redirected to Reporting Services. The events can be existing system events or custom events thrown from custom services.

Key Terms Used With Reporting Services

Term	Definition
Cube, Multi-Dimensional Cube, Multidimensional database	A data structure that has at least three dimensions. It is the fundamental structure for information in an OLAP (On-Line Analytical Processing) system.
Dimension	A collection of data of the same type. For example, “Time” could be a dimension.
Fact	A single piece of raw data.
Fact Meta Data	Data that describes the fact. Examples are data types and acceptable values, mappings to the data source, descriptions of any transformation of the data, and any information needed to support the fact model.
Fact Name	The reference name for a specific data field. This referenced name must be used consistently to ensure accurate reporting. For example, using the same data name for two different types of data could cause an unintentional mix of data.
Fact Model	A named, versioned application resource that contains a group of fact sets, meta data about the named fact, and measures, dimensions, and links for the data.
Fact Pattern	Defines a parent/child relationship between two fact sets.
Fact Set	A group of logically categorized facts.
Fact Repository	Specified data storage area.
Hierarchy	Standard tree view (parent with one or many children) organization for positions in a dimension.
JPivot	A third-party, custom JSP (Java Server Pages) tag library that allows users to perform common OLAP (Mondrian and XMLA) tasks.
Level	Within a dimension, facts may have various positions for detail. For example, a time dimension may have levels for minute (level 1), hour (level 2), day (level 3), etc.
Listener	Reporting Services uses listeners to wait (listen) for specified events to take place which have been defined as triggers for data extraction. The listeners used are installed with the application.
Measure	A numeric fact or result of a calculation.
Mondrian	Mondrian is an OLAP engine written in Java. Reporting Services automatically creates Mondrian schemas for any fact models checked in to the application. You can then use these schemas with third-party tools like JPivot.

Term	Definition
OLAP	Online Analytical Processing systems used to analyze large quantities of data in real time.
Retention	The amount of time defined to keep data in the fact repository.
Star Schema	A data structure that is has a central fact set that has pointers from other facts.
XMLA	Open industry-standard web service interface designed specifically for online analytical processing (OLAP) and data-mining functions.

Reporting Services Planning

Before you begin creating a fact model, you need to fully understand the domain that the fact model will represent. For example, some banking domains might be currency exchange, credit card, or consumer banking.

Get the Right Information by Asking the Right Questions

Begin by asking the following questions:

- What information do you want to see?
- How often does the information need to be updated - on demand or on a schedule, or both?
- Who will access the information and how often?
- How should the information be broken down: by time increments (monthly, yearly)?
- What other measurements are needed (by salesperson, department, company, product line, etc.)?
- What raw data is needed to get the information?
- Where does the raw data come from?
- What format is the raw data in?
- Which pieces of data will need to be combined, and how, to get to the final form of the information you want to see?

By answering these questions, you can come up with a blueprint for your Reporting Services installation.

What is a Fact?

The lowest level of information that you will need to define in Reporting Services is a *fact*. A fact is a single piece of raw data. For example, a fact for sales data might be a product ID or a price, or a salesperson’s name. A fact for expenses might be a type of utility, such as electric or water. Or it could be an item that your company purchases, such as a box of paper. Or, it could be the salary per pay period for an employee.

Define facts for only the items you really need to answer your planning questions. The saying “More is better” does not apply to fact models. The goal is to have exactly the facts you need and no more. Extra information will only slow processing and report generation, and clog the fact repository with unnecessary data.

Grouping Facts and Defining Relationships

Determine how the raw data should be assembled to achieve the desired end result. Once you have decided on the facts, you will determine what calculations must be performed and what relationships between facts must be defined.

In Reporting Services, you can define:

- Fact sets – Groups of related facts

- Measures – Performs calculations such as sums and counts on facts
- Dimensions – Defines a hierarchical relationship among facts
- Fact Patterns – Identifies a primary fact set in a fact model and explicitly links it to one or more child fact sets

Fact Model Design Affects How Your Data is Presented

It is essential to spend time planning your fact model. Usually, there are multiple ways to build a fact model. Your planning objective is to decide how to build the fact model so that it enables you to get the reports you need. There can be different ways to define facts and relationships in a fact model; however, when building the fact model, always consider how the arrangement of facts and the way they are grouped will affect the reports.

Default reports are generated for each fact model when it is checked in to the application. To provide the most flexibility, the detail reports contain all possible options. You can edit the reports in the Report Editor to limit or change which data is included.

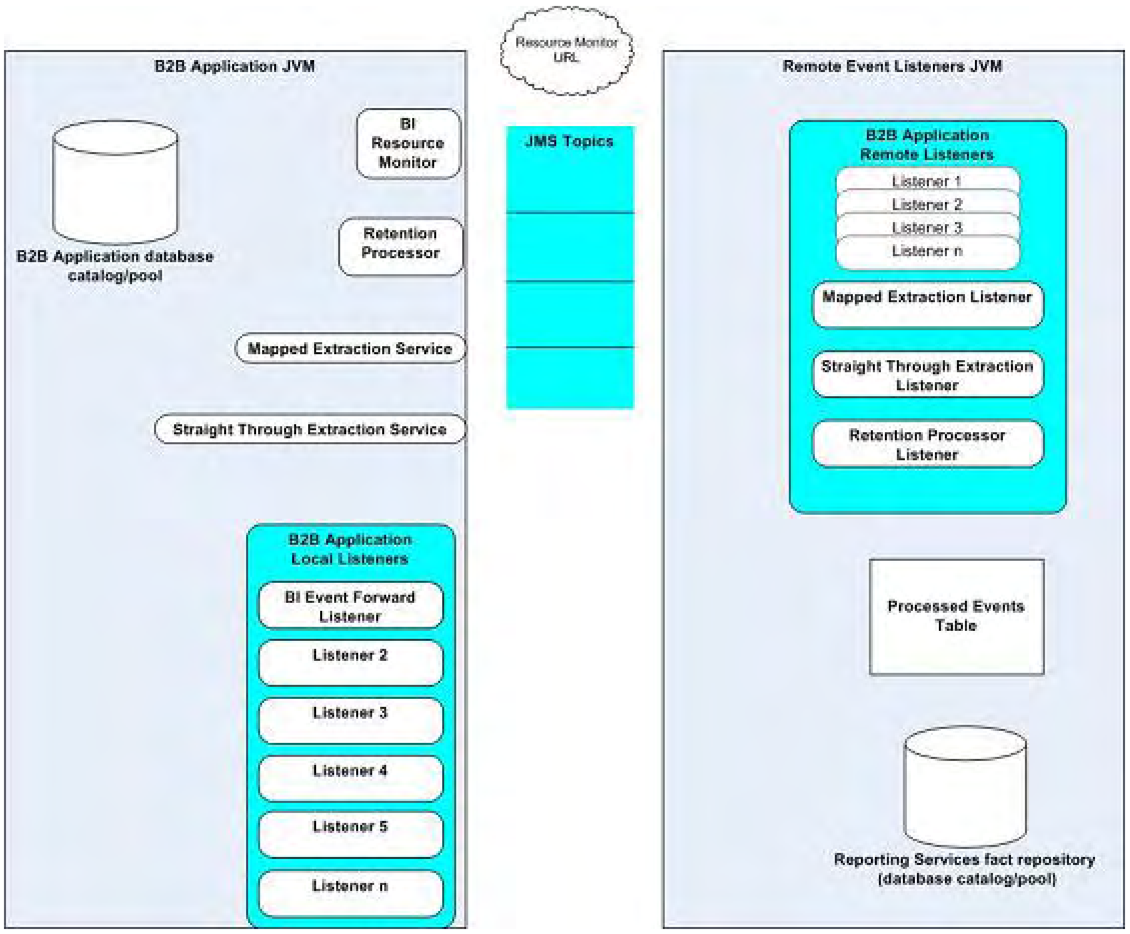
Reports are created as follows:

- A default detail and a summary report are created for each fact set that is not joined to any other fact set in the fact model, if a measure was defined.
- If a fact set is used in a fact pattern, a detail report and a summary report are created for each grouping where all the fact sets in the fact pattern are present in the report. This may produce a detail report with a large number of columns that you can customize.
- If a fact set does not have a grouping, a detail and summary NoGroup reports are generated.
- Each level creates a group report that is grouped by the last column in the fact path. That is, if you entered BookInfo/ISBN as the fact path for a level, the group report would be grouped by ISBN.
- Measures are used to create summary and group footer sections of the reports.

How Reporting Services Works

The repository is populated with data by a series of actions starting from an event taking place in the application.

To understand how Reporting Services works, it will help to understand the architecture first:



Resource Monitor URL
B2B Application JVM
BI Resource Monitor
Retention Processor
B2B Application database catalog/pool
Mapped Extraction Service
Straight Through Extraction Service
B2B Application Local Listeners
BI Event Forward Listener
Listener 2
Listener 3
Listener 4
Listener 5
Listener n
JMS Topics
Remote Event Listeners JVM
B2B Application Remote Listeners
Listener 1
Listener 2
Listener 3
Listener n
Mapped Extraction Listener
Straight Through Extraction Listener
Retention Processor Listener
Processed Events Table
Reporting Services fact repository (database catalog/pool)

Reporting Services uses remote event listeners to communicate with the application. These remote listeners reside in a separate JVM from the application called the Remote Event Listeners JVM. The two JVMs (application and Remote Event Listeners) pass messages and acknowledgements through JMS topics.

