

IBM WebSphere Commerce Analyzer



Technical Reference

Version 5.5

IBM WebSphere Commerce Analyzer



Technical Reference

Version 5.5

Note

Before using this information and the product it supports, read the information in Appendix B, "Notices", on page 93.

First Edition, First Revision (September 2003)

This edition applies to version 5.5 of IBM WebSphere Commerce Analyzer and to all subsequent releases and modifications until otherwise indicated in new editions.

IBM welcomes your comments. You can send your comments by using the online IBM WebSphere Commerce documentation feedback form, available at the following URL:

<http://www-3.ibm.com/software/genservers/commerce/rcf.html>

© Copyright International Business Machines Corporation 2000, 2003. All rights reserved.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

About this book	v
Conventions used in this book	v
Related information	v
Support Web sites	vi

Part 1. Overview 1

Chapter 1. What is WebSphere Commerce Analyzer?	3
Who uses WCA?	3
How does WCA work with WebSphere Commerce?	4

Chapter 2. Business questions and business reports	7
Examining the categories of business questions	7

Part 2. Working with the WCA server 11

Chapter 3. Maintaining the WCA server	13
Backing up the WCA databases.	13
Removing the datamart using the DB2 Control Center Drop menu	13
Determining language and currency properties for a store.	14
Changing the default reporting currency.	15
Removing a store	15
Maintaining reference texts	16
Determining the properties of the WCA fiscal calendar	20
Making fiscal year modifications	22
Changing the buffer pools, tables, and table spaces for customized DMS	23
Keeping product and concept hierarchy tables populated	24

Chapter 4. Tracing and logging	25
Generated log files	25
TraceLog viewer.	25
Trace levels	26

Chapter 5. Improving performance	27
Reorganizing the WCA tables	27
Updating MAX_SYNCH_MINUTES if new daily data is very large	27

Part 3. The WCA ETL processes 29

Chapter 6. The WCA ETL flows	31
---	-----------

Chapter 7. Replication and extraction	33
--	-----------

Chapter 8. Using the ETL Driver	37
Supported command tags	38
Supported global commands	38
Supported local commands	39
Guidelines for Compatible DB2 Script Files.	42
INSERT SQL statements	42
UPDATE SQL statements	43
Modes for combination SQL statements	43
Running under cursor mode.	43
Running under export/import/load mode	44
Error handling	45

Part 4. Customization 47

Chapter 9. Before Customizing WCA	49
WCA internal operations	49
Extension and revision	49
Extension to WCA	50
Revision to WCA	50
Customization scenarios	50
WebSphere Commerce implementation changes scenarios	50
Datamart processing scenarios	51
Missing data from the WCA datamart scenarios	52

Chapter 10. Customizing the WCA datamarts	55
Overview of the WCA datamarts	55
Datamart coding standards	55
Datamart customization tasks	55
Changing tables	55

Chapter 11. Customizing extraction	57
Extraction coding standards	58
Customizing ETL	58
Creating a customized .sql file	58
Defining the SQL statements.	59
Adding the execution commands	59
Adding the file to the proper location	60
Creating a customized script file	60
Adding customized script file error messages	60
Creating a user-defined program to access the SQL file	60
Maintaining the tables.	61
Adding a step in the Data Warehouse Center control database	61
Scheduling the step to run	62

Chapter 12. Customizing replication	63
Overview of replication	63
Capture and Apply.	63
Staging tables.	64
Replication coding standards	64
Replication customization tasks.	65

Creating a new replication table	65	Determining the recency, frequency, and monetary value	76
Increasing the log space	65	Appendix A. WCA parameters	77
Calculating table space sizes.	65	List of pre-defined WCA parameters	77
Modifying replication control tables	68	Creating or changing parameters	87
Starting the Capture process.	69	Parameter table after configuration	88
Registering the replication process.	69	Appendix B. Notices	93
Running the replication steps	69	Trademarks	94
Scheduling replication and extraction.	70	Glossary	97
Chapter 13. Customizing data mining	71	Index	99
Overview of data mining.	71		
Examples of database model for clustering	72		
Data mining customization tasks	73		
Creating data mining models	73		
Registering new data mining models	74		

About this book

This book provides information for system administrators and marketing analysts about using IBM® WebSphere® Commerce Analyzer, Version 5.5 (also called WCA). After you install, configure, and perform the post-configuration setup of the WCA server, use this book for information about using WCA, which includes topics such as:

- How to perform regular maintenance activities on the WCA server
- How to improve the performance of the WCA server
- How to customize WCA.

Conventions used in this book

This book uses the following highlighting conventions:

Boldface type indicates commands or graphical user interface (GUI) controls such as names of fields, buttons, or menu choices.

Monospace type indicates an example, text you type, or text that is displayed on the screen.

Italic type indicates new terms, book titles, CD labels, or variable information that must be replaced by an actual value.

Related information

IBM WebSphere Commerce Analyzer, Version 5.5 Technical Reference and *IBM WebSphere Commerce Analyzer, Version 5.5 Datamart Reference* are available from the WebSphere Commerce Web site:


- Business Edition:
http://www.ibm.com/software/webservers/commerce/wc_be/lit-tech-general.html
- Professional Edition:
http://www.ibm.com/software/webservers/commerce/wc_pe/lit-tech-general.html
- WebSphere Commerce - Express:
www.ibm.com/software/commerce/express/

The *IBM WebSphere Commerce Analyzer, Version 5.5 Installation and Configuration Guide* and the README are also available from the Web site.

Note: The *IBM WebSphere Commerce Analyzer, Version 5.5 Installation and Configuration Guide* is also available on the *IBM WebSphere Commerce Analyzer* CD in the *locale* directory. After installation, the books for those locales that are installed can also be found on the WCA server in the %IWDA_DIR%\doc\locale directory. The English books are installed in the %IWDA_DIR%\doc\en_US directory.

- *locale* is the locale of the computer. For example, for United States English the locale is en_US.
- %IWDA_DIR% is an environment variable that represents the Windows® directory where WCA is installed. By default, this directory is:
C:\Program Files\IBM\WCA

The following table shows the documentation provided with WCA, along with descriptions and file names.

Book	Description	PDF File name
<i>IBM WebSphere Commerce Analyzer, Version 5.5 Installation and Configuration Guide</i>	Provides information that is required to install and configure the WCA server.	install.pdf
<i>IBM WebSphere Commerce Analyzer, Version 5.5 Technical Reference</i>	Provides information about the following topics: <ul style="list-style-type: none"> • Maintaining the WCA server • Solving problems that might occur • Improving performance •  Using IBM DB2® Intelligent Miner™ for Data • Contains information for the System Administrator about: <ul style="list-style-type: none"> – Customer scenarios – Extending the database schema (adding to the business reports; this might be adding fields to the WCA datamart or, more often, adding a report) – Customizing the database schema by adding to the business reports, adding a report, or changing the schema provided with WCA) 	techref.pdf
<i>IBM WebSphere Commerce Analyzer, Version 5.5 Datamart Reference</i>	Contains information about WCA datamart tables and views.	datamrt.pdf
README file	Provides last-minute information about WCA	README.txt

You can use Adobe Acrobat Reader to view the PDF files. Go to www.adobe.com for information about Adobe Acrobat Reader.

For information about WebSphere Commerce, see the following documents:

- *IBM WebSphere Commerce Quick Beginnings*
- *IBM WebSphere Commerce Installation Guide*
- *IBM WebSphere Commerce Fundamentals*

Support Web sites

Check the WebSphere Commerce Web site for WebSphere Commerce FixPaks. Any updates to WCA will be included in the WebSphere Commerce FixPaks:


- www.ibm.com/software/commerce/support/

Updates to this guide are found at at:

- www.ibm.com/software/commerce/library/

See the following Web sites for support information about products that are included with WCA 5.5:

IBM DB2 Universal Database™ Enterprise Server Edition
www.ibm.com/software/data/db2/udb/support.html

 **IBM DB2 Intelligent Miner for Data**
www.ibm.com/software/data/iminer/fordata/support.html

Part 1. Overview

This section provides an overview of IBM WebSphere Commerce Analyzer, Version 5.5, and includes the following topics:

- What is WebSphere Commerce Analyzer?
- Business questions and business reports

Chapter 1. What is WebSphere Commerce Analyzer?

IBM WebSphere Commerce Analyzer, Version 5.5 (also called WCA) is an optionally installable feature of WebSphere Commerce. WebSphere Commerce provides tools for creating and maintaining an online store. WCA provides information that address the marketing and merchandising activities in the store. You can use this information to help manage the success of the store.

WCA creates and maintains a datamart containing information that is needed to generate business reports about the stores. The datamart is an IBM DB2 relational database that is created on the WCA server. The datamart contains data that is extracted from the WebSphere Commerce transactional database server and reorganized for efficient reporting. IBM DB2 provides the tools you need for database administration.


The business manager accesses the business reports from the browser-based WebSphere Commerce Accelerator, which is installed with WebSphere Commerce.

Note: A business manager could be a marketing, merchandising, or sales manager.

After you install WCA and run the replication and extraction processes for the first time, replication and extraction of new data from the WebSphere Commerce database to the WCA datamart can be scheduled to run periodically. For example, you might schedule these processes to occur just after midnight or at some other time when there is little activity for the store you are managing.

WCA can provide information about multiple stores. The WCA datamart supports multiple languages; however, the reporting application you use might limit the reports to one language.

The following software is provided with WCA:

- IBM DB2 Universal Database Enterprise Server Edition, Version 8.1
-  IBM DB2 Intelligent Miner for Data, Version 8.1

Who uses WCA?

There are three types of WCA users. In this book they are called system administrators, business analysts, and business managers. In some businesses, more than three people might fill these roles, and they might have different titles.

The system administrator is the person who installs, configures, and keeps the WCA server operational. The system administrator also performs the following tasks:

- Administers DB2 and the DB2 Warehouse Center
- Schedules generation of the business reports
- Performs maintenance activities, such as backups, on the WCA server
- Diagnoses and resolves problems that might occur

The business analyst is knowledgeable about data analysis and data mining. For WCA, the business analyst works with the system administrator to do the following:

- **Professional** > **Business** Use the data mining functions of Intelligent Miner for Data
- Define how the business reports are customized

The business manager is concerned with the operation of the stores from a business perspective. This person develops the marketing strategy and tracks the success of the stores. The business manager determines the types of customers the stores target and plans promotional events and their associated advertising.

How does WCA work with WebSphere Commerce?

The following diagram shows the relationship between the components of WebSphere Commerce and WCA.

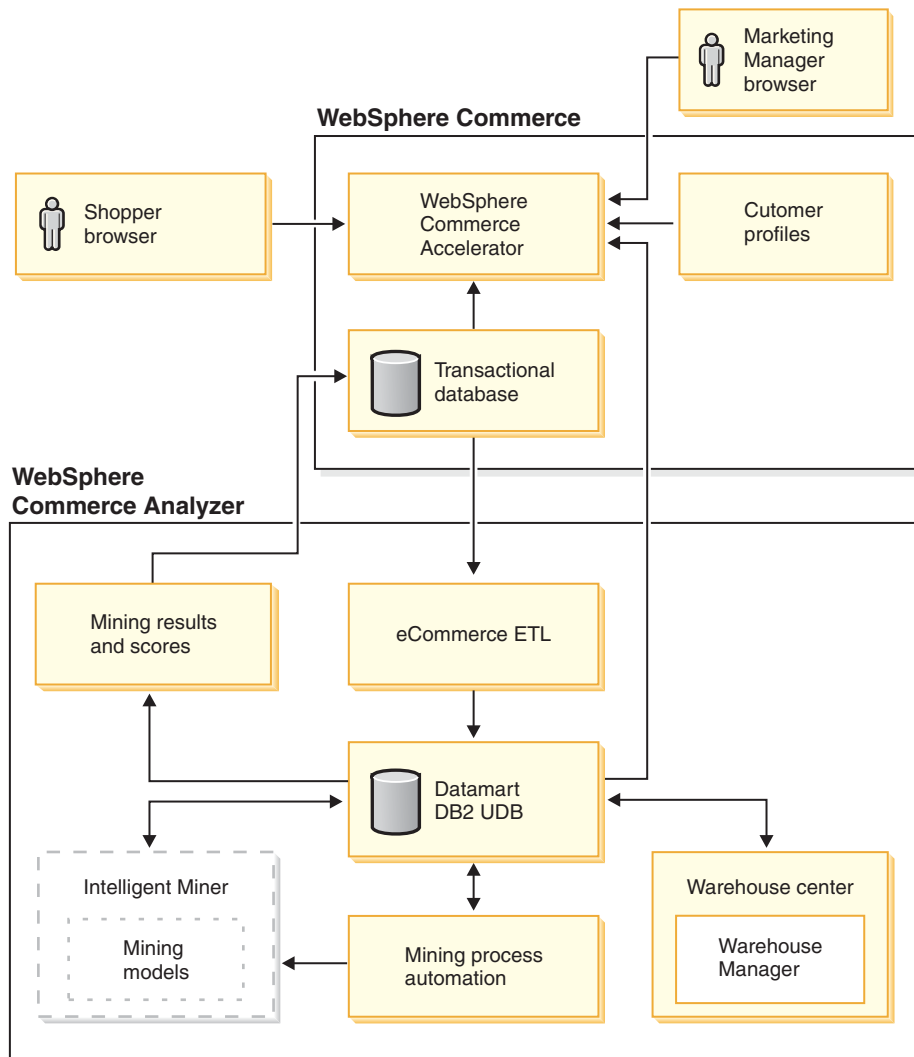



Figure 1. WebSphere Commerce and WCA components

Shoppers use Web browsers to shop at a store that was created using WebSphere Commerce. During a shopping session, a shopper browses the products, views the displayed ads, and might purchase products. WebSphere Commerce collects detailed data about the products and ads that were viewed and the items that

were purchased during each session. This customer session data is stored on the WebSphere Commerce transactional database server.

WCA replicates the customer session data, along with product and promotion data from the WebSphere Commerce transactional database server, into temporary tables on the WCA server. This is called replication. WCA then transforms the data into tables that can be used for reporting and stores it in the WCA datamart. This is called extraction. During the initial replication, all existing data is obtained from the WebSphere Commerce database server. Subsequently, only new data is obtained.

 IBM DB2 Intelligent Miner for Data mines the data in the WCA datamart. It discovers patterns in the data that are useful for answering specific business questions related to data mining and then stores the results in the WCA datamart. From the datamart, the results can be used to produce data mining-related business reports.

You can customize the business reports to better meet the requirements of your business. For information about how to customize the business reports, see the *IBM WebSphere Commerce Analyzer, Version 5.5 Technical Reference*.