Information for Reporting Services is passed from the application JVM to the Remote Event Listeners JVM by several methods:

- Straight Through Extraction service – designed to pass data that is already in the correct format for the database schema from the process data in a business process to the fact repository. In the Remote Event Listener JVM, events passed from this service are picked up by the Straight Through Extraction Listener and passed to the fact repository.
- Mapped Extraction service – designed to pass data from the application datastream through a map, which formats the data correctly for the fact repository. This service can also be used in passThrough mode if the data is already in the correct format. In the Remote Event Listener JVM, data is picked up by the Mapped Extraction listener and passed to the fact repository.
- Retention Processor Listener – determines when Reporting Services data gets purged. The information can come from the system's default retention schedule or the fact model's retention schedule. By default, the listener is run once daily as part of a scheduled business process (MESAVisToolkit_RetentionProcessor.bp).
- Resource Monitor – Checks at a set interval to see if the Remote Event Listener JVM is up and running.
- Event Forward Listener - Sends events to the fact repository. By default, some system events are automatically sent to the fact repository (see System Fact Models for more information). You can add other events by un-commenting events in this property file. Note that the data from the event must be in the correct format for the schema. The event is picked up by the Remote Event Listener and passed to the fact repository.

Reporting Services Event Handling

Reporting Services has “guaranteed delivery” of data - that is, data is not lost if there is a problem with the JVM, for example, or with a map, document, or database table. The following points give more detail about various aspects of event handling.

- Events are processed as “all or nothing.” That is, if the system goes down during processing, no part of the event transaction gets committed-everything is rolled back and the event goes to an errorQueue.
- The JMS topic used between the application JVM and the Remote Event Listeners JVM are persisted by default.
- Reporting Services Exception Notifications: When various exceptions occur, such as a missing map, document or table, the exception is logged and an e-mail is sent to the user specified in the mailTo field in the bi.properties file. The event then gets sent to an error queue in the JMS topics.
- The resource monitor pings Reporting Services on a set interval and if it does not get a response from the Reporting Services JVM in a set amount of time (default is 10 minutes), it sends an email to the user specified in the bi.properties file.

  The amount of e-mail is controlled by the emailThrottle field in the bi.properties file. All e-mails from Reporting Services are subject to the throttle to prevent spamming. If the threshold is met and the throttle is invoked, an e-mail is sent to notify the user how many emails were not sent, but did result in log entries.
- To use a separate database or connection pool for Reporting Services, you must deploy the fact models on it. This is to ensure that the Processed Events table is available on each database or pool for duplication event detection.

Reporting Services Listeners

Reporting Services uses the following listeners to extract data and populate the fact repository:

- Straight Through Extraction - Simple one-to-one extractions. Can only handle a single thread at one time. Data must be in process data, and in the correct format for the fact repository table.
- Mapped Extraction - Structured extractions based on maps. Can handle multiple threads. Can be used in passThrough mode, which is similar to Straight Through extraction, but is multi-threaded. passThrough mode does not require a map, but the data must be in the correct format for the fact repository table. Data must be in the primary document for both mapped and passThrough modes.
- Event Forwarding Listener - Converts existing events to Straight Through events. Data must be in the correct format for the fact repository table.

Additional Information for Testing Listeners

By default, the event listeners are automatically started with the application on all platforms - you do not need to manually start or stop them. However, when testing, you may want manually control the listeners or see additional debugging messages.

UNIX Environment

To start the listeners, run the startListeners.sh command from the bin folder under your application install folder.

There are two optional parameters:

- debug - turns on additional debugging messages.
- <logfilename> - sends all messages to the specified logfile name.

You can use one or both of the parameters, in any order. If the debug parameter is used, the system turns on those messages. If debug is not used, the system assumes it’s a logfile name. The following is an example of the Start Listeners command for UNIX:

```
startListeners.sh debug <bi.log>
```

To stop the listeners, run the stopListeners.sh command from the bin folder under your application install folder. The following is an example of the Stop Listeners command for UNIX: stopListeners.sh

Windows Environment

When using the application on a Windows system, the event listeners are controlled by Windows services.

- To start the listeners, run the startEventListenersWindowsService.cmd script from the *install_dir*\bin folder.
- To stop the listeners, run the stopEventListenersWindowsService.cmd script from the *install_dir*\bin folder.

Properties Files Used By Reporting Services

The following property files are used in conjunction with Reporting Services. These are located with the rest of the application properties files, in the [*install_dir*]/install/properties folder, except where noted otherwise:

- startlisteners.properties.in - Contains information about JMS topics and other configuration information for the listeners used by Reporting Services. This must be directly edited; you cannot use customer_overrides.properties for changes to this file.
- bi.properties.in - Has new email properties for notifications.
- event.properties.in - Creates an error queue and its administrative scripts. Time-to-live for remote events/messages is infinite (same for error queue).
- event forward listener property file - Used to send data from system events to the fact repository. All options in the property file are commented out by default. To have data from any of the events passed to the fact repository, you must edit the file and un-comment the desired events. You must create a fact model and check it in to the application. This creates the schema for Reporting Services to use when collecting and passing the event data. Also, if your data is coming from a business process, turn on event notification for the business process. This is done by setting the Event Reporting Level parameter during business process check in.

Reporting Services Database Considerations

The fact repository can be any of the databases supported for the application on a particular platform.

When planning your Reporting Services installation, decide how you want to set up your fact models and databases. The following guidelines can help you find the safest and most efficient methodology for your data and environment:

Databases:

- Consider whether or not the database you use for the application is the database best suited to Reporting Services usage, as well. By default, Reporting Services will build the repository in the same database as the application. Some databases may have restrictions, such as keyword or column restrictions, so bear these in mind when selecting a database to use. For example, manual updating of the table structure may be required for the Date field for custom fact models installed using DB2, which does not accept NULL values.

Catalogs and Pools:

- If using multiple fact models that will have the same fact set names, you must use different database pools for each fact model.
- Even if the fact models will not use the same fact sets, other reasons, such as security, might make using separate database pools for each fact repository a better solution.

It is important to consider the database layout when designing your fact models; the parts of the fact model directly relate to the parts of the database, as follows:

- Database name = name of fact repository
- Table name = Fact set name
- Column name = Fact name
- Max columns per table = Max facts in a table
- Max table row length = The total table row length cannot exceed the length shown in the following table

The limitations for each database are shown in the following table:

	Database name max length (bytes)	Table name max length (bytes)	Column name max length (bytes)	Max columns per table	Max table row length (bytes)
DB2 8.1	8	128	128	1012	32677
MySQL 4.1	64	64	64	2599	2GB
Oracle 9I	8	30	30	1000	2000

	Database name max length (bytes)	**Table name max length (bytes)**	**Column name max length (bytes)**	**Max columns per table**	**Max table row length (bytes)**
Oracle 10g	8	30	30	1000	2000
MSSQL/SQLServer 2000/2005	128	128	128	1024	8060
Informix 7.30C1		18	18	994	32356

Specifying a Different Database or Pool for Reporting Services

By default, Reporting Services uses the same database and pool for its fact repository as your application system uses for its database. However, if you are using multiple fact models that will contain the same fact set names, you must set up different database pools for each fact model/fact repository.

To use a different database pool:

1. Open the customer_overrides.properties file, which is located in the [*install_dir*]/install/properties folder on your application system.
2. Add the following line, substituting the name of the pool to be used for the italicized example text:

   bi.poolName=*mysamplePool*
3. Add a JDBC entry for the database pool you defined in step 2.
4. Save the file.

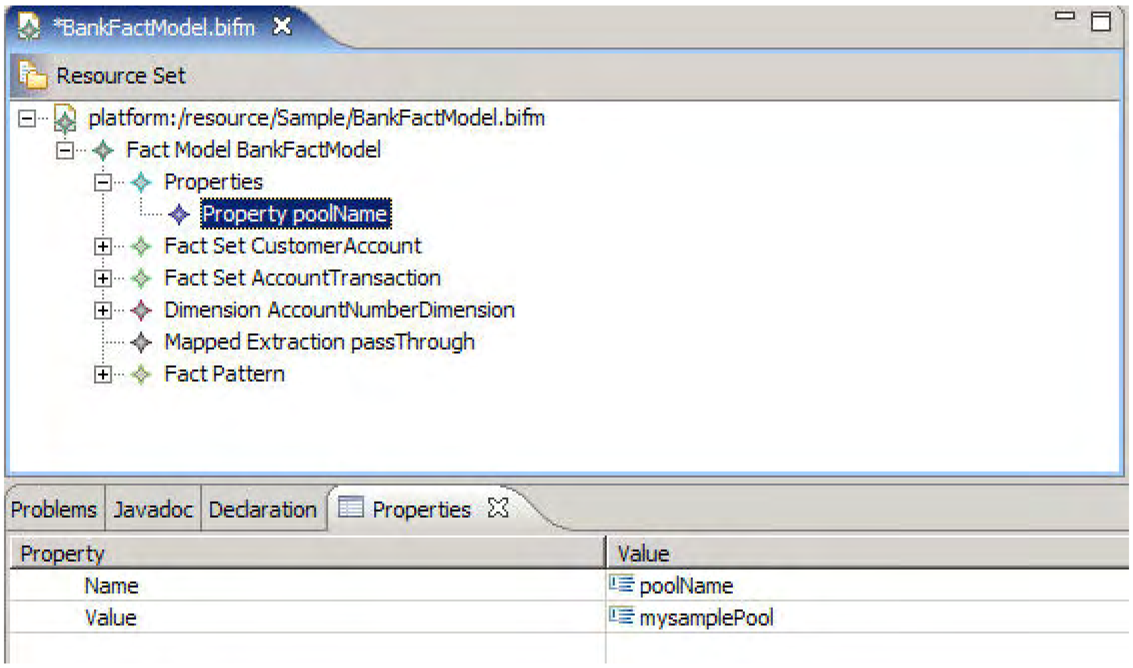
   Note: See *Sample Customer Overrides Properties File* for examples of entries to the customer_overrides.properties file.
5. Open startlisteners.properties.in, which is located in the [*install_dir*]/install/properties folder on your application system.
6. In the file, locate the ifdef statement for your environment. Add lines for each vendor jar file that you will use in the section for the ifdef statement. For example:

   ```
   #:ifdef WINDOWS
   VENDORJAR_DB=&JAR_JAVA_HOME;/jre/lib/ext/&JDBC_DRIVER;
   #:else
   VENDORJAR_DB=&JDBC_DRIVER;
   setSystemProperty@java.io.tmpdir=&INSTALL_DIR;/tmp
   #:endif
   ```

   Note: You can choose any names for the entries, but each name must be unique. They do not have to share the same prefix (VENDORJAR_DB, for example), but using this type of convention may make it easier to locate the entries in the file.
7. Save and make a backup copy of the file.

   Caution: If you apply a new application patch or update, this file will be overwritten. Keep a backup copy so that you do not have to recreate all of your changes after a product update. You can add your changes to the new version of the startlisteners.properties.in file after running an update.

8. Install the jar files for each vendor using the install3rdParty script in the *install_dir*]/install/bin folder on your application system.
9. In your fact model, add a Properties element. Under the Properties, add a Property child element. In the Name field, enter poolName. In the value field, enter the name of the pool that you referenced in the properties files. For example:



Property	Value
Name	poolName
Value	mysamplePool

10. Validate the fact model (either validate just the new properties element, or validate the entire fact model from the root element), save it, and check in the fact model to the application.
11. Run setupfiles.sh (UNIX) or setupfiles.cmd (Windows) to apply the properties changes.
12. Stop and restart the application.
13. Restart the event listeners.

How Reporting Services Determines Which Database Pool to Use

Reporting Services will use the default database pool used by the application, unless you have specified a different pool name in another location. Reporting Services checks for and uses database pools in this order of precedence:

1. First, Reporting Services checks for a pool name specified in the fact model itself (as a Properties/Property element).
2. If no pool name is in the fact model, it checks for a pool name in the customer_overrides.properties file (as an override value for bi.properties).
3. If no pool name is specified in either of the first two locations, it checks for and uses the pool name specified in the bi.properties file.
4. Last, if no pool name is specified in any of the first three locations, it will use the pool name specified in the jdbc.properties file.

Sample Customer Overrides Properties File

The following example entries for a customer_overrides.properties file show two pools specified for use with fact repositories.

Note: Only one pool can be specified as the Reporting Services pool (bi.poolName). Only reports for the fact repository that uses the Reporting Services pool can be scheduled; any others must be run manually.

In the example, italicized text indicates that this is an item which requires a value for your system.

```
bi.poolName=myPool
jdbcServiceCustomer.oraclePool.driver=oracle.jdbc.driver.OracleDriver
jdbcServiceCustomer.oraclePool.url=jdbc:oracle:thin:@myoracleserver:0000:mycatalog
jdbcServiceCustomer.oraclePool.user=userID
jdbcServiceCustomer.oraclePool.password=password
jdbcServiceCustomer.oraclePool.prop_TCP.NODELAY=YES
jdbcServiceCustomer.oraclePool.maxconn=30
jdbcServiceCustomer.oraclePool.catalog=mycatalog
jdbcServiceCustomer.oraclePool.schema=myschema
jdbcServiceCustomer.oraclePool.type=remote
jdbcServiceCustomer.oraclePool.dbvendor=oracle
jdbcServiceCustomer.oraclePool.buffersize=500
jdbcServiceCustomer.oraclePool.maxsize=28
jdbcServiceCustomer.oraclePool.initsize=1
jdbcServiceCustomer.oraclePool.factory=com.sterlingcommerce.woodstock.util.frame.jdbc.ConnectionFactory
jdbcServiceCustomer.oraclePool.behaviour=2
jdbcServiceCustomer.oraclePool.waittime=1000
jdbcServiceCustomer.myPool.driver=com.mysql.jdbc.Driver
jdbcServiceCustomer.myPool.url=

jdbc:mysql://myserver:00000/anothercatalog?useUnicode=true&characterEncoding=UTF-8
jdbcServiceCustomer.myPool.user=userID
jdbcServiceCustomer.myPool.password=password
jdbcServiceCustomer.myPool.maxconn=20
jdbcServiceCustomer.myPool.catalog=anothercatalog
jdbcServiceCustomer.myPool.dbvendor=mysql
jdbcServiceCustomer.myPool.buffersize=500
jdbcServiceCustomer.myPool.maxsize=28
jdbcServiceCustomer.myPool.initsize=1
jdbcServiceCustomer.myPool.factory=
   com.sterlingcommerce.woodstock.util.frame.jdbc.ConnectionFactory
jdbcServiceCustomer.myPool.behaviour=2
jdbcServiceCustomer.myPool.waittime=1000
```

Reporting Services Security Considerations

Before you begin using Reporting Services, decide who will have access to the Fact Model Editor and other Eclipse plug-ins, as well as to the other parts of the application: schema, report, service, map, and business processing functionality. Decide where the fact repository will be located.

Modeling the Data for Reporting Services

In Reporting Services, defining your data is done using fact models and mapping. A *fact model* is a named, versioned application resource that contains facts, metadata about the facts, and measures, dimensions, and links for the data. All of the information that you ultimately have access to in reports is derived from the fact model. You use the Fact Model Editor to create fact models.

To get data into the fact model, the Fact Model Editor pulls facts directly from a datastream using a mapped or straight through extraction.

Fact Models Included in Reporting Services

Reporting Services comes with three fully-operational fact models and report sets for application systems and operations data. You can check out the schemas for the fact models and open them in the Fact Model Editor in Eclipse. Do not change these fact models. If you want to make changes, it is strongly recommended that you copy the fact model and make changes to the copy. Changing the original system and operational fact models can cause unpredictable results.

The included fact models are:

- system.bifm – Tracks system log information
- BIFactModel.bifm – collects data from processed events.
- WFFactModel.bifm – Tracks detailed information about business process events in the application.

Example Bank Fact Model

Although not installed with Reporting Services, the BankFactModel (BankFactModel.bifm) sample is available in the [*Install_dir*]/install/bi/samples folder. This fact model is intended for informational purposes only.

Building a Fact Model

Before you create the fact model, you must create an Eclipse project in which to store it.

Note: For fact models, it is recommended that you use the Eclipse Modeling Framework project type.

To create an Eclipse project and a fact model:

1. Select **File > New > Project**.
2. On the New Project wizard, select the type of project you want to create: **Eclipse Modeling Framework > Empty EMF Project**.
3. Enter a name for the project and click **Finish**. You can now build the fact model.
4. In the Eclipse Package Explorer, select the project you just created.
5. Select **File > New > Other > Sterling Commerce Reporting Services > Fact Model**.
6. Enter a name for the fact model file. Use the default extension, .bifm. Click **Finish**.
7. Double-click the name of the fact model in the Package Explorer to open it in the Resource Set pane.
8. Select **Properties** from the tabs below the Resource Set pane.
9. Select the fact model line in the Resource Set pane and enter the name for the fact model in the Name property field in the Properties pane. This must be the same name that you entered in step 6 for the fact model filename (without the .bifm extension).

Defining a Fact Set

In the fact model, right-click the fact model line in the Resource Set pane.

1. Select **New Child > Fact Set**.
2. In the Properties pane, enter a name for the fact set.

   Add child elements to the fact set as needed. To add a child element, right-click the fact set and select New Child and then the type of element to create (Fact, Measure, Retention).

Defining a Dimension

Specifying dimensions for your fact model enable you to categorize your data. For example, you could create the dimension “time” and beneath this category, have levels for day, week, month, and year. Those levels can then be aggregated to define a hierarchical structure for the data - combining months into the yearly data, for example.

In the fact model, right-click the fact model line in the Resource Set pane.

1. Select **New Child > Dimension**.
2. In the Properties pane, enter a name for the dimension.

   Add only one hierarchy child element to the dimension. To add a hierarchy, right-click the dimension and select **New Child > Hierarchy**.

   Each hierarchy can contain multiple level elements. To add a level element, right-click the hierarchy and select **New Child > Level**.