Reporting Framework, the reporting toolkit provided with WebSphere Commerce, produces the business reports from the data in the WCA datamart. Other reporting applications can also be integrated to produce business reports.

Chapter 2. Business questions and business reports

The *business manager* is the primary user of the business reports. The information in this chapter gives you an idea of the types of business questions that the business reports can address. This chapter also provides information about how to view the business reports to help make sure that you generate the reports properly.

WCA bases the datamart design on the categories of business questions that are documented in this book. The reports generated by your reporting application might address a subset of these questions, and they might address additional types of questions.

For information about the business reports generated by your reporting application and information about business report problems that might occur and their solutions, see the documentation that accompanies your reporting application.

Examining the categories of business questions

You can see the categories of business questions that are addressed by the business reports in the following list:

Questions related to B2B direct

This category enables you to determine the effectiveness of your business-to-business transactions. Questions in this category focus on the following:

- The sales revenue by account profile geography
- The total sales value, by time period, of the winning Request for Quote (RFQ) responses that have become orders
- The total sales value of the contracts or orders that are the result of auctions
- The number of recurring orders as a percentage of the total orders placed and their value relative to the overall store sales
- The percentage of buyers that use various negotiation models

An example of a question in this category is: Who are the highly profitable and highly unprofitable customers in terms of sales revenue, frequency of orders, and frequency of returns?

Questions related to Catalogs

This category enables you to analyze the catalog entries and products sold. Questions in this category focus on the following:

- The products sold by type and price.
- The catalog entries that are not published.

An example of a question in this category is: Which catalog entries are not published?

Questions related to Products

This category enables you to examine the details of abandoned products and sales values. Questions in this category focus on the following:

- The last 10 products sold.
- The bottom 10 sales values.

- The details about the sales value and products sold.
- The products that are selected and then abandoned.

An example of a question in this category is: What are the bottom 10 products ?

Questions related to Campaign Management

This category enables you to understand the performance of your marketing campaigns. Questions in this category focus on the following:

- The interest shown by customers in the initiatives that are displayed
- The attractiveness and effectiveness of Web site campaigns

An example of a question in this category is: How many times did the customer click on the displayed initiatives?

Questions related to Product Advisor

This category assesses the usefulness of the metaphors available to store visitors. Questions in this category address the three metaphors (Sales Assistance, Product Exploration, and Product Comparison) in the WebSphere Commerce Product Advisor. These questions provide the ability to determine the usage of each metaphor and its contribution toward revenue.

An example of a question in this category is: What percentage of shoppers uses each metaphor?

Questions related to Sales and Orders

This category enables you to understand the trend in sales transactions for your company and assess the factors that drive them. Questions in this category focus on topics such as the following:

- Evaluation of sales in terms of value and units that are sold, and a view of the results across various dimensions
- Characteristics of customers in relation to the number of orders and order value
- The current state of orders

An example of a question in this category is: What is the distribution of customers who bought product 1n times?

Questions related to Shoppers

This category assesses the relationship between shoppers and the orders they are likely to place. Questions in this category focus on topics such as:

- Characteristics of shoppers in relation to order quantity and order value
- Characteristics of shoppers in relation to the categories and products viewed

An example of a question in this category is: What is the relationship between shoppers and the average order value?

Questions related to Web Site Traffic and Navigation

This category enables you to analyze and determine customer trends in store visits. Questions in this category focus on topics such as the following:

- The breakdown of site visitors across various dimensions
- Web sites that refer the largest number of visitors to the site

An example of a question in this category is: What is the breakdown of store visitors by geography?

Part 2. Working with the WCA server

This section discusses how to work with the WCA server. It includes the following topics:

- Maintaining the WCA server
- Tracing and logging
- Improving performance

Chapter 3. Maintaining the WCA server

The sections that follow provide information about maintenance activities on the WCA server:

Backing up the WCA databases

IBM DB2 provides tools to back up the database for archival storage. For instructions, open the DB2 Information Center. Click **Tasks**, expand the **Backup and Restore folder**, and select **Backing up a database** in the DB2 Information Center window.

Note: If you use the instructions to recreate the datamart and then run the replication, promote, or extract processes on the new datamart, you cannot work with the backed-up data because the synchronization points are removed.

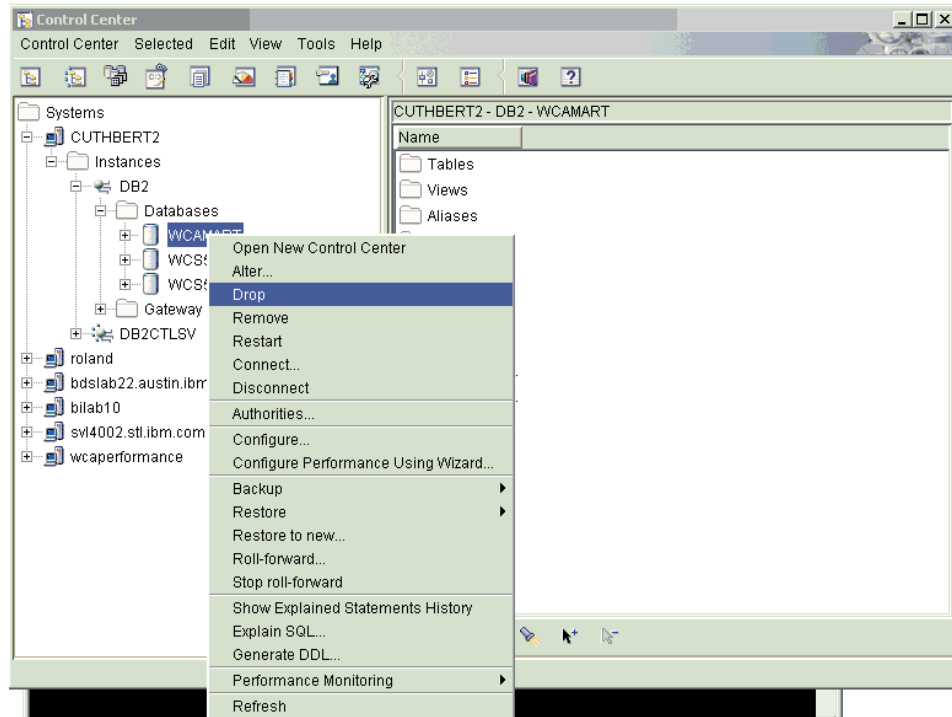
Removing the datamart using the DB2 Control Center Drop menu

Use the **Drop** menu item to completely remove the datamart from the system. See the DB2 documentation for more information about the **Drop** menu item in the DB2 Control Center (pictured below).

Note: If you are using the DB2 Control Center to remove the WCA datamart, do not select **Remove**. The **Remove** menu item removes access to a database,

but does not remove the associated database files. Using **Remove** can lead to unexpected behavior when using the WCA configuration tool.

Figure 2. The DB2 Control Center



Determining language and currency properties for a store

In the Select Online Stores and the Language and Currency for Reports configuration step, the currencies supported by a store are shown in the **Report Currency** list box for the selected store. If multiple stores are selected, the list box contains the common supported currencies. Currency conversions must be available from each default store currency to the report currency as well as for each supported currency to the default store currency.

To review the languages, currencies, or currency conversions that are supported by a store, view the following WebSphere Commerce tables:

CURLIST

Identifies the supported currencies for a store.

CURCONVERT

Identifies the supported currency conversions for a store.

STORE

Identifies the store ID.

STOREENT

Identifies the default currency for a store.

STORELANG

Identifies the languages supported for a store.

For more information about currency conversions for the stores, see the WebSphere Commerce documentation.

Changing the default reporting currency

If you want to change the default reporting currency for a store, do the following:

1. Stop the replication and extraction processes from running during the time it takes to update the various tables with the new reporting currency. When you update the data to the new reporting currency, it might not complete while you run the replication and extraction processes.
2. Make a note of the parameter values for the `TIME_CUT_OFF_LOCAL` parameter. The following command updates the `TIME_CUT_OFF_PREV` parameter to facilitate changing the `REPORT_CURRENCY`:

```
db2 update wca.parameters set (param_value) = '1000-01-01-00.00.00'
```

where `param_type=TIME_CUT_OFF_PREV`

3. Launch the WCA Parameter Manager. Select the **Stores** entry to open the Select Online Stores and the Language and Currency for Reports window. Select the currency from the **Report Currency** list box and apply your change.

Note: For more information about the Parameter Manager, see "Changing the configuration" in *IBM WebSphere Commerce Analyzer Installation and Configuration Guide, Version 5.5*

4. Run **Effective Calculation Start** from the IBM DB2 Warehouse Center. It launches the following steps:
 - EffCalc Orderitems
 - EffCalc Orders
 - EffCalc Fact Event
 - EffCalc Fact Metaphor®
 - EffCalc Interest List
 - EffCalc Product Pricing

Removing a store

If you no longer want to produce the reports for a store, do the following procedures to remove the store:

1. Determine the store ID of the store on which you no longer want to report.
2. Launch the WCA Parameter Manager. Select the **Stores** entry to open the Select Online Stores and the Language and Currency for Reports window.
3. Clear the check box associated with the store ID and apply your change.

After you complete the previous steps, the replication and extraction processes no longer extract data for the store. However, the historical data for the store still exists. When you want to completely remove all of the data for the store, you must remove information in the fact and dimension tables that are based on the store.

To remove the old store data from the tables, do the following steps:

1. Connect to the WCA datamart as the owner of the datamart.
2. Remove all of the rows in the fact tables containing the store ID of the store that you want to remove.
3. Remove all of the rows in dimension tables that do not directly depend on the store ID.

4. Remove all of the rows in the dimension tables containing the store ID of the store you want to remove.
5. Remove the row in the WCA.STORE table containing the store ID of the store you want to remove.

Maintaining reference texts

WCA uses reference tables when descriptions are required in multiple languages for rows in the datamart. The LANGUAGE_ID column in the replication table enables a replication tool to limit the language displayed. The REPORT_LANGUAGE parameter type in the WCA.PARAMETERS table is used in conjunction with the LANGUAGE_ID column in each reference table to provide the information in the WCA language chosen as the default language. This parameter is also used by the WCA ETL process to map from WebSphere Commerce.

Some reference tables are copied from the WebSphere Commerce datamart while others are populated using a predefined properties file, which contains additional descriptions for the WCA datamart.

During the configuration step Load References and Financial Periods Texts, the reference tables are populated and period descriptions are added for each period loaded into the period table.

If the WebSphere Commerce setup has been customized, review the default descriptions and add or modify the references as required before running the WCA configuration.

The population of reference tables is controlled with the %IWDA_DIR%\lib\reftable.properties file.

The language-specific texts are defined in the %IWDA_DIR%\lib\nls\Reference_Table.properties file, respectively %IWDA_DIR%\lib\nls\Reference_Table_locale.properties.

To populate additional reference tables, modify the reftable.properties file and add the appropriate insert statement. The strings enclosed in %-signs within the Reference_Table.properties file are used as keys, with the exception of schema and lang, which are populated automatically. Be sure to continue the uninterrupted numbering scheme for the table keys.

To add additional keys for a reference table, modify the Reference_Table.properties-files for all needed languages. Make sure to add values for the specified keys used in the reftable.properties file. For example, if the keys id, sdesc, and ldesc are used, you must add values for all three keys in the Reference_Table.properties file.

If a reference is missing, the WCA extraction process indicates a failure during the **Start Extraction** Warehouse Center step to add or remove checks.

To include the missing data, you can use the Load References and Financial Periods window to load the modified references, or add them directly into the indicated table. See "Loading references and financial periods" in *IBM WebSphere Commerce Analyzer Installation and Configuration Guide, Version 5.5*.

The following table shows the WCA set of reference tables that have direct mappings to WebSphere Commerce tables.

Table 1. WCA reference tables

WCA Table.Column names	WebSphere Commerce Table.Column name	WebSphere Commerce version
ACCT_STATUS_REF. ACCT_STATUS_ID	ACCOUNT.STATE	<ul style="list-style-type: none"> WebSphere Commerce Business Edition, Version 5.5
AD_TYPE_REF. AD_TYPE_MAP	CPPMN.PROMODISPTYPE	<ul style="list-style-type: none"> WebSphere Commerce Professional Edition, Version 5.5 WebSphere Commerce Business Edition, Version 5.5
ADJUST_LEVEL_REF. ADJUST_LEVEL_MAP	ORDIADJUST.DISPLAYLEVEL	<ul style="list-style-type: none"> WebSphere Commerce Professional Edition, Version 5.5 WebSphere Commerce Business Edition, Version 5.5
ADJUST_TYPE_REF. ADJUST_TYPE.MAP	ORDIADJUST.CALUSAGE_ID	<ul style="list-style-type: none"> WebSphere Commerce Professional Edition, Version 5.5 WebSphere Commerce Business Edition, Version 5.5
ADDRESS_TYPE_REF. ADDRESS_TYPE	ADDRESS.ADDRESS_TYPE	<ul style="list-style-type: none"> WebSphere Commerce Professional Edition, Version 5.5 WebSphere Commerce Business Edition, Version 5.5
AGE_RANGE_REF. AGE_RANGE	USERDEMO.AGE	<ul style="list-style-type: none"> WebSphere Commerce Professional Edition, Version 5.5 WebSphere Commerce Business Edition, Version 5.5
APV_STATUS_REF. APV_STATUS_ID	MEMBER.STATE	<ul style="list-style-type: none"> WebSphere Commerce Business Edition, Version 5.5
BUYERPOTYP_REF. BUYERPOTYP_ID	BUYERPO.BUYERPOTYP_ID	<ul style="list-style-type: none"> WebSphere Commerce Business Edition, Version 5.5
CON_STATUS_REF. CON_STATUS_ID	CONTRACT.STATE	<ul style="list-style-type: none"> WebSphere Commerce Business Edition, Version 5.5
CPN_STATUS_REF. CPN_STATUS_MAP	CPPMN.STATUS	<ul style="list-style-type: none"> WebSphere Commerce Professional Edition, Version 5.5 WebSphere Commerce Business Edition, Version 5.5

Table 1. WCA reference tables (continued)