Defining a Mapped Extraction

Mapped extractions enable you to manipulate or translate data prior to using it in Reporting Services. Before you can define a mapped extraction, you must create the translation map to be used by the extract. Either before or after defining the mapped extraction, you must create a Mapped Extraction service configuration and a business process that will extract the data and put it in the repository.

In the fact model, right-click the fact model line in the Resource Set pane.

1. Select **New Child > Mapped Extraction**.
2. In the Properties pane, enter a name for the mapped extraction.
3. Click the ellipsis (...) to the right of the Map Name field, which displays a map entry page.
4. Enter the name of the map to be used for this extraction and click **Add**. Repeat to add other maps, if necessary. Once you have added all maps, you can rearrange them using the Up and Down buttons.

   Note: All maps added to a single mapped extraction must be of the same type – either XML or DB (database).

Defining a Fact Pattern

The fact pattern is the primary fact set in a fact model, when using a star schema, or non-hierarchical fact model.

In the fact model, right-click the fact model line in the Resource Set pane.

1. Select **New Child > Fact Pattern**.
2. There are no fields for the fact pattern itself; however, you must add one root fact set element to the fact pattern. You can optionally add a retention element.

To add a root fact set, right-click the fact pattern and select **New Child > Root Fact Set**.

Each root fact set can contain one or more fact set ref elements. To add a fact set ref element, right-click the root fact set and select **New Child > Fact Set Ref**.

Defining Properties

Use properties to specify a different database to be used for the fact repository.

In the fact model, right-click the fact model line in the Resource Set pane.

1. Select **New Child > Properties**.
2. There are no fields for the Properties element itself; however, you must add at least one property element beneath it.

   To add a property, right-click the fact pattern and select **New Child > Property**.

Validating the Fact Model

You can manually validate the fact model or any individual components in the Fact Model Editor. The Fact Model Editor validation checks for well-formedness and does some datatype validation.

When you check in a fact model, report definition, or report format to the application, it validates the structure and content of the of the component against the schema and does internal consistency checks.

When you make changes to the fact model, you should validate it at the root level.

1. To validate a single element, select the element and click **Fact Model Editor > Validate** from the drop down menus. You can validate elements at any level in the fact model. For example, you can select and validate a fact set and its child elements, or a fact itself.
2. To validate the entire fact model, select the root fact model element and click **Fact Model Editor > Validate** from the drop down menus.

   If the selected element is valid, the message “Validation completed successfully” displays. If the selected element is not valid, an error message is displayed. Click Details to see more specific information about why validation failed.

Extracting Reporting Services Data

Data extraction is selectively pulling information (based on your fact model) and using it to populate the Reporting Services fact repository. Once you have set up your fact model and checked it in, you are ready to set up data extraction.

There are two types of data extractions used: mapped extractions and straight through extractions.

Mapped Extraction

A mapped extraction enables you to use a map to obtain data from a business process.

Using the Map Editor, create a map to specify what data to pull from the datastream and how to transform or translate the data. Add the Mapped Extraction service to the business process that you will use to collect the data. The translation engine then transforms or translates the data according to your map, and the converted data is moved to and stored in the fact repository.

If your data is already in the correct format for the fact repository (the schema format), you can use the "passThrough" option, which does no data translation. Using a mapped extraction element in the fact model, enter "passThrough" as the name of the mapped extraction and leave the Map Name field blank. Leave the output type at default.

Some considerations for mapped extractions are:

- A fact model can contain more than one mapped extraction element.
- A mapped extraction can refer to more than one map.
- A single mapped extraction can contain only maps of the same type, either XML or DB. To have a fact model use both XML and DB maps, create two separate mapped extraction elements, and include the XML maps in one and the DB maps in the other. It is currently recommended that you use XML mapped extractions, unless you have a great deal of expertise in creating and using databases in general, and creating and using database maps in the Map Editor.
- Transactionality across mapped extraction processing is only supported for XML maps, not for database maps. That is, if the system goes down during translation processing of an XML map or any other processing of the mapped extraction, no part of the extraction transaction gets committed—everything is rolled back and the event goes to an errorQueue.

Straight Through Extraction

Data is already in the correct format for the data repository and is extracted and placed directly in the fact repository with no transformation or translation. Data is collected from process data using the Straight Through Extraction service in a business process. Identify the data you want by specifying the EventSchemaID in the service configuration in the GPM.

Mapped Extraction Setup Checklist

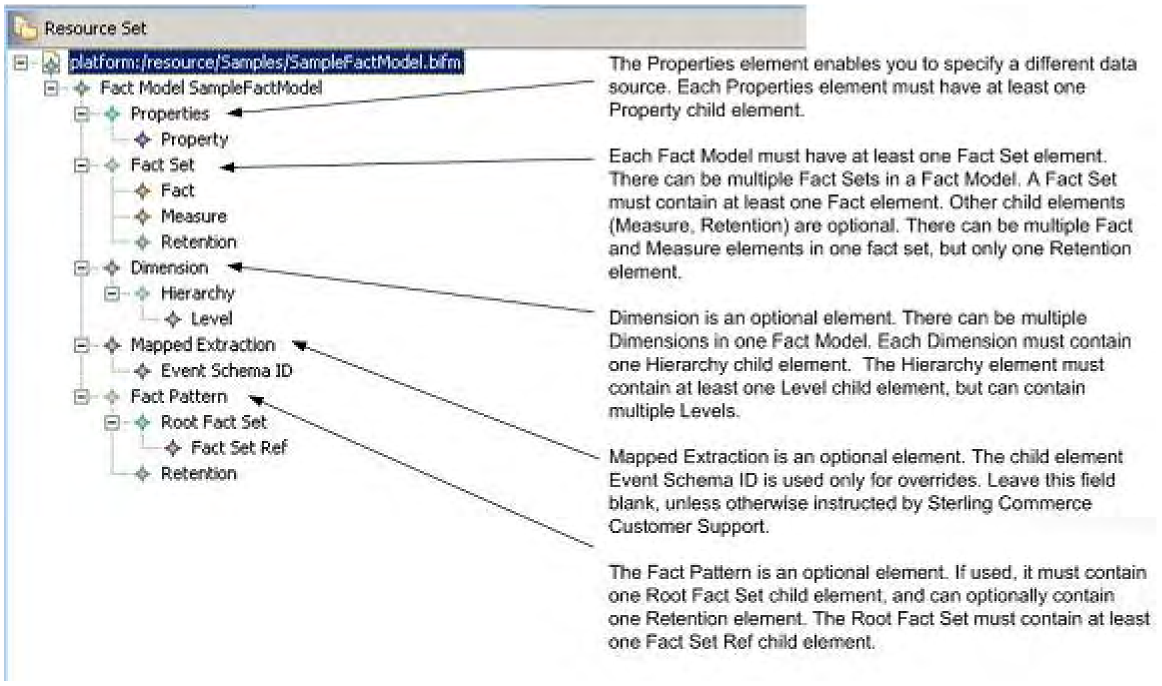
Extracting data has many steps and separate processes. It is recommended that you follow the order of tasks shown in the following table when setting up a mapped extraction.

Task	Description
Create the fact model	When creating your fact model, add the Mapped Extraction element. Decide on a map name now, and add it to the Mapped Extraction element. 1. Enter the map name here, for reference later: Map name: ____ 2. Enter the Mapped Extraction element name here, for future reference (needed when creating the business process that will extract the data): Mapped Extraction name: ____
Check in the fact model	Checking in the fact model creates an xsd file, which is the schema you will use to create the output format of your transformation map.
Check out the xsd file	Check out the default version of the xsd (you may have gone through multiple iterations of checking in the fact model and resetting the default each time). Ensure that the file is can be accessed by the application Map Editor.
Create a new map	Using the Map Editor, create a map. Name the map using the name that you entered in the Fact Model (see the Create the Fact Model step). When creating the map, follow these steps: • Use the map type **Sterling Integrator** • For Input format, select the format of your data (for example, delimited, variable, or XML) • For Output format, when using a map for Reporting Services, you must select **XML** or **SQL** (database connectivity). Only use SQL if your mapped extraction is to a database. It is recommended that you extract data to XML instead of to a database. If you choose **XML**, click **Customize**. Choose **Schema** as the Customization File Type, and browse to the xsd file you checked out of the application. Select the file, and continue with creating the map.
Check in the map to the application	Once you have validated and compiled the map, check in both the source (.mxl) and compiled (.txo) versions to the application.
Test the map	One way to test the map is to create a simple business process that contains just the Translation service. In the GPM, select the map to use for translation, and set the Output to Process Data parameter to **Yes**. This enables you to see the results of your translation right from the Business Process Execution Manager page. By doing this, you can isolate and fix any translation issues prior to using the map as part of Reporting Services.
Create a business process for Reporting Services	The business process that you create for data extraction must contain the Mapped Extraction service.

Task	Description
	Your final business process may be very complex, but for testing, create a business process that contains just the Mapped Extraction service. This enables you to verify that the data is being extracted to the fact repository correctly, and to view the reports. By limiting the number of steps in the business process, you can focus on the mapped extraction step. When you define the parameters for the Mapped Extraction service in the GPM: • Type the name of the fact model (without the .bifm extension) in the Fact Model field. • Type the name of the mapped extraction element in the fact model (see the Create the Fact Model step) **Note:** These fields are case-sensitive – ensure that the capitalization matches exactly the way the name appears in the Fact Model itself.
Check in the business process	Once you have validated and saved the business process, check in the business process to the application.
Create sample input files	Create a sample input file in the correct format for the input side of your map, and ensure it is available to your application system.
Execute the business process	Run the business process. On the Business Manager page, search for the business process using either the **Search** or **List** option. On the results page, click **Execution Manager**. On the Execution Manager page, click Execute. On the Execute page, type in or browse to the name of your sample input file, and click **Go!**
Check results by looking at reports	You can view your data by viewing the default summary and detail reports generated for each fact model. Go to **Operations > Reports** > and select **Create New Report Configuration** and click **Go!** Select the report you want from the list. The report names include the name of the associated fact model.
If necessary, make changes to the fact model.	If the data was not extracted to the fact repository correctly, review the bizintell.log to see if any errors were logged when the mapped extraction ran. Also, verify that the event listeners are running.

Fact Model Components

graphic   The following shows a fact model in the Reporting Services Fact Model Editor in Eclipse:



The Properties element enables you to specify a different data source. Each Properties element must have at least one Property child element.

Each Fact Model must have at least one Fact Set element. There can be multiple Fact Sets in a Fact Model. A Fact Set must contain at least one Fact element. Other child elements (Measure, Retention) are optional. There can be multiple Fact and Measure elements in one fact set, but only one Retention element.

Dimension is an optional element. There can be multiple Dimensions in one Fact Model. Each Dimension must contain one Hierarchy child element. The Hierarchy element must contain at least one Level child element, but can contain multiple Levels.

Mapped Extraction is an optional element. The child element Event Schema ID is used only for overrides. Leave this field blank, unless otherwise instructed by Sterling Commerce Customer Support.

The Fact Pattern is an optional element. If used, it must contain one Root Fact Set child element, and can optionally contain one Retention element. The Root Fact Set must contain at least one Fact Set Ref child element.

The following sections describe each element shown in the previous graphic:

Properties

Properties enable you to specify a different data source for your Reporting Services fact repository and can have the following child element:

Field	Description
Property	Enter the name of the data source to be used instead of the default. Required if Properties is used.

Properties Child Element: Property

The following table contains the fields that are included in a property element:

Field	Description
Name	Enter a name to be associated with this property in the fact model and schemas. For example, poolName.
Value	Enter the database to be used for the fact repository for this fact model.

Fact Set

A *fact set* is a group of logically categorized facts. A fact set can have the following child elements:

Field	Description
Fact	A single piece of raw data. At least one fact per fact set is required. A fact is the most basic, granular element of a fact model.
Measure	A numeric fact or result of a calculation.
Retention	Enables you to define the amount of time the data is kept in the repository.

Fact Set Child Element: Fact

The following table contains the fields that are included in a fact element:

Field	Description
Label	If you want to call this fact something other than the name in reports, enter a label. For example, the fact name is BookTitle" and you want to save space on the reports. Enter "Title," as a label for this fact, and the label will be displayed as the column heading for this field on reports instead of "BookTitle." Optional.
Max Length	Default is 255. Valid values are 1-255. Used for string type only.
Name	This is the name that will be associated with the fact in the fact model and in the schemas created from it for reports, Mondrian cubes, and the fact model in the application. Required. Alphanumeric. Any constraints on naming for your database apply to naming fact sets and facts. Also, the names cannot contain spaces or special characters except underscore and hyphen (unless prohibited by your database). You cannot use keywords as fact names.
Type	Select a type from the list: • String – Alphanumeric. Default type • Date – Format the date in the Report Editor • Timestamp – Format the timestamp in the Report Editor • Integer – Positive or negative number or zero • Long – Long Integer field • Float – Real number, single precision floating point • Double – Real number, double precision floating point • Boolean – Uses values of True and False.
Unique	Valid values are true and false. Defines whether there can be more than one occurrence of the fact name in the fact set. Default is false.
NotNull	Simple boolean, defaults to false. Optional.
sql-type	String value. To allow Unicode character support on iSeries, use this specific value to override the default string field type:

Field	Description
	VARGRAPHIC(30) CCSID 13488

Fact Set Child Element: Measure

The following table contains the fields that are included in a measure element:

Field	Description
Aggregate	You can have the Fact Model Editor perform the following operations on a fact within the same fact set: • sum – For numeric data types only. Calculates the total of the facts. • count – For all data types. Counts total number of facts. • min – For numeric data types only. Returns the minimum value included in the facts. • max – For numeric data types only. Returns the minimum value included in the facts. • avg – Reserved for future use. To use the Measure to aggregate data from one fact, select one of the aggregate types, then type the Fact name in the Fact Name field. Type a representative name for this measure in the Name field, such as “SumBooks.” **Note:** The fact used must be part of the same Fact Set as the measure. Sum, Count, Min, Max, Average. Optional.
Fact Name	Enter the name of the fact to use.
Name	Enter a name to describe this measure, for example, “CountWidgets.”

Fact Set Child Element: Retention

The following table contains the fields that are included in a retention element:

Field	Description
Cascade	Deprecated. Leave at default.
Days	Enter the number of days to retain the information in this fact set in the repository. Optional. Valid values are 0-2,147,483,648.
Hours	Enter the number of hours to retain the information in this fact set in the repository. Optional. Valid values are 0-2,147,483,648.
Months	Enter the number of months to retain the information in this fact set in the repository. Optional. Valid values are 0-2,147,483,648.
Purge Time	Enter the time to start the purge in hours, minutes and seconds (24 hour clock).
Weeks	Enter the number of weeks to retain the information in this fact set in the repository. Optional. Valid values are 0-2,147,483,648.
Years	Enter the number of years to retain the information in this fact set in the repository. Optional. Valid values are 0-2,147,483,648.

Dimension

A dimension is a collection of data of the same type. Examples of dimensions are:

• Time (Year, Month, Day, or Hour, Minute, Second)
• Corporate structure (Division, Department, Functional Group; Sales Region, Sales Team, Salesperson)
• Location (Country, State, City, Street)

A dimension must have a hierarchy child element.

Field	Description
Hierarchy	Standard tree view (parent with one or many children) organization for positions in a dimension. Required if Dimension is used. Hierarchies can have level child elements.
Level	Describes the hierarchical structure of the dimension. Arrange in increasing levels of granularity. For example, a Time dimension and hierarchy might have the following levels: • Yearly • Monthly • Weekly Required if Dimension and Hierarchy are used.

Dimension Child Element: Hierarchy

The following table contains the fields that are included in a hierarchy element:

Field	Description
HasAll	Leave at default. Used by Mondrian schema.
Name	This is the name that will be associated with the hierarchy in the fact model and in the schemas created from it for reports, Mondrian cubes, and the fact model in the application. Cannot contain spaces or special characters except hyphen and underscore. Required.

Hierarchy Child Element: Level

The following table contains the fields that are included in a level element:

Field	Description
Expression	Leave at default. Reserved for future use.
Fact Path	Enter the path to the fact defined by this level in the format: *fact set/fact*
Unique Member	Whether this is unique. Valid values are true and false. Default is true.

Mapped Extraction

A mapped extraction enables you to use a map to obtain Reporting Services data from a business process. You can specify one or more maps to use for each mapped extraction. If a mapped extraction uses multiple maps, the maps must all be the same type: either any-to-XML or any-to-JDBC. You cannot use both types in one mapped extraction. However, a fact model can have more than one mapped extraction, so different map types can be used as parts of different mapped extractions.

Field	Description
Map Name	Click the ellipsis (...). On the window that is displayed, enter the name of a map to be used for this mapped extraction in the Value field. Click Add to move the map to the list on the right. Add any additional maps, then use the Up and Down buttons to put the maps into the correct order. The map to be used first should be at the top of the list; the map to be used last at the end of the list. **Note:** If using the passThrough option, leave this field blank.
Name	Enter a unique name for this mapped extraction. You will enter this name as a parameter for the Mapped Extract service when creating the business process. If your data is already in the correct format for the fact repository, you can enter passThrough. **Note:** The .xsd file created for the fact model by the application represents the correct format.
Output Type	Select an option: • If mapping to JDBC or using passThrough, select DB (default) • If mapping to XML, select XML

Mapped Extraction Child Element: Event Schema ID

A Mapped Extraction can have the following child element:

Field	Description
Event Schema ID	Identifies the event that invokes a mapped extraction. This field is automatically assigned by the application when the fact model is checked in, and should be left blank in the fact model. Currently, this is for internal use only.

Fact Pattern

A fact pattern is a description of the logical construct for a fact set. A fact pattern is limited to a single parent fact that may have multiple children.

The fact pattern element itself has no fields.

The fact pattern is optional and can have the following child elements:

Field	Description
Retention	Time to store fact pattern data in the repository before purging.
Root Fact Set	The fact set that is the primary information that ties all other information in the fact model together.

Fact Pattern Child Element: Root Fact Set

The following table contains the fields that are included in a root fact set element:

Field	Description
Key Fact	Links the parent fact set to the child fact set.
Path	Enter the path to the root fact set. Required if fact pattern is used.