WCA Table.Column names	WebSphere Commerce Table.Column name	WebSphere Commerce version
CPN_TYPE_REF. CPN_TYPE_MAP	CPPMN.PURCHASECONDTYPE	<ul style="list-style-type: none"> WebSphere Commerce Professional Edition, Version 5.5 WebSphere Commerce Business Edition, Version 5.5
DAY_OF_WK_REF	No corresponding WebSphere Commerce Table	
DAY_RANGE_REF	No corresponding WebSphere Commerce Table	
DR_MEMBER_TYPE_REF	No corresponding WebSphere Commerce Table	
EVENT_TYPE_REF	No corresponding WebSphere Commerce Table	
GENDER_REF. GENDER	USERDEMO.GENDER	<ul style="list-style-type: none"> WebSphere Commerce Professional Edition, Version 5.5 WebSphere Commerce Business Edition, Version 5.5
GENDER_REF. GENDER	No corresponding WebSphere Commerce Table	
HOUR_TYPE_REF	No corresponding WebSphere Commerce Table	
INCOME_REF. INCOME_RANGE	USERDEMO.INCOME	<ul style="list-style-type: none"> WebSphere Commerce Professional Edition, Version 5.5 WebSphere Commerce Business Edition, Version 5.5
INVNTY_STAT_REF. INVNTY_STAT_SDESC	ORDERITEMS.INVENTORY STATUS	<ul style="list-style-type: none"> WebSphere Commerce Professional Edition, Version 5.5 WebSphere Commerce Business Edition, Version 5.5
LAST_UPDATED_REF	No corresponding WebSphere Commerce Table	
MARITAL_STAT_REF. MARITAL_STATUS	USERDEMO.MARITAL_STATUS	<ul style="list-style-type: none"> WebSphere Commerce Professional Edition, Version 5.5 WebSphere Commerce Business Edition, Version 5.5

Table 1. WCA reference tables (continued)

WCA Table.Column names	WebSphere Commerce Table.Column name	WebSphere Commerce version
MEMBER_TYPE_REF. MEMBER_TYPE	USERS.REGISTERTYPE	<ul style="list-style-type: none"> WebSphere Commerce Professional Edition, Version 5.5 WebSphere Commerce Business Edition, Version 5.5
MPE_TYPE_REF. MPE_TYPE	MPETYPE.NAME	<ul style="list-style-type: none"> WebSphere Commerce Professional Edition, Version 5.5 WebSphere Commerce Business Edition, Version 5.5
MPF_TYPE_REF. MPF_TYPE_DESC	USERS.PROFILETYPE	<ul style="list-style-type: none"> WebSphere Commerce Business Edition, Version 5.5
ORDER_STATUS_REF. ORDER_STATUS	ORDERS.STATUS ORDERITEMS.STATUS	<ul style="list-style-type: none"> WebSphere Commerce Professional Edition, Version 5.5 WebSphere Commerce Business Edition, Version 5.5
ORIGIN_REF. ORIGIN_ID	CONTRACT.ORIGIN	<ul style="list-style-type: none"> WebSphere Commerce Business Edition, Version 5.5
PARTROLE_REF. PARTROLE_ID	PARTICIPNT.PARTROLE_ID	<ul style="list-style-type: none"> WebSphere Commerce Business Edition, Version 5.5
PER_AGGR_REF	No corresponding WebSphere Commerce Table	
RANK_RANGE_REF	No corresponding WebSphere Commerce Table	
RFQ_ENDRES_REF. RFQ_ENDRES_ID	RFQ.ENDRESULT	<ul style="list-style-type: none"> WebSphere Commerce Business Edition, Version 5.5
RFQ_STATUS_REF. RFQ_STATUS_ID	RFQ.STATE	<ul style="list-style-type: none"> WebSphere Commerce Business Edition, Version 5.5
RSP_STATUS_REF. RSP_STATUS_ID	RFQRSP.STATE	<ul style="list-style-type: none"> WebSphere Commerce Business Edition, Version 5.5
SEG_ATTRIBUTE_REF	No corresponding WebSphere Commerce Table	
SEGMENTATION_REF	No corresponding WebSphere Commerce Table	
TRADETYPE_REF. TRDTYPE_ID	TRADING.TRDTYPE_ID	<ul style="list-style-type: none"> WebSphere Commerce Business Edition, Version 5.5

Table 1. WCA reference tables (continued)

WCA Table.Column names	WebSphere Commerce Table.Column name	WebSphere Commerce version
TRD_STATUS_REF. TRD_STATUS_ID	TRADING.STATE	<ul style="list-style-type: none"> WebSphere Commerce Business Edition, Version 5.5

Determining the properties of the WCA fiscal calendar

During the configuration step Load References and Financial Periods Texts, the period table is loaded according to the selections for fiscal year start and number of periods to load, and period descriptions are added for each period depending on the language selection. The period load is based on the Gregorian calendar.

The WCA.PERIOD table contains information about each day of the fiscal year. The WCA.PER_DESC_REF field contains language-specific descriptive texts for each period id in the period table.

Note: The PER_ID and CALENDAR_DATE fields are the only fields in the WCA.PERIOD table that affect the extraction process. Do not change these fields or columns.

The following rules apply to the WCA.PERIOD table for the default properties:

Table 2. WCA.PERIOD table rules

Rule	Value
The first day of the WCA default fiscal calendar year	January 1
The last day of the WCA default fiscal calendar year	December 31
The WCA default fiscal calendar observes leap years	February 29, 2000, February 29, 2004, and so on.
Fiscal months have the same boundaries as the Gregorian calendar	<ul style="list-style-type: none"> January 1 - January 31 = Month 1 of the fiscal year February 1 - February 28 or 29 = Month 2 of the fiscal year
Fiscal quarters have the same boundaries as the Gregorian calendar	<ul style="list-style-type: none"> January, February, March = Quarter 1 of the fiscal year April, May, June = Quarter 2 of the fiscal year
The boundaries for the week of fiscal month, the week of the fiscal quarter, and the week of fiscal year.	<ul style="list-style-type: none"> July 31, 2002: <ul style="list-style-type: none"> – WK_OF_FM = 5 – WK_OF_FQ = 5 – WK_OF_FY = 31 August 1, 2002: <ul style="list-style-type: none"> – WK_OF_MO = 1 – WK_OF_FQ = 5 – WK_OF_FY = 31

Note: The boundaries for week of the fiscal year, week of the fiscal quarter, week of the fiscal month correspond to the boundaries used by the Gregorian Calendar with the first day of the year being January, 1st. For example, if the fiscal year 2003 starts on July 1, 2003:

- July 1, 2003: Day = 1, Week = 1, Month = 1, Quarter = 1, Week in Quarter = 1.
- July 31, 2003: Day = 31, Week = 5, Month = 1, Quarter = 1, Week in Quarter = 5
- August, 15, 2003: Day = 46, Week = 3, Month = 2, Quarter = 1, Week in Quarter = 7

The following table shows how each column of the WCA.PERIOD table is populated for default population:

Table 3. Default population of WCA.PERIOD table

Column	Description
PER_ID	The generated ID (starting from 1).
PER_AGGR_ID	The default value for all rows is 1, which indicates that these periods are a portion of the day (for example, second, minute, hour, and so on) This column joins to the PER_AGGR_REF column.
PER_DESC_ID	The same as the generated ID by default. It joins to the PER_DESC_REF column.
CALENDAR_DATE	The date field, which uses the DB2 date type for PER_ID using year, month, and day (for example: 1 = January 1, 2000, 2 = January 2, 2000).
DAY_OF_WK	The day of the week, which resets each week. The default starts with 1 = Monday, 2 = Tuesday, and so on.
DAY_OF_WK_ID	The ID of the day of the week, which joins to the DAY_OF_WK_REF column. Default = DAY_OF_WK.
DAY_OF_FM	The day of the fiscal month, which resets each month.
DAY_OF_FM_ID	The ID of the day of the fiscal month, which joins to DAY_OF_FM_REF. The default is DAY_OF_FM.
DAY_OF_FY	The day of the fiscal year, which resets each year.
DAY_OF_FY	The ID of the day of the fiscal year, which joins to DAY_OF_FY_REF. The default is DAY_OF_FY.
WK_OF_FM	The week of the fiscal month, which resets each month. The defaults are 1 = the first 7 days of the month, 2 = the second 7 days of the month, and so on.
WK_OF_FM_ID	The ID of the week of the fiscal month, which joins to WK_OF_FM_REF. Default = WK_OF_FM.
WK_OF_FQ	The week of the fiscal quarter, which resets each quarter. The defaults are: 1 = the first 7 days of the quarter, 2 = the second 7 days of the quarter, and so on.
WK_OF_FQ_ID	The ID of the week of the fiscal quarter, which joins to WK_OF_FQ_REF. The default is WK_OF_FQ.
WK_OF_FY	The week of the fiscal year, which resets each year. The defaults are: 1 = the first 7 days of the year, 2 = the second 7 days of the year, and so on.
WK_OF_FY_ID	The ID of the week of the fiscal year, which joins to WK_OF_FY_REF. The default is WK_OF_FY.
MON_OF_FY	The month of the fiscal year, which resets each year. The defaults are based on Gregorian calendar months: January, February, March, and so on.

Table 3. Default population of WCA.PERIOD table (continued)

Column	Description
MON_OF_FY_ID	The month of the fiscal year, which joins to MON_OF_FY_REF. The default is MON_OF_FY.
QTR_OF_FY	The quarter of the fiscal year, which resets each year. The defaults are based on Gregorian calendar quarters: Jan, Feb, Mar = 1, and so on.
QTR_OF_FY_ID	The quarter of the fiscal year, which joins to QTR_OF_FY_REF. The default is QTR_OF_FY.
FISCAL_YR	The fiscal year.
WEEKDAY_FLG	The weekday flag. The following are the defaults: 1 MON - FRI 0 SAT - SUN
HOLIDAY_FLG	The holiday flag, which is always 0 by default and requires customization.

The population of the period description table WCA.PER_DESC_REF is controlled by %IWDA_DIR%\lib\refperiod.properties. LOWRANGE and HIGHRANGE define the lowest and highest values for each period reference table.

Table 4. Sources of descriptions for the WCA.PERIOD table

Column	Description
WK_OF_FM	The range is 01 – 05.
WK_OF_FQ	The range is 01 – 15.
WK_OF_FY	The range is 01 – 55.
MON_OF_FY	The range is 01 – 12.
QTR_OF_FY	The range is 01 – 04.
DAY_OF_FM	The range is 01 – 31.
DAY_OF_FY	The range is 01 – 366.

The language-specific texts and the description patterns are defined in the %IWDA_DIR%\lib\nls\Reference_Table.properties file, respectively %IWDA_DIR%\lib\nls\Reference_Table_locale.properties.

For example, to populate the column DAYDESC the key PER_DESC_REF.DAYDESC is used to find the pattern. The default pattern in the Reference_Table_en_US.properties file is FY{0}D{4}. The values in braces are replaced with the actual values for the fiscal year (4 digits) and the fiscal day (3 digits), for example, FY2003D035.

You can change the pattern to modify the description, for example, FY{0}-{2}-{9} would result in FY2003-02-04 for the example above. See the Reference_Table.properties file for a complete list of available options.

Making fiscal year modifications

In the Load References and Financial Periods Texts window in the WCA Configuration, you can select the start of the fiscal year and the number of years for which periods should be loaded.

If you have completed the step, but need to change the start of the fiscal year, do the following steps before you run replication/extraction for the first time:

1. Close the WCA Configuration Manager and the WCA Parameter Manager.
2. Delete all entries from the WCA.PERIOD table and the WCA.PER_DESC_REF table.
3. Delete the parameter types FISCAL_YEAR_START and FISCAL_PERIODS_UNTIL from WCA.PARAMETERS.
4. Open the StepMgr.prefs file in the %IWDA_DATA%\tmp directory.
5. Find the entries for cfg.fyloaded and cfg.fystart and remove them.
6. Save the StepMgr.prefs file.
7. Launch the WCA Parameter Manager and re-run the step to load financial periods.

If replication/extraction has already been run, you must re-run the WCA Configuration Manager to create a new datamart (the period IDs are used in several tables, such as WCA.FACT_ORDERS and WCA.FACT_ORDERITEMS).

Changing the buffer pools, tables, and table spaces for customized DMS

During the configuration step Create WebSphere Commerce Analyzer Datamart, the datamart is set up with pre-defined buffer pools, tables and table spaces.

You can review and change table space settings by clicking the **Customize** button. If you need to make changes to buffer pools, tables, or additional changes to table space settings, you can change the scripts that are used for this step.

The following scripts are used for schema WCA:

```
wca_drp_bufferpools_dms.sql
wca_drp_tbsp_dms.sql
wca_drp_tables.sql
wca_drp_indexes.sql
wca.crt_bufferpools_dms.sql
wca.crt_tbsp_dms.sql
wca.crt_tables_dms.sql
wca.crt_indexes_dms.sql
```

The following scripts are used for schema WSA:

```
wsa_drp_bufferpools_dms.sql
wsa_drp_tbsp_dms.sql
wsa_drp_tables.sql
wsa_drp_indexes.sql
wsa.crt_bufferpools_dms.sql
wsa.crt_tbsp_dms.sql
wsa.crt_tables_dms.sql
wsa.crt_indexes_dms.sql
```

The location of the script depends on the version of the WebSphere Commerce server:

```
%IWDA_DIR%\bin\db2\55be_ext.
```

To modify the scripts, copy the original scripts to the:

%IWDA_DATA%\tmp directory and make any changes necessary before running the Create WebSphere Commerce Analyzer Datamart configuration step.

Keeping product and concept hierarchy tables populated

If you delete records from WCA tables, do not remove records from the tables in this section. These tables must be kept fully populated to support the population of the product and concept hierarchy tables.

Tables that support population of product hierarchy tables:

- IWH.CATGRPREL_R
- IWH.CATENTREL_R
- IWH.CATENTRY_R
- IWH.CATTOGRP_R
- IWH.CATGPENREL_R

Tables that support population of concept hierarchy tables:

- IWH.ICKNOWLEDG_R
- IWH.ICKNOWDESC_R

Chapter 4. Tracing and logging

WCA provides an advanced tracing and logging component (hereafter referred to as *TraceLog*). TraceLog is used primarily for IBM support purposes.

TraceLog offers these benefits:

- Provides comprehensive tracing and logging for WCA runnable components
- Detects WCA product error conditions
- Permits remote error condition diagnosis for IBM Support
- Incorporates WCA and some non-WCA log files

To run TraceLog, enter the following command at a command prompt:

```
%IWD_DIR%\bin\runlm.bat
```

Note: There is also an entry provided in the Startup menu called **Log Viewer** that will invoke TraceLog.

Generated log files

The TraceLog facility generates three primary log files:

- **Table of Contents Log**

The Table of Contents log keeps track of the tasks, steps, and error events that occur during the operation of WCA.

- **Configuration Log**

The Configuration log can be accessed by clicking the **View Log** button in the WCA Configuration Manager panels, or you can jump to the directory in which it is located:

```
~\%IWD_IWDA_DATA%\log
```

- **System Log**

The System log is generated by the ETL Driver during normal operation. It contains information on which ETL steps were executed and whether any errors were produced. The logs are stored in separate daily log files in an XML document format. The Log Browser program is used to view these logs.