Root Fact Set Child Element: Fact Set Ref

The following table contains the field that is included in a fact set ref element:

Field	Description
Path	Enter the fact set associated with this fact set ref. Required.

Fact Pattern Child Element: Retention

The following table contains the fields that are included in a retention element for a fact pattern:

Field	Description
Cascade	Deprecated. Leave at default.
Days	Enter the number of days to retain the information in this fact set in the repository. Optional. Valid values are 0-2,147,483,648.
Hours	Enter the number of hours to retain the information in this fact set in the repository. Optional. Valid values are 0-2,147,483,648.
Months	Enter the number of months to retain the information in this fact set in the repository. Optional. Valid values are 0-2,147,483,648.
Purge Time	Enter the time to start the purge in hours, minutes and seconds (24 hour clock).
Weeks	Enter the number of weeks to retain the information in this fact set in the repository. Optional. Valid values are 0-2,147,483,648.
Years	Enter the number of years to retain the information in this fact set in the repository. Optional. Valid values are 0-2,147,483,648.

Fact Model Naming Considerations

Because the fact set name ultimately becomes the name of a table in the fact repository, and the default fact repository is the application database, it is possible to create fact set names that are not supported in the application. For example:

- Giving a fact set a name that happens to be the same as an application table adds extra columns to the table. This can destroy the application system’s ability to run.
- Naming a fact set the same as one of the columns in the database table causes data to start persisting in the table, causing issues in the application.

Do not use application table names as fact set names. You can check the database repository to verify which names are used, however this is not a fail-safe method, because future application patches may collide with your fact sets. Therefore, best practice is to have a separate Reporting Services repository outside of the application database. A separate fact repository enables you to name fact sets as you please.

Reporting Services Resources

There are several application resources associated with Reporting Services. They can be divided into two types: versioned and transient. It is important to understand how each type is handled in the application, and how changing some resources and checking them back into the application will impact the fact repository and reporting.

- **Versioned resources - fact models, report definitions, report formats**

  Each time you check in the resource, the system treats it as a new object. A versioned resource has the following characteristics:

  - Multiple copies of the resource can be saved in the system with the same name.
  - There is a default version of the resource. For Reporting Services, this is the last version checked in. However, you can manually select a different version to be the default.
  - The date and time for each version of the resource is recorded.
  - Each version of the resource is kept in the database.

    Whenever you check in a fact model, there is a cascading effect that creates new versions of the rest of the resources.

  Note: A default report format is supplied with Reporting Services.

- **Transient resources - Report.xml, language properties files, report properties file, Mondrian schema**

  This type of resource These are not versioned. They are regenerated each time a fact model, report definition, or report format is checked in. Each resource type is described in the following list:

  - Report.xml – contains information about all reports accessed through the application Reporting wizard.
  - Language properties files – used to define screens. Unless needed for localization, do not edit these files.
  - Report properties file – This file is stored on the file system.
  - Mondrian schema (.mond) – This schema can be exported and used with JPivot or other tools to generate a Mondrian cube.

Importing and Exporting Reporting Services Resources

For Reporting Services, when importing or exporting resources, you must manually select the fact model and its associated report definitions and any custom report formats or report configurations. Report definitions, report formats, and report configurations are not automatically bundled with a selected fact model for import or export – you must manually select each file to be imported or exported. When importing a fact model and its associated report files, you must import them in the following order to avoid any possibility of having autogenerated resources overwrite your imported files:

1. Fact model
2. Report formats
3. Customized report definitions
4. Customized report configurations

Checking In a Fact Model

Once you have completed your fact model in the Fact Model Editor, you must check in the fact model to the application. Checking in a fact model creates schemas for the default detail and summary reports, and Mondrian cube schema. These are the keys to viewing your data.

Note: The first time you check in a fact model, it also creates a fact repository for the fact model.

Checking in a Fact Model for the First Time

1. From the application interface, select **Deployment** > **Schemas**.
2. Select **Check in new XML Schema or DTD**.
3. Enter the path and fact model filename, or browse to the file.
4. Enter check in comments, and click **Next**.
5. The default is the name of the fact model file.
6. Select a schema type (in this case, select Reporting Services Fact Model). Click **Next**, then **Finish**.

   Note: The first time you check in a new fact model, the check in process may take a few minutes.

Editing a Fact Model

You edit fact models in the Fact Model Editor Eclipse plug-in. Before doing so, you must check out the fact model from the application.

1. From the application interface, select **Deployment** > **Schemas**.
2. In the List pane, under By Schema Type, select **Fact Model**.
3. On the results page, click **Source Manager** next to the fact model to be checked out.
4. On the Source Manager page, click check out next to the desired version of the fact model.
5. Type or browse to the path on your system where the fact model should be saved.

Checking in an Edited Fact Model

1. From the application interface, select **Deployment > Schemas**.
2. In the List pane, under By Schema Type, select **Fact Model**.
3. On the results page, click **Source Manager** next to the fact model to be checked in.
4. On the Source Manager page, click **Go!** next to check in a new version of this schema.
5. Type or browse to the file on your system and click **OK**.
6. On the Set Default Version page, select the new version as the default.
7. On the Confirm page, click **Finish**.
8. Stop and restart the BI listener JVM. You must do this after checking in an updated version of a fact model. For information on stopping and restarting the listeners see the topic, "Additional Information for Testing Reporting Services Listeners."

Checking in an Updated Fact Model

1. From the application interface, select **Deployment > Schemas**.
2. In the List pane, under "by Schema Type," select **Fact Model** and click **Go!**.
3. On the Schema Management page, click **source manager** next to the fact model to be checked in.
4. On the Schema Source Manager page, click **Go!** next to "Check in a new version of this schema."
5. On the Select Schema page, type or browse to the file on your system and click **Next**.
6. On the Set Default Version page, select the updated version as the default and click **Next**.
7. On the Confirm page, click **Finish**.
8. Stop and restart the BI listener JVM. You must do this after checking in an updated version of a fact model. For information on stopping and restarting the listeners see the topic, "Additional Information for Testing Reporting Services Listeners."

Changing the Default Version of a Resource

1. From the application interface, select **Deployment > Schemas**.
2. In the List pane, under **By Schema Type**, select the type of resource: Fact Model, Report Definition or Report Format.
3. On the results page, click **Version Manager** next to the desired resource.
4. On the Version Manager page, in the Default column, select the desired version of the resource and click **Save**.

How Editing the Fact Model Impacts the Fact Repository

When you check in a new version of a fact model, it causes changes to the structure of the Fact Repository.

- Properties changes – only make changes to properties when instructed by Customer Support.
- Dimension changes – Causes grouping changes
- Facts
  - Changing a fact name: In the fact repository, each fact is a column. If you change a fact name in the fact model, a new column with the new name is generated when you check in the revised fact model. The old column (under the old fact name) is preserved for reporting and tracking purposes.

    You must manually migrate any data from the old fact column to the new column. See your database vendor's query tools documentation for information on using queries to migrate data.

    Caution: Reports based on the older version of the fact model *may* still run; however, because not all resources associated with a fact model are versioned, the report wizard will not display order lists correctly and can display fields that were added to the fact model after this report version was created. It is recommended that you do not run or rely on reports based on older versions of the fact model.

  - Deleting a Fact: There is no change to the fact repository, because the old column for the fact is preserved, regardless. However, reporting resources are impacted by deleting a fact.

    To work around the impact, you can change the name of the fact set; however, this also has implications. See the bullet about *Fact Set Changes*.

  - Data Type: You can change a data type to a like data type; for example, one numeric type to another. To go to a different data type (for example, string to floating decimal point), you can set the autoddl property in bi.properties to drop all tables for the fact model upon check in.

    Caution: This can also be set using a property in the fact model. However, it is not recommended to set this property in the fact model for the following reason: if you import the fact model and its components to a new system (for example, test to production), the setting is carried over with the fact model and could result in loss of data. Instead, use the bi.properties file, which is not imported with the fact model, so the setting does not carry over.

  - Fact length: You can increase a fact length but cannot decrease it, to avoid loss of data through truncation.

- Fact Patterns and Links: The only impact to the fact repository is that it is reindexed. The old indexes are preserved as well as the new.
- Fact Set Changes
  - Changing a fact set name: The fact set name is the table name, so changing the name of a fact set will force creation of a new table. Any data that was in the original name table will not be transferred or migrated by Reporting Services or the application. You would have to manually migrate all data.
  - Duplicate fact set names: Be careful that if you are using multiple fact models that you do not use the same fact set names within each. If you need to do so, you can configure separate database pools for each fact model, which will mean that each has its own fact repository.
  - Deleting a fact set: The table remains in the fact repository for that fact set but no further data is added to it.

Impact to Reporting When Fact Model is Changed

Report resources are changed by edits to the fact model.

- Resources are generated on fact model, report definition, or report format check in.
- Deleting a fact: The Report Configuration wizard pages and reports will no longer contain those fields.
- Adding a fact: The new field is added to the Report Configuration wizard pages. The Report Configuration wizard pages will always reflect the most recent version of a report definition. So it is possible that older report definitions may not display correctly in the Report Configuration wizard. One workaround is to give the fact set that contains the additional fact a new name. This generates a new report definition with its own Report Configuration wizard pages.
- Label changes: If you change a label in the fact model, the report wizard will display the new labels.