TraceLog viewer

The WCA Log Manager enables you to view the WCA comprehensive TraceLog-generated log file. The WCA Log Manager launches a Web browser and then uses the TraceLog Viewer tool to expand an entry and provide details on the state of the WCA system at the time of the event.

The TraceLog Viewer tool is a Java™ application that uses a navigation tree structure to organize all the log files. You can expand or contract the tree to zoom in and out on details of a logged event or logged error, including viewing the values of key parameters. The table of contents frame displays a list of trace logs, ordered by log type and date. If you expand a TraceLog entry, you will see tasks, steps, and actions. You can also see errors, notes, SQL commands, or file links.

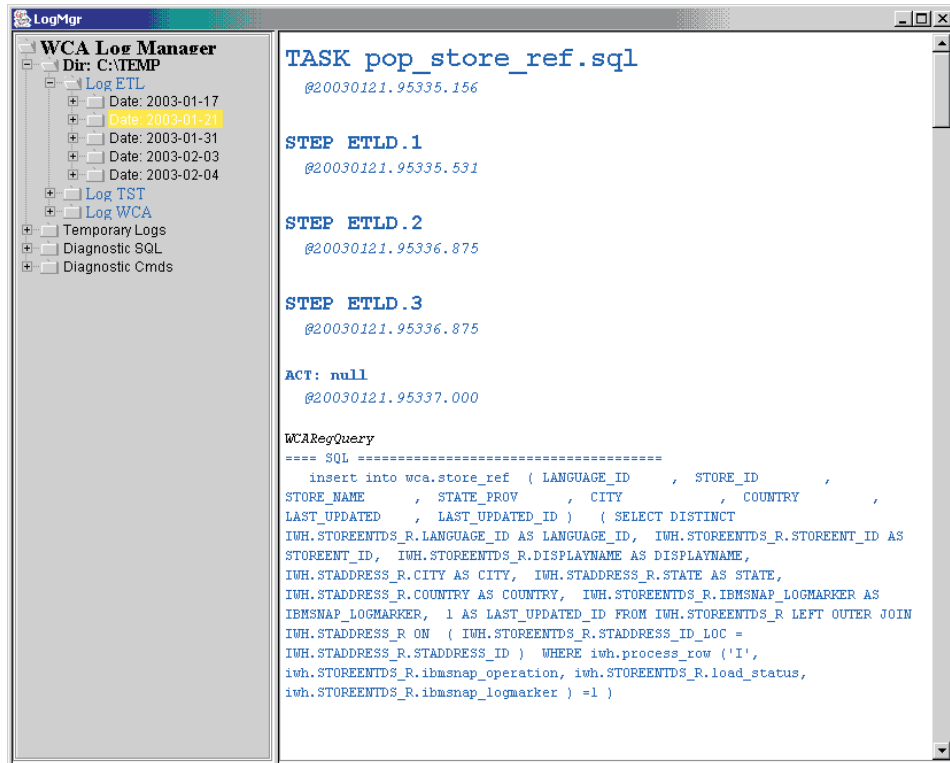


Figure 3. WCA Log Manager

Trace levels

If you need more troubleshooting information, you can turn on tracing by creating a text file called `tracelog.settings` in the following directory:

- `%IWDA_DIR%\log`

If you want to turn on traces for each ETL step, do the following:

- Create the file `tracelog.settings` with the following entry: `ALLTASKS=1`

If you only want traces for a specific step, do the following:

- Look at the Task title in the logmgr display, then type a line similar to the following:
 - `pop_store_ref.sql=1`

Note: Trace files can grow very large very quickly. Remember to turn the tracing off once the problem has been resolved.

Chapter 5. Improving performance

Use the sections in this chapter to improve the performance of the WCA server.

Reorganizing the WCA tables

Over time, when many updates are made to the tables in the WCA datamart, the tables can become fragmented. As a result, subsequent operations on the tables, such as insertions, deletions, updates, and queries take longer to perform. If you notice a degradation in query response times, you might consider reorganizing the tables, especially if the statistics generated by the **RUNSTATS** command are up-to-date.

To determine if you need to reorganize a table, use the **REORGCHK** command. If the output from this command indicates that you need to reorganize a table to improve performance, use the **REORG** command to reorganize the tables.

For detailed information about these commands, see the *IBM DB2 Command Reference*.

Updating **MAX_SYNC_MINUTES** if new daily data is very large

If the amount of new data every day is very large (for example, more than 1,000,000 records), use the following information to update the size of the data block used. You can also update the size of the data block when the log fills up.

If the accumulation of change data is greater than the size of the data block, the Apply program converts a single subscription cycle into many mini-cycles. Also, the Apply program reduces the backlog to manageable pieces. It retries any unsuccessful mini-cycles, and it reduces the size of the data block to match the available system resources. If replication fails during a mini-cycle, the Apply program retries the subscription set, beginning with the last successful mini-cycle. The number of minutes that you specify determines the size of the data block. This value is stored in the **MAX_SYNC_MINUTES** column of the **SUBSCRIPTION SET™** table.

To update the value, type the following commands at the command prompt:

```
db2 connect to wca_datamart_name
db2 update asn.ibmnap_subs_set set max_sync_minutes = number_of_minutes
db2 connect reset
```

Where:

WCA_datamart_name

This parameter provides the name of the WCA datamart.

number_of_minutes

This parameter specifies a value that you choose for the number of minutes. Start with **60** minutes and adjust the value if necessary to improve performance.

For example, you can type:

```
db2 connect to wcamart
db2 update asn.ibmsnap_subs_set set max_synch_minutes = 60
db2 connect reset
```

Part 3. The WCA ETL processes

This section contains information about the WCA ETL (extract, transform, and load) processes. ETL processes are the functions performed when pulling data out of one database and placing it into another of a different type. The following information is included in this section:

- The WCA ETL flows
- Replication and extraction
- Using the ETL Driver

For additional information about WCA ETL, see “Customizing ETL” on page 58 and “Customization scenarios” on page 50.

Chapter 6. The WCA ETL flows

You can determine how WCA runs steps in the IBM DB2 Warehouse Center. An ETL *flow* contains one starting step and one ending step, and all steps in-between are linked together using IBM DB2 Warehouse Center-controlled links.

In IBM WebSphere Commerce Analyzer, Version 5.5 the IBM DB2 Warehouse Center does not tell you the order in which these separate ETL flows are run.

The following table shows each ETL flow in this release, the order in which they are run, and a description of each.

Table 5. ETL flows

Executed When	Flow Order	Warehouse Center Subject Name	Warehouse Center Process Name	Warehouse Center Step Name	Description
Manually	-3	Performance	ODS runstats	ODS runstats	Performance Management flow. This flow can be scheduled or manually run to perform performance enhancing operations on the database (for example, runstats).
On Error	-2	ETL - WebSphere Commerce Analyzer	Product hierarchy	Remove Temporary Table	WCA Product Hierarchy Error Handler flow. Used to automatically clean up if there is an error in the X WCA Product hierarchy step.
On Error	-1	Advanced Utility	WCA Error Handler	WCA Error Handler	WCA Error Handler Flow. This flow is called whenever there is an error in any WCA step. It stops the Warehouse Server and reports the error.
Always	1	Advanced Start and End	Start Extraction	1. Start Extraction	WCA ETL start
Always	2	Advanced Start and End	Start Replication	2. Start Replication	WCA replication start

Table 5. ETL flows (continued)

Always	3	Advanced Start and End	Start Replication	First Replication Step	WCA WebSphere Commerce B2C replication flow
B2B Only	4	Advanced Start and End	WC - B2B Replication Start	WC - B2B Replication Start	WCA WebSphere Commerce B2B replication flow
Disabled	5	Advanced Start and End	WSA Replication Start	WSA Replication Start	IBM Tivoli Web Site Analyzer replication flow
Always	6	Advanced Start and End	Start Extraction	First Extraction Step	WCA standard extraction flow
B2B Only	7	ETL - WebSphere Commerce Analyzer (B2b)	B2b - ACCOUNT	X WCSb Account	WCA extended extraction Flow
Always	7	ETL - WebSphere Commerce Analyzer	fact orderitems	X WCSc Offer Orderitem	WCA B2C and B2B extraction flow, only if B2B is disabled.
Always	8	Advanced Start and End	Effective Calculation Start	Effective Calculation Start	WCA calculation flow, only if IBM Tivoli Web Site Analyzer support is disabled.
Disabled	8	Advanced Start and End	WSA - ETL Start	WSA - ETL Start	IBM Tivoli Web Site Analyzer extraction and WebSphere Commerce calculations flow
B2B Only	9	Advanced Start and End	WC - B2B ETL Start	WC - B2B ETL Start	WCA B2B extraction flow
IM Only	10	Data Mining	Start Data Mining Operations	Start Data Mining Operations	WCA data mining training flow
IM Only	11	Data Mining	Start Data Mining Operations	Start Data Mining Apply Operations	WCA data mining apply flow
IM & Closed Loop Only	12	Close Loop Data Mining	Data Mining Close Loop	Data Mining Close Loop Process	WCA data mining closed loop flow
IM & Closed Loop Only	12	Advanced Start and End	End Extraction	End Extraction	If closed loop is disabled

Chapter 7. Replication and extraction

WCA uses DB2 replication to bring information from WebSphere Commerce database tables into a staging area on the WCA Server. WebSphere Commerce is an example of a Source Server, and any table residing on a Source Server database is called a source table. Here are some guidelines that WCA follows while using DB2 replication:

- For each source table replicated, there is one WCA staging table.
- For each WCA staging table, there is one and only one source table.
- All WCA staging tables are replicated in condensed mode.
- For any given source table, every row is moved into the corresponding WCA staging table.
- In most cases, all columns in a source table are moved into a corresponding WCA staging table.
- Each WCA staging table contains a column named `LOAD_STATUS`, which indicates the current extraction count in which that row was processed. A value of -1 indicates that the row was not yet processed yet. `EXTRACTION_COUNT` is a WCA parameter. The default value for all new rows is -1.
- Each WCA Staging Table contains a column named `IBMSNAP_LOGMARKER`. This column indicates the time stamp in which the row was altered based on the time on the server. This is true except during the first time a source table is replicated. In this case, the time stamp indicates the time in which the row was replicated. This column is often compared with the WCA parameters `TIME_CUT_OFF`, `TIME_CUT_OFF_PREV`, `WSA_TIME_CUT_OFF`, and `WSA_TIME_CUT_OFF_PREV`, depending on the source server.
- Each WCA Staging Table contains a column named `IBMSNAP_OPERATION`. The valid values for this are: I, U, and D.
 - I** indicates that the operation that was done on this row was an insert operation.
 - U** indicates that the operation was an update operation.
 - D** indicates that the operation was a delete operation.
- Each WCA Staging Table contains a column named `IBMSNAP_COMMITSEQ`. This is necessary for replication processing, and not usually used by WCA.
- Each WCA Staging Table contains a column named `IBMSNAP_INTENTSEQ`. This is necessary for replication processing, and not usually used by WCA.
- If the source server is WebSphere Commerce, the schema name for the WCA staging tables is `IWH`.
- If the source server is WebSphere Commerce, the WCA staging table name has the same name as the source server's table name and has the prefix `_R`. In a few cases the tablename is misspelled or does not match due to historical reasons (for example, the source table name has changed between releases.)

The main goal of the WCA extraction is to take a transactional database schema represented in the WCA staging tables and transform it into a star schema. WCA uses a loose set of guidelines to achieve this task that is flexible depending on the requirements. Most extraction takes place in DB2 SQL files which are run through the ETL Driver documented in the next section. It is also true that some tables are populated by using stored procedures or external Java code.

Whether it is using the ETL Driver, stored procedures or Java code, the WCA extraction uses the following algorithm when moving data from the staging tables into the WCA datamart. This algorithm helps WCA determine when a row should be processed from a staging table by using the following factors.

- The type of query being performed
- The value of column IBMSNAP_OPERATION
- The value of column IBMSNAP_LOGMARKER
- The value of WCA Parameter EXTRACTION_COUNT (WebSphere Commerce Version 5.5)
- The value of WCA Parameter TIME_CUT_OFF (WebSphere Commerce 5.5). TIME_CUT_OFF is set at the beginning of the replication cycle.
- The value of WCA Parameter TIME_CUT_OFF_PREV (WebSphere Commerce 5.5). TIME_CUT_OFF_PREV is the last successful extraction time.
- The value of WCA Parameter TIME_WINDOW, which is the time between the last extraction and the beginning of the last replication cycle.

These factors are taken into account by calling the following WCA user-defined functions:

- For WebSphere Commerce Version 5.5:

```
IWH.PROCESS_ROW  
(OPERATION, IBMSNAP_OPERATION, LOAD_STATUS, IBMSNAP_LOGMARKER)
```

where:

– OPERATION is:

- I** is looking for rows within the extraction time window that haven't been inserted into the WCA mart.
- U** is looking for rows within the extraction time window which have been inserted previously into the WCA mart but have since been updated.
- D** is looking for rows within the extraction time window which have inserted previously into the WCA mart but have recently been deleted.
- L** is looking for rows within the extraction time window regardless of whether they've been processed or not.
- N** is looking for rows which have not been inserted into the WCA mart regardless of whether they are within the extraction time window.
- A** is looking for rows which have not been inserted or updated during the current time window.
- J** is looking for rows inserted after the previous extraction that have not been inserted into the WCA mart.
- V** is looking for rows inserted after the previous extraction which have been inserted previously into the WCA mart but have since been updated.

– IBMSNAP_OPERATION is:

- I** Replication specific variable indicating this row was inserted
- U** Replication specific variable indicating this row was updated
- D** Replication specific variable indicating this row was deleted

– LOAD_STATUS is:

An integer that indicates the last time this row was processed by any WCA ETL operation. A -1 indicates that this row was never processed. Whenever a row from a staging table is processed, the LOAD_STATUS column is updated with the value of the current source specific extraction count parameter. Extraction count is simply a counter that is incremented each time a WCA Extraction cycle finishes. For a Websphere Commerce 5.5 source the parameter is EXTRACTION_COUNT.

An integer that indicates the last time this row was processed by any WCA ETL operation. A -1 indicates that this row was never processed. Whenever a row from a staging table is processed, the LOAD_STATUS column is updated with the value of the current source specific extraction count parameter. Extraction count is simply a counter that is incremented each time a WCA Extraction cycle finishes. For a Websphere Commerce 5.5 source the parameter is EXTRACTION_COUNT.

– IBMSNAP_LOGMARKER is:

The source specific timestamp when the operation specified in IBMSNAP_OPERATION took place. The one exception to this is the first time a table is replicated; in that case it is the source specific timestamp during which the replication took place. This column is compared to the extraction time window for the specific source

Chapter 8. Using the ETL Driver

WCA performs many ETL operations in the process of preparing the WCA datamart. Most of these operations are performed using the *ETL Driver*. The ETL Driver is a program which reads a common DB2 batch file or SQL file and executes each of the statements in order against a properly configured WCA datamart. The ETL Driver also enhances the DB2 batch file by using custom ETL Driver command tags, which can greatly increase the performance of the SQL files and perform some WCA extended processing requirements automatically. The ETL Driver is accessible through a user-defined program within the IBM DB2 Warehouse Center Control Database. All WCA ETL operations that use a DB2 batch file use the ETL Driver.

The ETL Driver has the following features:

- It processes a DB2 batch or SQL input file with INSERT, UPDATE, DELETE, DROP, CREATE TABLE, CREATE INDEX, COMMIT, RUNSTATS and REORG commands. Generally, any SQL statement that does not return a result set is supported. In addition to the SQL statements that are supported, the RUNSTATS and REORG DB2 commands are also supported. There are no other DB2 commands supported at this time. Please note that other restrictions might apply.
- It runs SQL statements as is. If there is no tagging specified for an SQL statement, the ETL Driver runs the statement exactly as if it were run from the DB2 command line.
- With proper ETL Driver tagging, the ETL Driver can rewrite queries to perform CURSOR INSERT, CURSOR UPDATE, and BULK LOAD operations. This can greatly increase the throughput of queries.
- **Professional** **Business** It controls the schedule of the WCA Intelligent Miner integration steps based on settings in the WCA Mining configuration panel.
- With proper ETL Driver tagging, the ETL Driver can perform necessary WCA ETL processing automatically. For example, it can update the source table to indicate rows are loaded, or update the last updated time stamp on the target table.

Note: Some code lines in the SQL files associated with WCA begin with:

```
--G--
```

or

```
--L--
```

It is important that you do not modify these lines. They are special command lines for running SQL in the ETL driver. In general, do not change SQL files predefined in WCA. If you do, you are changing the logic of ETL functions. For customization guidelines, see “Customizing ETL” on page 58.

Supported command tags

The ETL Driver is controlled through a set of ETL Driver command tags. These command tags are embedded in DB2 batch file comments. There are two types of ETL Driver command tags, *global* and *local*. Global command tags affect every SQL statement in the batch file, and can be located on any line in the batch file. Local command tags affect only the first statement following the local tag.

You can use any of the WCA supported global commands or local commands in your own customized .sql file.

Supported global commands

The global command applies to all SQL statements that are defined in the specified .sql file. It is easy to identify a global command because the line in the file always begins with:

```
--G--
```

There are currently only two supported global commands: **CommitStatement** and **RunStats**.

CommitStatement

By default, the ETL Driver does not issue commits after each SQL Statement, but only after all SQL Statements in a particular batch file have been run. The **CommitStatement** global command tag can be used to force the ETL Driver to issue a commit statement after each SQL Statement is run. Note that some local commands force the driver to perform commits regardless of this setting.

The syntax of this command is as follows:

```
<CommitStatement>commit_option</CommitStatement>
```

Where *commit_option* carries the value of either **true** or **false**. For example:

```
--G-- <CommitStatement>>true</CommitStatement>
```

RunStats

The ETL driver can run the **Runstats** DB2 command as part of a global command. It is recommended, however, that the user place the **RunStats** command in the DB2 batch script as a statement instead of using the ETL Driver's global command tag. This allows the user to control the exact syntax of the **RunStats** command as well as when the **Runstats** command runs within this file.

For example, the following command informs the ETL driver to compose a **RunStats** command against the *table_name_r* target table and then run the **RunStats** command either before or after running other SQL statements in the script file.

```
--G--<RunStats>  
--G--<TargetTable>table_name_r </TargetTable>  
--G--<RunPosition>position </RunPosition>  
--G--</RunStats>
```

Where *table_name_r* indicates the name of the table to which the **RunStats** command is targeted (for example: *iwh.users_r*), and where *position* carries the value of **before** or **after**.

Supported local commands

A local command only applies to the SQL statement with which it is associated. This is the next uncommented SQL statement that appears in the DB2 batch file following the local tag. Insert the local command before the statement that it applies to. The SQL statement always ends with a semicolon (;) delimiter character. Note that if you decide to comment out a locally tagged SQL statement, it is recommended that you also comment out that statement's local ETL Driver tags or delete the tags altogether. If you don't, those tags are applied to the next uncommented SQL statement and can lead to unexpected results.

You can perform these types of operations using the following local command tags:

- CursorInsert
- CursorUpdate
- LoadInsert

You can identify the local command because the line is prefixed with:

```
--L--
```

The schema for the local command tags are as follows:

```
<ELEMENT CursorInsert (CommitRows,UpdateSource*, LoadOption*, LoadParam*)>
<ELEMENT CursorUpdate (CommitRows, PrimaryKeys, UpdateType, SelectPrimaryKeys*,
UpdateCondition*)>
<ELEMENT LoadInsert (LoadOption, LoadParam*)>
```

See “Running under cursor mode” on page 43 for examples of CursorInsert and CursorUpdate operations, and “Running under export/import/load mode” on page 44 for examples of LoadInsert and LoadUpdate operations.

CommitRows

The **CommitRows** command is applied to all INSERT and UPDATE cases using the **cursor** option.

```
<!ELEMENT CursorInsert (CommitRows, UpdateSource*, LoadOption*, LoadParam*)>
```

or:

```
<!ELEMENT CursorUpdate (CommitRows, PrimaryKeys, UpdateType,
                        electPrimaryKeys*, UpdateCondition*)>
--L-- <CommitRows>num_of_rows</CommitRows>
```

Where *num_of_rows* is the number of rows that should be processed before the incremental commit is performed.

UpdateSource

The **UpdateSource** tag is a special tag that is used to update the source information with the **CursorInsert** tag. See the section on WCA ETL sources for more information regarding this tag. It contains the tags of **SourceTable**, **SourceKeys**, and optional tags of **SourceSet** and **SourceCondition**.

```
<!ELEMENT CursorInsert (CommitRows, UpdateSource*, LoadOption*, LoadParam*)>
<!ELEMENT UpdateSource (SourceTable, SourceKeys, TargetTable*, TargetKeys*)>
--L-- <UpdateSource></UpdateSource>
```

SourceTable

Use the **SourceTable** command in the **UpdateSource** tag for the **CursorInsert** statement as part of the **UpdateSource** tag.

```

<!ELEMENT CursorInsert (CommitRows, UpdateSource*, LoadOption*, LoadParam*)>
<!ELEMENT UpdateSource (SourceTable, SourceKeys, TargetTable*, TargetKeys*)>
--L-- <SourceTable>src_table_name</SourceTable>

```

Where *src_table_name* is the name of the table where the source information is to be updated.

Only one source table can be specified.

SourceKeys

Use the **SourceKeys** command in **CursorInsert** for the INSERT SQL statement as part of the **UpdateSource** tag.

```

<!ELEMENT CursorInsert (CommitRows, UpdateSource*, LoadOption*, LoadParam*)>
<!ELEMENT UpdateSource (SourceTable, SourceKeys, TargetTable*, TargetKeys*)>
--L-- <SourceKeys>srckey1, srckey2, ..... , srckeym</SourceKeys>

```

Where *srckey1*, *srckey2*, , *srckeym* are a list of comma-delimited column names that can uniquely identify the rows in the source table that is to be updated.

TargetTable

Use the **TargetTable** command in the **UpdateSource** tag. The **TargetTable** command is a convenient method used to specify the target table name to the ETL Driver. The table name should be of the format SCHEMA.TABLENAME. This is an optional parameter, as the ETL Driver can automatically determine the target table name from the INSERT statement.

```

<ELEMENT CursorInsert (CommitRows, UpdateSource*, LoadOption*, LoadParam*)>
<ELEMENT UpdateSource (SourceTable, SourceKeys, TargetTable*, TargetKeys*)>
--L--<TargetTable>tgt_table_name</TargetTable>

```

TargetKeys

Use the **TargetKeys** command in the **UpdateSource** tag. The **TargetKeys** command is necessary when the column names specified in the **SourceKeys** command are not the same as the column names in the target table. If this command is used, there must be one column specified in the **TargetKeys** command per column specified in the **SourceKeys** command.

```

<ELEMENT CursorInsert (CommitRows,UpdateSource*,LoadOption*,LoadParam*)>
<ELEMENT UpdateSource (SourceTable,SourceKeys, TargetTable*, TargetKeys*)>
--L--<TargetKeys>tgtkey1,tgtkey2,.....,tgtkeym </TargetKeys>

```

LoadOption

The **LoadOption** command is shared between the **CursorInsert** tag and the **LoadInsert** tag. When included in a **CursorInsert** tag, it is only used when the ETL Driver determines it can use the **Export/Load** option instead of the **CursorInsert**. This occurs when the target table does not already have any rows. After that, the behavior is the same whether it is included in a **LoadInsert** or **CursorInsert** tag.

```

<ELEMENT LoadInsert (LoadOption,LoadParam*)>
<ELEMENT CursorInsert (CommitRows,UpdateSource*,LoadOption*,LoadParam*)>
--L--<LoadOption> load_option </LoadOption>

```

Where *load_option* can only be one of the following values:

import

The SQL statement breaks up into a SELECT SQL statement and an INSERT SQL statement. The SELECT SQL statement part is run using the **export** command, which exports the selected result into an external file with the delimited del format.

load The INSERT SQL statement part is run using the **import** or **load** command, based on the specified option.

LoadParam

Exact parameters vary, based on whether the **import** or **load** command is used.

```
<ELEMENT LoadInsert (LoadOption,LoadParam*)>
--L--<LoadParam>load_parameters <LoadParam>
```

Where *load_parameters* are parameters set by using the **modified by** command in the **import** or **load** commands.

Primary Keys

Use the **Primary Keys** command for UPDATE SQL statements. Omit this command if the list of the keys are the same as those listed in the PrimaryKey list.

```
<!ELEMENT CursorUpdate (CommitRows, PrimaryKeys, UpdateType,
                        SelectPrimaryKeys*, UpdateCondition*)>
<Primary Keys> key1, key2, ....., keym </Primary Keys>
```

Where *key1, key2,, keym* are a list of comma-delimited columns names that can uniquely identify the rows in the table to be updated.

SelectPrimaryKeys

Use the **SelectPrimaryKeys** command for UPDATE SQL statements. If the list of the keys are the same as those listed in the PrimaryKey list, you can omit this command.

```
<!ELEMENT CursorUpdate (CommitRows, PrimaryKeys, UpdateType,
                        SelectPrimaryKeys*, UpdateCondition*)>
--L-- <SelectPrimaryKeys>sk1, sk2,...,skym</SelectPrimaryKeys>
```

Where *sk1, sk2,.....,skym* are a list of comma-delimited column names that can uniquely identify the rows in the table where SELECT is run.

UpdateType

The **UpdateType** command is for UPDATE SQL statements only. There are two types of updates: *simple* or *composite*.

```
<!ELEMENT CursorUpdate (CommitRows, PrimaryKeys, UpdateType,
                        SelectPrimaryKeys*, UpdateCondition*)>
--L-- <UpdateType>type_of_update_sql</UpdateType>
```

Where *type_of_update_sql* can only be one of the following types:

simple

Indicates that the update is done on its own table without a SELECT statement. For example:

```
update table_name set (column_a,column_b) =
                    ( 'value_a','value_b')
    where column_c='xxx' and column_d is not null;
```

composite

Indicates that the update is done with a selection from one table and the update is done on another table. For example:

```
update table_name tb set (column_a, ...,column_n) =
    (select col_a, ....., col_n from table_b_name tb
     where some_conditions
     group by some_columns
    )
where
    some_other_conditions
```

UpdateCondition

The **UpdateCondition** command is for UPDATE SQL statements only. Only use this command for special conditions. The conditions listed are appended to the regular conditions.

```
<!ELEMENT CursorUpdate (CommitRows, PrimaryKeys, UpdateType,  
                        SelectPrimaryKeys*, UpdateCondition*)>  
--L-- <UpdateCondition>special_upd_conditions</UpdateCondition>
```

Where *special_upd_conditions* are additional conditions that must be specified on the UPDATE SQL statement after the statement is decomposed. The regular equal conditions on the key columns listed in PrimaryKeys are added automatically.

LoadInsert

A LoadInsert tag specifies the ETL Driver to always use the Export/Load or Export/Import option. The preferred method is to tag a query with the CursorInsert tag and let the ETL Driver determine whether to use a cursor insert or load method. However, it is still supported to always perform a load. See the LoadOption and LoadParam descriptions in the CursorInsert section.

Guidelines for Compatible DB2 Script Files

There is little benefit in running SQL Statements through the ETL Driver without having them tagged for cursor inserts and cursor updates, aside from a common logging mechanism. This section describes in more detail which SQL and DB2 commands are supported by the ETL Driver.

INSERT SQL statements

- All tagged INSERT statements must have the target columns explicitly defined in the query.

```
insert into tgt_table (tgt_col1, tgt_col2, ....., tgt_coln)  
( select  
    src_col1, src_col2, ....., src_coln  
    .....
```

If tagging is used,

```
insert into tgt_table (select src_col1, src_col2,...,src_coln...)
```

is not supported.

- Triggers do not work with the DB2 **load** command. If you use the **LoadInsert** operation to insert data, any columns that are populated or modified by a trigger must be manually populated either in the SQL statement or in an update statement. It is preferable to populate such columns in the current query if possible. When the driver performs a load, it automatically checks to see if the target table has a **TIMESTAMP** column called **LAST_UPDATED**. If so, the ETL Driver runs an UPDATE statement to update this column to the **CURRENT TIMESTAMP**. The ETL Driver will not handle any other columns automatically at this time. If the following insert query is used:

```
insert into wca.tablename (columnA, columnB)  
(select srcA, srcB from iwh.sourcetable)
```

and columnC is automatically populated using a trigger, the **LOAD** operation does not populate this column correctly. There are two methods to have columnC populated. The first is to populate it in the above SQL:

```
insert into wca.tablename (columnA, columnB, columnC)
(select srcA, srcB, srcC from iwh.sourcetable)
```

or to perform an additional update query:

```
update wca.tablename set columnC = (select value from iwh.sourcetable)
```

or

```
update wca.tablename set columnC = (value)
```

UPDATE SQL statements

- The simple UPDATE SQL statement requires the columns to be grouped in the set:

```
update tgt_table set (tgt_col1,...,tgt_coln)=(val1,...,valn)
where (...update_conditions...)
```

- The composite UPDATE SQL statement requires the update key columns be defined in the SELECT-UPDATE SQL statement:

```
update tgt_table tgt (tgt_col1,...,tgt_coln)
( select
  t1.src_col1,
  t1.src_col2,
  .....
  t1.src_coln
From
  schema.table_name1 t1
Where
  tgt.key1=t1.key1 and
  .....
)
```

Modes for combination SQL statements

Combination SQL statements can be run in either a cursor mode or by using the DB2 **export**, **import**, and **load** utilities, with incremental commits.