Note: There is a setting in the bi.properties file that tells the application to create a backup for the report.xml, report properties and language properties files. The backups are generated each time you check in one of the versioned resources. All of the backup files are preserved on the file system with the extensions bak.1, bak.2, etc. Use the customer_overrides.properties file to set the property value.

Importing and Exporting Reporting Services Resources

For Reporting Services, when importing or exporting resources, you must manually select the fact model and its associated report definitions and any custom report formats or report configurations. Report definitions, report formats, and report configurations are not automatically bundled with a selected fact model for import or export – you must manually select each file to be imported or exported. When importing a fact model and its associated report files, you must import them in the following order to avoid any possibility of having autogenerated resources overwrite your imported files:

1. Fact model
2. Report formats
3. Customized report definitions
4. Customized report configurations

How Changes to a Fact Model Impact Reports

Report resources are changed by edits to the fact model.

- Resources are generated on fact model, report definition, or report format check in.
- Deleting a fact: The Report Configuration wizard pages and reports will no longer contain those fields.
- Adding a fact: The new field is added to the Report Configuration wizard pages. The Report Configuration wizard pages will always reflect the most recent version of a report definition. So it is possible that older report definitions may not display correctly in the Report Configuration wizard. One workaround is to give the fact set that contains the additional fact a new name. This generates a new report definition with its own Report Configuration wizard pages.
- Label changes: If you change a label in the fact model, the report wizard will display the new labels.

Note: There is a setting in the bi.properties file that tells the application to create a backup for the report.xml, report properties and language properties files. The backups are generated each time you check in one of the versioned resources. All of the backup files are preserved on the file system with the extensions bak.1, bak.2, etc. Use the customer_overrides.properties file to set the property value.

Viewing Your Reporting Services Data

Inhere are three tasks that must be performed to view data: Reporting Services, t

- Creating the schemas for the fact model and its components (done by checking in the fact model to the application)
- Editing and creating reports (done after you view the default reports)
- Viewing reports or using the Mondrian cube to view data (done in JPivot outside of the application)

Default Reports

Once you in the fact model, you have access to the default details reports that were automatically created for the fact model (and summary reports for measures, if used). have checked

When the fact model is checked in, the application creates detail and summary reports for each set of linked fact sets within a fact model. It also creates detail and summary reports for each fact set that is not linked to any other fact set. These reports are accessible through **Operations > Reports** in the Admin UI of the application. See the Reporting documentation for the application for detailed information about the reports.

Reports for Fact Models Included with the Application

The following reports are generated for the fact models installed with the application:

Fact Model	Default Reports Generated for the Fact Model
BIFactModel.bifm	BIFactModelDetail_PROCESSEDEVENTS
system.bifm	System Log Detail Report
WFFactModel.bifm	WFFactModelDetail_CnxtCachWarn Report, WFFact Model Detail Supplied Reports

Creating, Editing, and Formatting Reporting Services Reports

Once you have checked in a fact model, you can view the autogenerated reports and note any changes that you want to make. See Reporting for information on viewing reports in the application.

You can check out report definitions in the application and open them in the Report Editor and edit them there. You can eliminate unwanted fields and groups from the report. You can change the order of filter fields. The order determines the default detail order list on the Report Configuration UI for a detail report.

Default Reports

Reporting Services provides a detail report that includes each fact and a group report for each dimension in the fact model.

Note: The report definition filename and the Name attribute within the report definition must be the same. If you save the report definition under a different filename, you must also search the report definition and change all instances of the Name attribute; otherwise, you will receive errors when you check in the report definition to the application. The easier way to create new reports from a report definition is to use the Report wizard in the application.

The default reports contain the following values and sections:

• Query – The system includes one query for each fact when creating the default report template.
• Primary Key – Optional. Subset of fields in the query that uniquely identifies a record.
• Group by – One report definition is created for each level in the fact model.

To edit your default reports, check out the report definitions and open them in the Reporting Services Report editor.

1. Once you have checked out a report definition from the application, you must copy it into the Report Editor. After saving the report file to a location on your hard drive, open the Report Editor in Eclipse.
2. Create a new project, or ensure that the existing project you want to use for this report definition is listed in the Package Explorer in Eclipse.
3. Drag and drop the report from its location on your hard drive to the Package Explorer in Eclipse.

Note: The report definition filename and the Name attribute within the report definition must be the same. If you save the report definition under a different filename, you must also search the report definition and change all instances of the Name attribute; otherwise, you will receive errors when you check in the report definition to the application. The easier way to create new reports from a report definition is to use the Report wizard in the application. See Reports for information about using the Report wizard.

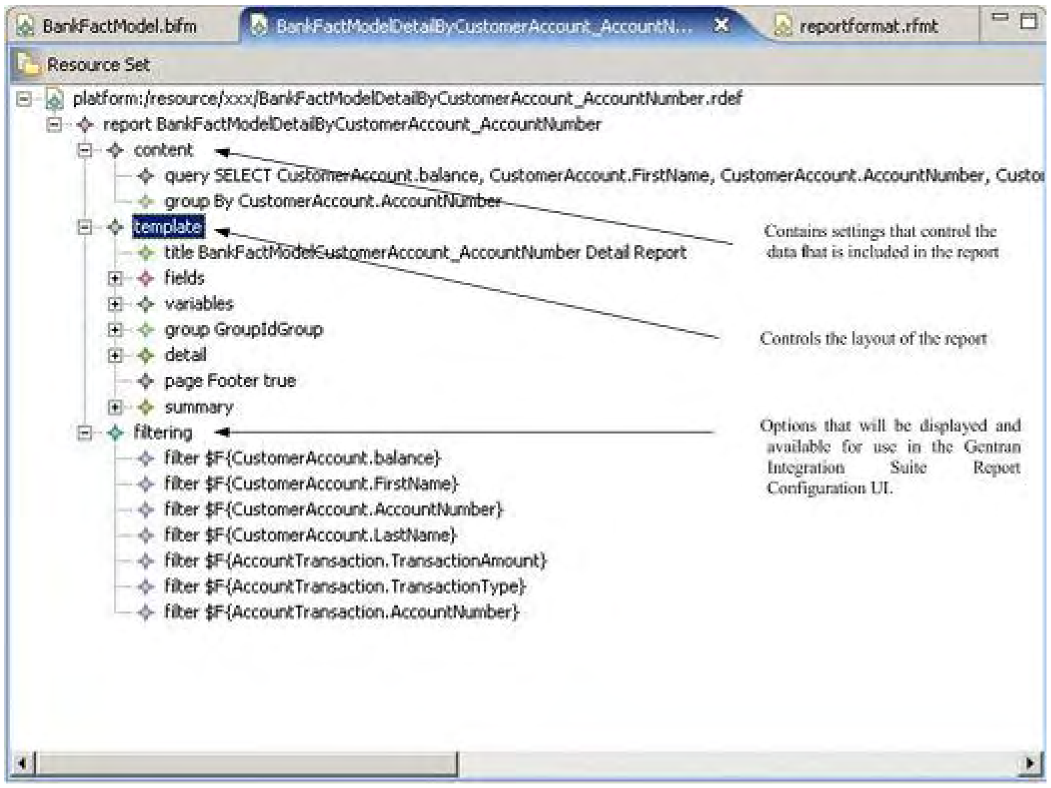
Default Report Content

The default reports contain the following values and sections:

• Query – The system includes one query for each fact when creating the default report template.
• Primary Key – Optional. Subset of fields in the query that uniquely identifies a record.
• Group by – One report definition is created for each level in the fact model.

Parts of a Report Definition

The following figure shows the three sections of a report definition:



```
platform:/resource/xxx/BankFactModelDetailByCustomerAccount_AccountNumber.rdef
  report BankFactModelDetailByCustomerAccount_AccountNumber
    content
      query SELECT CustomerAccount.balance, CustomerAccount.FirstName, CustomerAccount.AccountNumber, Custo
      group By CustomerAccount.AccountNumber
    template
      title BankFactModelCustomerAccount_AccountNumber Detail Report
      fields
      variables
      group GroupIdGroup
      detail
      page Footer true
      summary
    filtering
      filter $F{CustomerAccount.balance}
      filter $F{CustomerAccount.FirstName}
      filter $F{CustomerAccount.AccountNumber}
      filter $F{CustomerAccount.LastName}
      filter $F{AccountTransaction.TransactionAmount}
      filter $F{AccountTransaction.TransactionType}
      filter $F{AccountTransaction.AccountNumber}
```

Contains settings that control the data that is included in the report

Controls the layout of the report

Options that will be displayed and available for use in the Gentran Integration Suite Report Configuration UI.

Content

These are the query options, which include one for each fact. The types of queries are:

• Primary Key. Optional. Subset of fields in the query that uniquely identifies a record.

• Group by. One report section is created for each of the levels in the fact model.