Running under cursor mode

You can run SELECT-INSERT and SELECT-UPDATE types of combination SQL statements in cursor mode with incremental commit statements. For the SELECT-INSERT mode, an additional UPDATE of the source table with the key value takes place while the INSERT is being done with the cursor.

Following are example command sets that are defined for the cursor mode types of operations.

CursorInsert

```
<!ELEMENT CursorInsert (CommitRows, UpdateSource*)>
<!ELEMENT UpdateSource (SourceTable, SourceKeys, SourceSet*, SourceCondition*)>
```

An example of cursor insert with update source might be:

```
--L-- <CurserInsert>
--L--   <CommitRows>1000</CommitRows>
--L--   <UpdateSource>
--L--     <SourceTable>iwh.test_r</SourceTable>
--L--     <SourceKeys>p1</SourceKeys>
--L--   </UpdateSource>
--L-- </CurserInsert>
insert into WCA Advancedet1.test2 (p1,a1,a2,a3)
(select
```

```

    p1, a1, a2, a3
  from
    iwh.test_r
);

```

CursorUpdate

Update with cursor (CursorUpdate):

```
<!ELEMENT CursorUpdate (CommitRows, PrimaryKeys, UpdateType,
                        electPrimaryKeys*, UpdateCondition*)>
```

An example of a composite cursor update where the select primary key has the same column name as the update primary key:

```

--L-- <CurserUpdate>
--L--   <CommitRows>1000</CommitRows>
--L--   <PrimaryKeys>p1</PrimaryKeys>
--L--   <UpdateType>composite</UpdateType>
--L-- </CurserUpdate>
update WCA Advancedet1.test1 t1 set (a1,a2) =
(select
  a1,a2
from
  WCA Advancedet1.test2 t2
where
  t1.p1=t2.p1 and
  t2.p1>2
)
where t1.p1 in (select p1 from WCA Advancedet1.test2 where p1>2);

```

Running under export/import/load mode

Use the DB2 **export**, **import**, and **load** utilities to move large amounts of data. The ETL driver enables you to run combination SQL statements (such as SELECT-INSERT and SELECT-UPDATE) and to perform incremental commits using these DB2 utilities.

The SQL statement is broken up into a SELECT SQL statement and an INSERT SQL statement. The SELECT SQL statement part is run with the **export** command, which exports the selected result into an external file. The INSERT SQL part is run by the **import** or **load** command, based on the specified option.

Following are example command sets that are defined for the export/import/load mode types of operations:

LoadInsert

Insert with export/load (LoadInsert):

```
<!ELEMENT LoadInsert (LoadOption, LoadParam*)>
```

An example of a load insert using the **load** command with options:

```

--L-- <LoadInsert>
--L--   <LoadOption>load</LoadOption>
--L--   <LoadParam>modified by forcein</LoadParam>
--L-- </LoadInsert>
insert into WCA Advancedet1.test2 (p1,a1,a2,a3)
(select
  p1, a1, a2, a3
from
  WCA Advancedet1.test3
);

```

LoadUpdate

Update with export/load ():

```
<!ELEMENT LoadUpdate (LoadOption, PrimaryKeys, UpdateType,  
SelectPrimarysKeys*, UpdateCondition*, LoadParam*)>
```

An example of a composite load update:

```
--L-- <LoadUpdate>  
--L--   <LoadOption>import</LoadOption>  
--L--   <PrimaryKeys>p1</PrimaryKeys>  
--L--   <UpdateType>simple</UpdateType>  
--L-- </LoadUpdate>  
update WCA Advancedet1.test2 set (a2,a3)=(100,'t2a3') where p1<>6;
```

Error handling

The ETL Driver is designed to stop when an error occurs (when running an SQL Statement) or when there are internal errors. When using IBM DB2 Warehouse Center, the user-defined program that runs the ETL Driver fails and then exports an error using the IBM DB2 Warehouse Center logging scheme. When an error occurs, the best way to discover the problem is to look at the definition of the step that failed to find the actual SQL file name that was passed to the user-defined program. If this file name is pop_fact_orderitems.sql, there is a log file in the %IWDA_DATA% directory named pop_fact_orderitems.log. The file details what the ETL Driver did and also provides a more detailed error message.



Part 4. Customization

This section contains information about how to customize IBM WebSphere Commerce Analyzer. It includes the following topics:

- Before Customizing WCA
- Customizing the WCA datamarts
- Customizing extraction
- Customizing replication
- Customizing data mining

Chapter 9. Before Customizing WCA

WCA is a fully customizable system where users can enhance the datamart with new tables and columns, add new ETL steps to the process, or change key WCA parameters.

Users interested in customizing WCA should have experience in DB2, WebSphere Commerce, replication, DB2 Warehouse Center,   Intelligent Miner, and ETL before attempting to make any changes. If changes are made, they must be accurately documented since they will not carry over to newer versions of WCA.

WCA internal operations

WCA encompasses three databases:

- WebSphere Commerce source database, which contains the WebSphere Commerce customer, product, and order information from which data is extracted to build the WCA datamart.
- WCA datamart target database, which contains all of the customer, product, and order historical information. It is continuously maintained and updated by the WCA replication and ETL processes.
- WCA Warehouse Center control database, which contains all WCA and ETL step definitions, WCA ETL schedules, and WCA ETL flows.

WCA replication processes run continuously, copying data into staging tables on the WebSphere Commerce and WCA databases. Periodically, WCA ETL processes extract, transform and load data from the WCA staging tables to the WCA datamart. The extraction processes also perform additional steps to update calculated fields and prepare the data for data mining.

The WCA data mining processes apply a collection of mining models to the WCA datamarts. Finally, WCA closed loop steps copy the mining results back to WebSphere Commerce for inclusion in their profiling strategy.

Extension and revision

In WCA, there are two types of customization, *extension* and *revision*.

Extension This term refers to additions to the WCA components (for example, new reports and new datamart tables).



Hereafter, when using the word *customization*, this manual refers to the *extension* definition.

Revision This term refers to changes to the existing WCA components (for example, reports, datamart, and the ETL process).

It is recommended that you make extensions rather than revisions. Extensions are not lost and are preserved in future upgrades to WCA. Some or all of the revisions can be lost in a project upgrade.



Extension to WCA

Extension means that you can do the following:

- Change copies of existing WCA generated reports
- Add new tables to the WCA schema
- Add new rows to the WCA schema tables
- Add extraction steps to handle data extraction
- Add user-defined functions to transform the extracted data copy
-   Change copies of existing or add new IBM DB2 Intelligent Miner for Data mining bases

Revision to WCA

Revision means that you can do the following:

- Adjust or change existing WCA generated reports
- Add columns or increase the column sizes of WCA datamart tables
- Add new rows to the WCA schema tables
- Alter SQL queries to fine tune or enhance existing reports
- Change extraction steps to extract and transform WCA datamart tables
-   Change existing IBM DB2 Intelligent Miner for Data mining bases

Some or all of the revisions can be lost during upgrades to a new version of WCA.

Customization scenarios

As discussed in “WCA internal operations” on page 49, customization of WCA might require changes to several process points in order to complete the necessary customization. Listed below are several customer scenarios and the process points to be completed in each.

Before you customize WCA you must first:

- Identify what the new requirements of the WCA datamart are.
- Become more familiar with the WCA ETL processes documented previously.
- Become familiar with Warehouse Center and the tools it offers.
- Be prepared to document all changes performed to the WCA ETL processes.

To help you identify what the new requirements of the WCA datamart are, refer to the following scenarios:

WebSphere Commerce implementation changes scenarios

Scenario 1a: The WebSphere Commerce implementation for order processing has been modified from the shipped default. This has resulted in a new set of order status codes.

- Understand the current WebSphere Commerce implementation and what each of the new status codes mean. In particular, understand which codes mean pending orders and which mean revenue should be recognized.
- See “Maintaining reference texts” on page 16 to ensure that the WCA reference tables contain all of the order status codes with the correct descriptions. Follow the instructions in this section to make the appropriate changes.

- See Appendix A, “WCA parameters”, on page 77 to find a list of all parameters dealing with order status codes. Use the WCA Parameter Editor to make changes as necessary.
- If WCA is already processing, see “Changing the default reporting currency” on page 15. Since order status codes affect the population of several columns in WCA, you might want to follow this procedure to update those columns based on the new status codes. It is preferable to start with a new WCA installation and make any changes on top of it.

Scenario 1b: The WebSphere Commerce implementation uses different classifications for user demographics than the shipped default.

- Understand the current WebSphere Commerce implementation and what each of the new classifications are.
- See “Maintaining reference texts” on page 16 to ensure that the WCA reference tables contain all of the status codes with the correct descriptions. Follow the instructions in this section to make the appropriate changes.
- If WCA is already processing, then by design all the status codes should be present, but the descriptions might be incorrect. The descriptions do not affect WCA processing but usually appear in reports that can lead to confusing results. It is always possible to change the descriptions in the reference tables after WCA has starting processing. A potential issue is if the same code changes its meaning over time. WCA does not deal with this case out of the box.

Datamart processing scenarios

Scenario 2a: The time period for determining when a shopping cart is really abandoned is too short.

- See Appendix A, “WCA parameters”, on page 77 for a list of all parameters dealing with the number of minutes to allow before a shopping cart is considered abandoned. Use the WCA Parameter Editor to make changes as necessary.
- If WCA is already processing, see “Changing the default reporting currency” on page 15. Since the abandoned minutes affects flags populated in the WCA datamart these instructions might be useful to apply the current settings to previous records.

Scenario 2b: The method used to determine the generic list price for a product is inadequate.

- See Appendix A, “WCA parameters”, on page 77 for a list of all parameters dealing with the product price aggregation function. Use the WCA Parameter Editor to make changes as necessary.
- If WCA is already processing, see “Changing the default reporting currency” on page 15 to apply these changes to the current data in the WCA datamart.

Scenario 2c: Orders to be attributed to campaigns are not being attributed correctly.

- See Appendix A, “WCA parameters”, on page 77 for a list of all parameters dealing with order status codes and the fact event table. These parameters are prefaced with FE. Use the WCA Parameter Editor to make changes as necessary.
- If WCA is already processing, see “Changing the default reporting currency” on page 15 to apply these changes to the current data in the WCA datamart.

Scenario 2d: Some data in the WCA datamart is too raw, and requires additional processing or transformations.

- Understand the WCA datamart and the WCA ETL flows. See “Overview of the WCA datamarts” on page 55 and Chapter 6, “The WCA ETL flows”, on page 31 for more information.
- Identify where the data is located in the WCA datamart.
- Identify in which WCA ETL flow the data in question is actually loaded during the WCA ETL processing.
- Identify in which WCA ETL flow that data is transformed.
- Identify how that information should look and be used.
- Identify the procedure for changing the data so that it becomes the required data.
- Identify what fact tables (if any) or dimension tables (if any) that the new data is related to.
- See Chapter 11, “Customizing extraction”, on page 57 for information about adding this step to the WCA ETL process.

Missing data from the WCA datamart scenarios

Scenario 3a: The user requires information available in WebSphere Commerce and in the WCA datamart but it is not included in any reports.

- Understand the WCA datamart and the WCA ETL flows. See “Overview of the WCA datamarts” on page 55 and Chapter 6, “The WCA ETL flows”, on page 31 for more information.
- Identify where the data is located in the WCA datamart.
- Identify how that information should look and be used.
- Refer to your reporting tool’s documentation to determine how to modify reports.

Scenario 3b: The user requires information available in WebSphere Commerce but is not available in the WCA datamart. (In this scenario, the information is available in the WCA staging area.)

- Understand the WCA datamart and the WCA ETL flows. See “Overview of the WCA datamarts” on page 55 and Chapter 6, “The WCA ETL flows”, on page 31 for more information.
- Identify where the data is located in the WebSphere Commerce database. Locate the table name.
- Identify where the data is located in the WCA staging area by using the table name above. Locate the corresponding table in the IWH schema.
- Identify how that information should look and be used.
- Identify what fact tables (if any) or dimension tables (if any) that the new data is related to.
- See Chapter 10, “Customizing the WCA datamarts”, on page 55 for information about how to modify the WCA datamart.
- Identify the WCA extraction flow where the data should be moved from the staging table.
- See Chapter 11, “Customizing extraction”, on page 57 for information about how to add this step to the WCA ETL process.
- Refer to your reporting tool’s documentation to determine how to modify reports to include this new data.

Scenario 3c: The user requires information available in WebSphere Commerce but is not available in the WCA datamart. (In this scenario, the information is not available in the WCA staging area.)

- Understand the WCA datamart and the WCA ETL flows. See “Overview of the WCA datamarts” on page 55 and Chapter 6, “The WCA ETL flows”, on page 31 for more information.
- Identify where the data is located in the WebSphere Commerce database. Locate the table name.
- Identify the replication flow where the WCA staging table should be populated.
- See Chapter 12, “Customizing replication”, on page 63 for information about how to add this information into a new staging table.
- Identify how that information should look and be used.
- Identify what fact tables (if any) or dimension tables (if any) that the new data is related to.
- See Chapter 11, “Customizing extraction”, on page 57 for information about how to modify the WCA datamart.
- Identify the WCA extraction flow where the data should be moved from the staging table.
- See Chapter 11, “Customizing extraction”, on page 57 for information about how to add this step to the WCA ETL process.
- Refer to your reporting tool’s documentation to determine how to modify reports to include this new data.

Scenario 3d: The user requires information not available in WebSphere Commerce; it comes from an external source. In this case, this information is updated on a regular basis.

- Understand the WCA datamart and the WCA ETL flows. See “Overview of the WCA datamarts” on page 55 and Chapter 6, “The WCA ETL flows”, on page 31 for more information.
- Identify where the data is located.
- Identify if this information belongs in the WCA staging area or directly in the WCA datamart.
- Identify the best method to move that data from the source into the WCA staging area datamart. (replication, straight SQL, or other non-database moves).
- If using a WCA staging area, identify the replication flow where this information should be populated.
- If using a WCA staging area, see Chapter 12, “Customizing replication”, on page 63 for information about how to add this information into a new staging table.
- Identify how that information should look and be used.
- Identify what fact tables (if any) or dimension tables (if any) that the new data is related to.
- See Chapter 10, “Customizing the WCA datamarts”, on page 55 for information about how to modify the WCA datamart.
- Identify the WCA extraction flow where the data should be moved from the staging table or other data source into the WCA datamart.
- See Chapter 11, “Customizing extraction”, on page 57 for information about how to add this step to the WCA ETL process.
- Refer to your reporting tool’s documentation to determine how to modify reports to include this new data.

Scenario 3e: The user requires information not available in WebSphere Commerce; it comes from an external source. In this case, this information is only updated on an infrequent basis.

- Identify where the data is located.
- Identify the best method to move the data from the source into the WCA staging area datamart. (replication, straight SQL, or other non-database moves).
- Identify how that information should look and be used.
- Identify what fact tables (if any) or dimension tables (if any) that the new data is related to.
- See Chapter 10, “Customizing the WCA datamarts”, on page 55 for information about how to modify the WCA datamart.
- Refer to your reporting tool’s documentation to determine how to modify reports to include this new data.

Scenario 3f: The user wants to incorporate new mining models into the WCA implementation.

- See Chapter 13, “Customizing data mining”, on page 71 for information about how to customize data mining.

Chapter 10. Customizing the WCA datamarts

This section discusses how to customize the WCA datamarts. The following topics are discussed:

- Datamart coding standards
- Datamart customization tasks
- Recording the changes

Overview of the WCA datamarts

WCA transforms the operational data from WebSphere Commerce staging tables into multidimensional form and stores it in the WCA datamart.

The WCA datamarts are published and customizable. Refer to the *IBM WebSphere Commerce Analyzer Datamart Reference, Version 5.5* for a complete description of:

- WCA tables and views

Datamart coding standards

You can customize the datamart at any time. You must follow a few rules and guidelines to preserve the operational integrity of the datamart.

Keep the following list in mind when you customize the datamart:

- Do not change the definition of the primary key for WCA tables.
- Do not add constraints to, or delete constraints from, WCA tables.
- Do not delete any WCA tables, views, columns, or rows.
- Do not alter any WCA views joining, sorting, or grouping by clauses.
- Do not change the meaning of a column in a WCA table:
 - Do not change the datatype.
 - Do not change the sourcing of its data or overwrite with any sourcing.
- Do not delete any indexes in WCA tables as this can affect the program efficiency.
- Do not remove any WCA user-defined functions (UDF).
- Do not remove any WCA stored procedures.
- Do not remove WCA bufferpool or tablespaces.

Datamart customization tasks

To automate multiple datamart changes, you should create platform-specific scripting files to automate these types of changes.

Changing tables

This section contains information you must be aware of before making changes to tables in the WCA datamart.

Adding a table

When you add tables to the WCA datamart, use the prefix *UX* to identify the table name as a user extension of the WCA datamart. For example, when adding a table with additional member information, type:

```
create table UX_MEMBER (  
    member_id    bigint not null,  
    custcode     bigint not null,  
    primary key  (member_id)  
)
```

Deleting a table

Do not delete any WCA tables, views, columns, or rows.

Adding a column to an existing table

When you add columns to existing WCA tables, use the prefix *UX* to identify the column name as a user extension of the WCA table.

Run the following SQL statement to add the user-defined `UX_MYMEMBER_ID` column to the `PRODUCT` table:

```
alter table PRODUCT add UX_MYMEMBER_ID big;
```

For example, If you wanted to add the column `MEMBER_ID` to the `PRODUCT` table, you would type:

```
alter table MEMBER add column UX_CUSTCODE bigint not null;
```

In many cases, adding a column requires additional changes be made to WCA replication and WCA ETL processes to get the column populated.

Changing a column in an existing table

Do not change the meaning of a column in a WCA table:

- Do not change the datatype.
- Do not change the sourcing of its data or overwrite with any sourcing.

Recording the changes

If you modify or append data, you must record the changes and ownership of those changes. Record an ID in the `LAST_UPDATED_REF` table. Be sure to choose a unique integer ID (`LAST_UPDATED_ID`) and a unique description ID.

Chapter 11. Customizing extraction

If you want to customize WCA ETL processes you must first consider the following:

- Decide what data you want to place in the WCA datamart.
- Decide which tables must be created and/or what columns contain the data.
- Determine the source of the data.
- Follow the WCA conventions and restrictions to develop SQL to perform the task.
- Create a new WCA process step.
- Link the process step to the proper WCA ETL process flow.

Decisions on what data is needed is the responsibility of the user. In general, the data comes from one of three sources:

- The WebSphere Commerce transactional database
- The WCA replication staging tables
- Some other external source

In the case where WebSphere Commerce transaction data is not already replicated in a WCA staging table, see Chapter 12, “Customizing replication”, on page 63 for information about how to set up the necessary tables, table spaces, and registrations required by replication.

In all cases listed previously, you must develop SQL that extracts, transforms, moves and loads the data into the WCA datamart. This might require creating additional temporary tables, views or files to assist in the conversion and transfer. Follow the WCA datamart conventions so as not to break any existing WCA ETL processes and to make it easy to locate customizations.

You must now create a WCA ETL process step and determine which ETL process flow this step should be a part of. Your steps should always be linked between the second to last step and the last step of the chosen flow. This ensures that the data required for the customized step is already processed. It is also important to remember that the last step of each flow must remain the last step of that flow. Do not link a customized step after the last step of any flow. Currently, new flows can not be created. This ensures that information relevant to perform your step is present and ready to use.

As a general example, if your step requires WCA data to complete your data transformation, and the WCA datamart was the result of some effectiveness calculation steps, your ETL step should be linked to an ETL process that follows the effectiveness calculations process flow.

The WCA ETL driver utility is provided as an alternative to using SQL within the DB2 warehouse step to accomplish your data ETL. The WCA ETL driver provides many useful features, including the ability to create a detailed daily trace log of all steps run without the need for additional customizations. See Chapter 8, “Using the ETL Driver”, on page 37 for information about how to construct ETL driver steps.

Extraction coding standards

You must follow these coding standards when you are using the extraction process:

- When you can add your own functions to the WCA datamart, prefix the customer-defined function name with *UX* to identify it as a user extension of the WCA datamart.
- When you can add stored procedures to the WCA datamart, prefix the stored procedure name with *UX* to identify it as a user extension of the WCA datamart.
- Create platform-specific scripting files to automate these types of changes. On the Microsoft® Windows platform, these files are known as *.bat (batch) files*. .
Prefix each of the script names with *UX*. When you name any of the environment variables, temporary pipes, files, or directories, use the *UX* prefix in its name.

Prefix the script file error messages with *UX* to help in diagnosing WCA problems. Use the error message scheme: **UX-*nnnn*Y**, where **UX** is the WCA error message prefix, *nnnn* is a 4-digit error code, and *Y* is replaced with **F** (fatal), **E** (error), **W** (warning), or **I** (informational).

- Create a IBM DB2 Warehouse Center script to automate the data Extract, Transform, Move, and Load (ETL) process for the customer-defined extensions. Prefix the customer-defined step, function, and externally called script file names with *UX*.

You can add steps to the end of the WCA ETL process so that you can run them in the same time cycle. Do not run them at the same time and compete for system resources.

You can add steps that are independent of the WCA ETL process. These scripts must not run at the same time as the WCA ETL scripts as it can affect the performance of the product.

Customizing ETL

This section discusses how you can customize ETL. The following topics are included:

- Creating a customized .sql file
- Defining the SQL statements
- Adding the execution commands
- Adding the customized .sql file to the proper location
- Creating a customized script file
- Adding customized script file error messages
- Creating a user-defined program to access the SQL file
- Maintaining the tables
- Adding a step in the Data Warehouse Center control database
- Scheduling the step to run

Creating a customized .sql file

Rather than create individual SQL statements to transform data from the source to the target to update datamart tables, you can create one .sql file that contains multiple DB2 update commands as SQL statements.

In this example, the DB2 command updates the `UX_MEMBERID` column in the `PRODUCT` table:

```
UPDATE WCA.PRODUCT SET UX_MEMBERID=(SELECT UX_MEMBERID FROM WCA.PRODUCT_TMP
WHERE WCA.PRODUCT.PRODUCT_ID= WCA.PRODUCT_TMP.PRODUCT_ID AND
WCA.ORDERITEMS_TMP.LASTUPDATE>=TIMESTAMP(WCA.PARAM_VALUE('TIME_CUT_OFF_PREV'))
WHERE PRODUCT_ID IN (SELECT PT.PRODUCT_ID FROM WCA.PRODUCT_TMP PT
WHERE PT.LASTUPDATE >= TIMESTAMP(WCA.PARAM_VALUE('TIME_CUT_OFF_PREV')))
```

Or, run the following DB2 create command to create a temporary table in the DB2 Data Warehouse Center control database and to extract the data from the online database. In the following example, the user-defined table UX_ACME_CUST_MAP table is created and then populated.

```
CREATE UX_ACME_CUST_MAP TABLE
(
    UX_ACMECARD_ID INTEGER NOT NULL,
    UX_CUSTTYPE INTEGER NOT NULL,
    UX_GEOCODE INTEGER NOT NULL,
    PRIMARY KEY (UX_ACMECARD_ID)
)
```

Defining the SQL statements

To define and add the command set to your SQL statement or statements in the .sql file:

1. Type the SQL statements that you want to run in a plain text file.
2. End each SQL statement with the semicolon (;) delimiter.
Any SQL statement (without returned result sets) can be included in the file. Also, the DB2 commands, such as RUNSTATS and REORG, are valid entries.
3. Save the file with a file extension of .sql.
4. Test the .sql file in the DB2 command-line window by typing:
db2 -tvf *sql_filename.sql*

Adding the execution commands

To add one or more execution commands:

1. *Only for the SELECT-INSERT and UPDATE SQL statements:* If the SQL statement involves large data transactions, you must add additional execution command tags for the SQL statements. Otherwise, skip to step 3.
2. Exit the saved .sql file after adding additional commands.
3. Check the SQL statement for any special requirements for using this feature.
4. Use the special command notations for global and local commands in the .sql file.
 - Global commands are identified by lines that start with:
--G--

Add global command tags, if needed. Multiple command lines can be supplied.
 - Local commands are identified by lines that start with
--L--

Add local command tags for required SQL statements. The local command only applies to the SQL statement to which it is associated. Insert the local command before the SQL statement it applies to.
5. Make sure that you end the SQL statement with the semicolon (;) delimiter.
6. Save the .sql file after editing.
7. Test the .sql file using the command-line environment.

Adding the file to the proper location

To add the .sql file to the proper location:

1. During production, different versions of the files for different data sources are saved under different subdirectories. The current directory is the directory where:

IWDA_DIR

Identifies the directory where WCA is installed. You set this directory location during installation.

wcs_source

Identifies the directory where the WebSphere Commerce data source is located.

2. Save the .sql file under the proper directory according to the version of the WebSphere Commerce data source.

Creating a customized script file

You can create a customized, platform-specific, scripting file that you can use to automate changes. For example:

1. Connect to the WCA datamart by typing the following SQL statement and specifying as parameters the WCA database name (WCA1001), and the user name and password of the administrator of the WCA datamart:

```
CONNECT TO WCA1001 USER USERNAME USING PASSWORD
```

2. Run the DB2 **-tvf** command to create and populate user-defined tables:
`db2 -tvf newupdates.sql`
3. Disconnect from the database.

On the Windows platform, these scripting files are known as *.bat (batch) files*.

Prefix each of the script names with UX. When you name any of the environment variables, temporary pipes, files, or directories, use the UX prefix in its name.

Adding customized script file error messages

Prefix your user-defined script file error messages with UX to help in diagnosing WCA problems. Use the error message scheme: **UX-*nnnn*Y**, where **UX** is the WCA error message prefix, *nnnn* is a 4-digit error code, and **Y** is replaced with **F** (fatal), **E** (error), **W** (warning), or **I** (informational).

Errors can be handled by writing messages to the Data Warehouse Center feedback file identified by the environment variable \$VWP_LOG. This file has the format of:

```
<RC>Integer</RC>  
<MSG>Text</MSG>  
<COMMENT>Text</COMMENT>
```

This file should be removed at the beginning and created only if there is an error. If the integer is non-zero, Warehouse Center assumes an error.

Creating a user-defined program to access the SQL file

All ETL steps in the DB2 Data Warehouse Center are implemented as calls to the ETL driver using *user-defined programs*. You can create a IBM DB2 Warehouse Center script to automate the data Extract, Transform, Move, and Load (ETL) process for the customer-defined extensions. Prefix the customer-defined step, function, and externally called script file names with UX.

To create an IBM DB2 Warehouse Center user-defined program, supply the step name as a parameter.

Maintaining the tables

To create a step in the IBM DB2 Warehouse Center to maintain the table, you have two options:

- Using IBM DB2 Warehouse Center to schedule the process

You can add steps that are independent of the WCA ETL process by using IBM DB2 Warehouse Center to schedule the process on a regular basis. However, IBM DB2 Warehouse Center must complete the step before the regular datamart extraction process runs. These scripts must not run at the same time as the WCA ETL scripts as it can affect the performance of the product.

If both processes run at the same time, performance can be adversely affected.

- Attaching the step to the standard extraction process

You can attach the step to the standard extraction process so that it runs at the end of the WCA ETL process. Do not run them at the same time and compete for system resources.

Before making your decision, review the Business Intelligence tutorial. Click **Start** —> **Programs** —> **IBM DB2 Set-up Tools**—>**First Steps** —>. In the First Steps window, click **Work with Tutorials** and then click *Business Intelligence Tutorial: Introduction to the Data Warehouse Center*. Review the following lessons:

- Defining data transformation and movement
- Testing IBM DB2 Warehouse Center steps
- Scheduling IBM DB2 Warehouse Center processes

Adding a step in the Data Warehouse Center control database

To add a step in the Data Warehouse Center control database:

1. Login to the Data Warehouse Center control database.
2. On the Process window, either open an existing process or create a new process.
3. Click the external program icon from the list that is displayed.
4. Select **WCA Extraction** —> **WCAUPD**, and then add the selected external program from the Process window.
5. Add a target table selected from the **Warehouse Target** —> **Advanced Target Tables** → **Tables** list.
6. Add a data link from the added external program to the added target table.
7. Double-click the newly added external program icon. A Properties window for this new external program opens.
8. On the User Defined Program window, provide a valid name in the **Name** field.
9. On the Parameters windows, edit the value for the following two parameters:
 - For the SQL parameter, add the name of the .sql file.
 - For the Option parameter, type a value of **Y**, **N**, or some other string. If you provide the value **Y**, the ETL driver automatically commits a transaction after each SQL statement in the file is run. Otherwise, all SQL statements in the .sql file are run without any special commit unless the **commit** statement is inserted whenever a **commit** statement is required.
10. On the Process Options window, select the proper Agent site. Use .
11. Click **OK** to save the changed property information.