Template

The following table describes the template options that can be changed:

Option	Description
Title	Text field for title. This includes two subfields: • Title label – appears on the report • Value – designates the variable or field
Field	One field is created for each fact in the fact model.
Variable	Created when there are measures in the fact model. If there is a Groupby in the content, then two variables are created for each measure in the fact model; one variable is for the group result and another variable for the entire report summary. When there are no groups, then, for each measure, only the entire report variable is created.
Group	Groups determine the "Report By" options available in the application Reporting UI, and the sections in the report. Groups contain the following attributes: • GroupID – Select true to display GroupID in the page footer. • Within GroupID, there is a text field that contains a Title label and Value. The Value is a Java expression composed of fields/variables. Fields and variables would be wrapped by $F() and $V(), respectively.
Group Footer	This creates the group summary section of the report. It contains the following: • Text field for group footer, which contains a Title label and Value. The value is a Java expression composed of fields/variables. Fields and variables would be wrapped by $F() and $V(), respectively. • Line • Rectangle
Page Footer	Takes value or true or false. If true, shows page numbers on report in the right footer section.
Detail	One created for each fact. Determines what will display in the detailed report that is automatically generated.
Summary	Shows an aggregated summary for all dimensions (groups). Options are the same as Group property options.
Configure UI report wizard filtering options	By default, the system creates a filter option for every fact. • Control – type of UI control that should appear in the wizard for selecting filters. For example, for a text string, this would be a combo list box (select form drop down list), and for a numeric field it would be a text control (allows you to enter a string). • Fields. One is created for each fact. • Label. This is the text string that will appear as the label on this element (row header) on the report. **Note:** The order of filter fields in the definition determines the detail order list on the UI for a detail report.

Customizing the Appearance of Reporting Services Reports

You can create additional report formats by using the Reporting Services Format Editor in Eclipse. A default format is installed with Reporting Services. It is recommended that you do not change the original format, but you can create other “default” formats from it and change the formats and settings in those.

Note: To apply a different format to a report, you must edit the entry for the report in the bi.properties file.

Editing a Report Format

Examples of options you can specify are colors, fonts, spacing, or page layout for a specific report format.

Report format properties are based on the default report format. Any changes made override the defaults. Edits can be made to any of these properties without affecting the defaults for the others. To make changes that can be used on many reports, you can create a different default report format that can be referenced by more than one report.

Changes can be made to the following properties:

- Line
  - Align
  - Back color
  - Height
  - Width
  - X
  - Y
- Rectangle
  - Backcolor
  - Forecolor
  - Height
  - Position type
  - Width
  - X
  - Y
- Text Field Values, Label Title
  - Align
  - Back Color
  - Font Color
  - Font Size
  - Height
  - Value
  - Width
  - X
  - Y
- Title
  - Rectangle
  - Value
- Report
  - Base type

  - Base type label. Used in UI as the base label for Report Configurations when a new one is created, for example.
  - Name

- Query String
  - Query statement string used for the report

- Template
  - Bottom margin
  - Date pattern
  - Generate fields
  - Generate group ID
  - Left margin
  - Orientation
  - Page height
  - Page width
  - Right margin
  - Top margin

Reporting Services Best Practices

Databases

- Before you begin using Reporting Services, decide where your fact repository should reside. As a best practice, it should not be in the same database as your application database. Here are some issues to consider when deciding which database to use for your Reporting Services fact repository:

  What type of usage will the fact repository have? Will there be continuous, high volumes of data being extracted and stored to the fact repository, or will it be sporadic? Are there certain times of the day that will have high volume and others that will be slower? Will the data flow for Reporting Services impact the day-to-day operations of the application by using resources from the same pool?

  Plan your fact model with your selected database in mind. Ensure that you allow for any database restrictions on field or column width in the database you will use. For example, some versions of DB2 have a maximum column name size of 18 characters. So for DB2, if your fact names have more than 18 characters, the fact names will be truncated when the columns are created.
- Do your initial Reporting Services configurations and testing on a test system, then move the final version of your fact models, report definitions, and report formats to a production system.
- Configure your test system to drop tables each time you check in a new version of your fact model. This keeps your fact repository from getting too crowded with columns. (Each time you add a new fact to a fact set, a new column is added to that fact set’s table. All columns are retained permanently in the table, even if the facts they are associated with have been deleted from the fact set, so tables can become unnecessarily large.)

Eclipse Plug-ins

- Create a single workspace for each fact model and its associated reports and custom report formats. This will make it easier to keep track of which reports are associated with the fact model when in Eclipse.
- Customize the Eclipse perspective to display the Reporting Services and MESA Developer Studio plug-in options on the menus.

Fact Models

There may be several ways to design one particular fact model, but experiment to find the best way to construct the fact model for your needs. Think about how you want to see the data once it is compiled and design the fact model to achieve those results.

- Include only the data that you need in a fact model.
- Ensure that the structure of the fact model will collect data in a way that enables you to get the reports you want.

Using Default Reports as Templates

Before creating any new report definitions, review the autogenerated, default report definitions to see if they meet your needs. Instead of creating brand new report definitions, edit the existing ones in the Report Editor to suit your needs. You can also customize the default reports in the application reporting function by applying different filters and saving them as new report configurations.

Editing Default Reports

In Eclipse, you should always edit the default reports to remove unwanted fields and groups. When default report definitions are created, they include all possible groupings by default (one group per level in the fact model). You can then save them as new report definitions (or as new versions of the default report definitions).

Make copies of your report definitions and do any customization of the report definitions on these copies. Once you have the report definitions finalized, you can create and save any report configurations that you need based on the copies.

Maps/Data Extraction

Set up your mapped extractions to use XML whenever possible, rather than having the data extracted using JDBC. Using XML may be quicker: when using XML, have the Map Editor use the XML schema for your fact model to create the output side of your map automatically.

Data Retention

Set data retention for the fact repository by creating retention items in the fact model for the fact sets and fact patterns. The default is 60 days.

Testing

The Reporting Services log, BizIntell.log, can provide useful troubleshooting and debugging information while testing your Reporting Services installation. Turn the logging level to **On** for BizIntell.log while testing. To get to the log settings, go to **Operations > System > Logs**, go to the Reporting Services pane, and click the Edit icon.

Reporting Services Frequently Asked Questions

Can I Use Reporting Services to Generate Mondrian Cubes?

For each fact model that is checked into the application, a Mondrian cube schema is created. You can export and use this with tools such as JPivot.

Is Clustering Supported by Reporting Services?

Application clustering is supported as follows: each application node would have its own JMS provider instance and Reporting Services JVM. These would not speak to one another: all Reporting Services events and communication between Reporting Services and JMS queue and application JVM are local (that is, within the same node).

Reporting Services Checklist

Use the following checklist to help you plan your Reporting Services implementation. It is recommended that you follow the sequence shown in the table for these tasks.

Process	Description
Phase 1 – Planning	• What business information do you want to see? • What business processes and services are needed to extract the selected information from the datastream? • Does any of your data need to be translated or converted before it goes to the repository? • What type of database will be used for the fact repository?
Phase 2 – Installation	
Install the application	You must install the application first.
Install MESA Developer Studio and Reporting Services	Reporting Services requires the MESA Studio plug-in for Eclipse. MESA Studio is licensed and purchased separately. Contact Sterling Commerce Customer Support for more information.
Phase 3 – Define and Extract Data The processes in this phase are iterative, although the order or number of steps may change with each iteration. It is recommended that you wait to customize your reports until you are satisfied that your fact model is correct, and that the autogenerated reports display the information that you want to see.	
Plan and create a fact model and define the data	You create the fact model in the Reporting Services Fact Model Editor in Eclipse. How you structure the fact model determines how you will ultimately be able to view your data. Once you have created the fact model, use it to define all of the data and relationships necessary.
Translate, transform or convert data to another format	If there is data that must be translated or transformed in some way, create maps in the Map Editor to perform the necessary functions. Check the maps in to the application and test them. In the fact model, add one or more mapped extraction components for the maps (depending on whether the maps will be used in a single business).
Set up data extraction	To populate the Reporting Services fact repository, you must create new business processes or edit existing business processes in the application to extract the data. Check in the business processes and test them.
Create reports, fact model schema and Mondrian schema	Checking in your fact model to the application creates a basic set of reports, a schema for a Mondrian cube, and the fact model schema.

Process	Description
Edit/debug the fact model	To edit a fact model, check it out of the application and save the file to the workspace folder for this project on your computer. Open the file in the Fact Model Editor in Eclipse.
Update reports and schemas with changes	After making changes to the fact model, you must check in the fact model again. When you check in the fact model, all associated files (reports, report formats, Mondrian schemas) are regenerated and reflect the updates to the fact model. If you need to go back to an earlier version of one or more files, .bak files are created and stored on the file system.
View your business data	View reports for your fact model. Ensure that they include the correct data, and that the data is presented as you need it.

Phase 4 – Edit Existing Reports and Create New Reports

Once you have completed all the iterations necessary of the preceding processes, you are ready to edit the existing reports or create new ones.

Edit reports and report formats	To edit the reports or the report formats, check out the report or format file and save it to the workspace folder for this project on your computer. Open the file in the Report Editor or Format Editor in Eclipse. You can also create custom reports and report formats using the editors.
Update reports and report formats	After changing a report or report format, check in the file. This creates a new version of the report or format.

Copyright

Index