12. On Process windows, save the added external program step and then close the window.

Scheduling the step to run

To schedule the step to run:

1. Add the step in the proper location in the execution sequence.
2. Promote the step to the Test mode and run the test for this step.
3. If the test is successful, promote the step into Production mode. After successful promotion, the step is ready for scheduled execution.

Chapter 12. Customizing replication

WCA handles replication and extraction of data for these types of e-commerce areas: accounts and contracts, campaigns and initiatives, calendar (time period data), catalog hierarchy, coupons and discounts, members, product advisor metaphors, offers, order items and orders, and requests for quote. You can collect e-commerce transaction data from other than WebSphere Commerce, such as Web site clickstream data.

IBM DB2 Warehouse Center performs the data consolidation and integration. All data sources can be collected and integrated into the WCA datamart. WCA provides the analysis of the data using data mining techniques. And finally, the WCA datamart becomes the main data source for reporting. The reporting application allows you to create and view reports that incorporate the data mining results.

Overview of replication

WCA replicates WebSphere Commerce transactional data to the WCA database by using DB2 replication technology. WCA transforms data replicated from WebSphere Commerce into *multidimensional form* and then stores the replicated data into the WCA datamart. Because transaction data can be very voluminous and analytics can be very processing intensive, WCA should be located on a different machine from WebSphere Commerce for performance reasons.

The WCA database accommodates not only the WCA datamart schema, but it is also the staging area that is used for replication and any *secondary datamart*. The secondary datamart consists of tables and views that might be introduced by the reporting application.

Capture and Apply

DB2 replication technology has two main steps:

Capture

The Capture program collects all the changes (inserts, updates, and deletes) that occur on the source tables in change data tables that are stored in the WebSphere Commerce database. If Capture is not running, no changes to the registered source tables are captured at the replication source systems.

Change data tables are stored in table spaces. When the table space reaches the size limit, Capture cannot capture any additional changes. Each change data table has its own table space.

Apply The Apply program brings change data to the target database and stores the data in the staging area. For WCA, the Apply program is located at the target database system (the WCA datamart). The Apply program can have its control tables located either locally or remotely. When Apply is running on the same machine as the target database, it pulls the data from the source. When Apply is running remotely from the target database, it pushes the data to the target. Only a limited number of rows can be propagated by Apply at one time. Apply automatically splits what needs to be propagated into several mini-subscriptions in several mini-cycles.

Normally, Capture and Apply can be either *on demand* (scheduled to occur at a specified time) or *continuous* (the work is uniformly distributed over time). WCA uses continuous Capture. After a replication cycle completes, another follows immediately rather than waiting for a continuous Apply. On-demand Capture and Apply helps distribute the replication load over a period of time.

Staging tables

Staging tables are replica of the WebSphere Commerce data source. The replication steps are designed using the replication tool of the IBM DB2 Warehouse Center to populate data into the staging tables. The WCA control database hosts all replication and extraction steps, and coordinates their scheduling and execution.

WCA replicates numerous WebSphere Commerce tables into its staging area. The replicated data is stored in staging tables on the WCA machine. The replicated tables are later used as the source for the extract, transform, move, and load (ETL). The staging tables make use of the ETL steps to populate data into the WCA datamart.

All staging tables are based on the *condense* replication option. The condensed staging table contains only the most current value for each row from the source table. A *noncondensed table* contains all changes made to each row in the source table, representing the history of changes to each row.

The name of the temporary staging table is the same as the source table name or is closely associated with the source table name. For example, the temporary staging table of the ADDRESS table is ADDRESS_R.

Each staging table has a **LOAD_STATUS** field. Load status checks whether the data is inserted into the WCA staging table. This field is a WCA system-defined field that is used when adding data.

In addition, the IBM DB2 Warehouse Center replication tool automatically adds these fields to each staging table:

IBMSNAP_INTENTSEQ

The log or journal record sequence number that uniquely identifies a change. This value is globally ascending.

IBMSNAP_OPERATION

A character value of I, U, or D, indicating an insert, update, or delete record, respectively. During the processing of ETL steps, any data having time information in this field that is between **time_cut_off** and **time_cut_off_previous** indicates data insertion or updates.

IBMSNAP_LOGMARKER

The approximate commit time at the source server. This column is always NULL after a full refresh. Incremental data processing is based on the time information of this field.

IBMSNAP_COMMITSEQ

The log record sequence number of the captured COMMIT statement.

Replication coding standards

For WCA, all replication steps must include the prefix **R WCS_**, **R WSA_**, **UX_**.

Because replication and extraction processes go hand in hand when performing the data extraction and transformation, also see “Extraction coding standards” on page 58.

Replication customization tasks

WCA automates the majority of the replication tasks. WCA provides predefined replication setup script files for all the WebSphere Commerce tables that WCA replicates. WCA converts all the WebSphere Commerce tables to your own environment before running the scripts.

The replication customization tasks include:

- Creating a new replication table
- Increasing the log space
- Calculating table space sizes
- Modifying replication control tables
- Starting the Capture process
- Registering the replication process
- Running the replication steps
- Scheduling replication and extraction

Creating a new replication table

To create a new replication data source table, do the following:

1. Register the replication for the table in the WCA database using either the DB2 Control Center (define as the Replication source), or the DB2 DJRA tool.
2. Create a replication step in the DB2 Data Warehouse Center by doing the following:
 - a. Create a stage table from the Replication Properties window.
 - b. Add the **Load-Status** column with integer type to the staging table.
3. Link the new replication step to the other replication step in the DB2 Warehouse Center.
4. Promote the new step to production mode.

Increasing the log space

The increase in log space needed for the replication source tables depends on the number of replication sources that are defined, the row length of the replication source, the number of changes to those tables, and the number of columns to be updated. Typically, the log space should be at least three times larger than the original log space needed for these tables.

Calculating table space sizes

The rep.sql file allocates table space for the change data (CD) tables, which are used to store data temporarily during replication. To determine the minimum size for a change data (CD) table, use the following formula:

$$\begin{aligned} \text{minimum_CD_size} = & \\ & ((21 \text{ bytes}) + \text{sum}(\text{length of all registered columns})) * \\ & (\text{number of inserts, updates, and deletes to source table}) * \\ & (\text{exception factor}) \end{aligned}$$

To determine the table space size for the table, use the following formula:

Table space size = *minimum_CD_size* ÷ 4096

Notes:

1. When calculating the number of bytes of data replicated (the first line of the formula), include 21 bytes for overhead data added to the CD tables by the Capture program.
2. To determine the number of inserts, updates, and deletes to the source table (the second line of the formula), use the following information:

When you estimate the space requirements for the CD tables, you do not need to allow space for all the records that exist in the source table associated with the CD table. The first time replication occurs, the data is exported into *_R* tables on the apply side, bypassing the CD tables. Therefore, the CD tables need only enough space for the number of records that you estimate will be updated, inserted, or deleted during the time between replication runs. To estimate the number of records, use the following formula:

$$\begin{aligned} \text{number of records updated} = & \\ & (\text{average number of rows}) * \\ & (\text{factor for inserts/updates/deletes}) * \\ & (\text{number of days capture data is held}) \end{aligned}$$

Where:

- (average number of rows) can be calculated using SQL similar to the following example:
(count(*)/count(distinct date(lastupdate))) from orders
- For (factor for inserts/updates/deletes), use a factor between 6 and 10 because of the way in which WebSphere Commerce issues updates, deletes, and inserts. (This value is an estimate.)
- (number of days capture data is held) is the number of days the CD table data will be kept.

For example, if the Capture program prunes applied rows from the CD table once daily, the interval is 24 hours. If the rows in the CD table are 100 bytes long (plus the 21 bytes for overhead), and 100,000 records are updated during a 24-hour period, the storage required for the CD table is about 12 MB. (This example uses an exception factor of 1.)

The following table shows table space sizes for the replication source tables, calculated with a value of 1000 updates applied during the interval and an exception factor of 1:

Table 6. Table space sizes for replication source tables

WebSphere Commerce Source Table	CD Table	WebSphere Commerce Table Space	CD Table Size (includes 21 bytes)	Minimum CD Size	Table Space (CD size ÷ 4096)
ADDRESS	CD_ADDRESS	TSADDRESS	3,221	3,221,000	787
ATTRIBUTE	CD_ATTRIBUTE	TSATTRIBUTE	1,145	1,145,000	280
CAMPAIGN	CD_CAMPAIGN	TSCAMPAIGN	1,285	1,285,000	314
CATALOG	CD_CATALOG	TSCATALOG	545	545,000	134
CATALOGDSC	CD_CATALOGDSC	TSCATALOGDSC	33,749	33,749,000	8,240
CATENTDESC	CD_CATENTDESC	TSCATENTDESC	131,677	131,677,000	32,148

Table 6. Table space sizes for replication source tables (continued)

WebSphere Commerce Source Table	CD Table	WebSphere Commerce Table Space	CD Table Size (includes 21 bytes)	Minimum CD Size	Table Space (CD size ÷ 4096)
CATENTREL	CD_CATENTREL	TSCATENTREL	617	617,000	151
CATENTRY	CD_CATENTRY	TSCATENTRY	1125	1,125,000	275
CATENTSHIP	CD_CATENTSHIP	TSCATENTSHIP	125	125,000	31
CATGPENREL	CD_CATGPENREL	TSCATGPENREL	307	307,000	75
CATGRPDESC	CD_CATGRPDESC	TSCATGRPDESC	34,007	34,007,000	8,303
CATGRPREL	CD_CATGRPREL	TSCATGRPREL	307	307,000	75
CATTOGRP	CD_CATTOGRP	TSCATTOGRP	53	53,000	13
CMPGNINTV	CD_CMPGNINTV	TSCMPGNINTV	29	29,000	8
CPGNLOG	CD_CPGNLOG	TSCPGNLOG	813	813,000	199
CPGNSTATS	CD_CPGNSTATS	TSCPGNSTATS	237	237,000	58
CURCONVERT	CD_CURCONVERT	TSCURCONVERT	69	69,000	17
FFMCENTDS	CD_FFMCENTDS	TSFFMCENTDS	32,813	32,813,000	8,011
FFMCENTER	CD_FFMCENTER	TSFFMCENTER	287	287,000	71
ICCNCPNLNK	CD_ICCNCPNLNK	TSICCNCPNLNK	293	293,000	72
ICCONSTRNT	CD_ICCONSTRNT	TSICCONSTRNT	56	56,000	14
ICEXPLFEAT	CD_ICEXPLFEAT	TSICEXPLFEAT	175	175,000	43
ICFEATPROP	CD_ICFEATPROP	TSICFEATROP	351	351,000	86
ICKNOWDESC	CD_ICKNOWDESC	TSICKNOWDESC	1053	1,053,000	258
ICKNOWLEDG	CD_ICKNOWLEDG	TSICKNOWLEDG	153	153,000	38
ICMETAPHOR	CD_ICMETAPHOR	TSICMETAPHOR	435	435,000	107
ICMETAREG	CD_ICMETAREG	TSICMETAREG	229	229,000	56
ICMREGDESC	CD_ICMREGDESC	TSICMREGDESC	257	257,000	63
IITEM	CD_IITEM	TSIITEM	353	353,000	87
INITIATIVE	CD_INITIATIVE	TSINITIATIVE	33,675	33,675,000	8,222
INTVMPE	CD_INTVMPE	TSINTVMPE	33	33,000	9
LISTPRICE	CD_LISTPRICE	TSLISTPRICE	116	116,000	29
MATYPE	CD_MATYPE	TSMATYPE	605	605,000	148
MPE	CD_MPE	TSMPE	351	351,000	86
MPETYPE	CD_MPETYPE	TSMPEETYPE	343	343,000	84
ORDERITEMS	CD_ORDERITEMS	TSORDERITEMS	1,179	1,179,000	288
ORDERS	CD_ORDERS	TSORDERS	751	751,000	184
ORDSTAT	CD_ORDSTAT	TSORDSTAT	872	872,000	213
PASTATS	CD_PASTATS	TSPASTATS	41	41,000	11
PCSTATS	CD_PCSTATS	TSPCSTATS	41	41,000	11
PESTATS	CD_PESTATS	TSPESTATS	105	105,000	26
SASTATS	CD_SASTATS	TSSASTATS	45	45,000	11
SHIPMODE	CD_SHIPMODE	TSSHIPMODE	6,151	615,000	151
STADDRESS	CD_STADDRESS	TSSTADDRESS	2,759	2,759,000	674

Table 6. Table space sizes for replication source tables (continued)

WebSphere Commerce Source Table	CD Table	WebSphere Commerce Table Space	CD Table Size (includes 21 bytes)	Minimum CD Size	Table Space (CD size ÷ 4096)
STORE	CD_STORE	TSSTORE	853	853,000	209
STOREENTDS	CD_STOREENTDS	TSSTOREENTDS	32,817	32,817,000	8,012
STORELANG	CD_STORELANG	TSSTORELANG	32	32,000	8
STORETRANS	CD_STORETRANS	TSSTORETRANS	33	33,000	9
USERDEMO	CD_USERDEMO	TSUSERDEMO	764	764,000	187
USERS	CD_USERS	TSUSERS	1,864	1,864,000	456
USRTRAFFIC	CD_USRTRAFFIC	TSUSRTRAFFIC	99,343	99,343,000	24,254
WTAXINFO	CD_WTAXINFO	TSWTAXINFO	137	137,000	34
Total					103,330

Modifying replication control tables

During the Replication Setup for Source Databases configuration step, replication control tables and table spaces are required on the server hosting the data source. See "Configuring replication" in the *IBM WebSphere Commerce Analyzer Installation and Configuration Guide, Version 5.5* for more information.

The file `%IWDA_DIR%\bin\db2\replication.bat` controls the replication setup for WebSphere Commerce Server.

The following files are used during replication setup:

rep.sql

Creates the replication control tables in WebSphere Commerce

rep_as400.sql

Creates the replication control tables in WebSphere Commerce (for OS/400®).

repContrlTables.sql

Creates the replication control tables in WCA.

Use the Create Replication Control Tables feature of *DJRA* to create the script of the replication control tables.

To change the `rep.sql` file, proceed as follows:

1. Make a copy of the `replication.bat` file in the `bin\db2` directory.
2. Edit the replication batch file:
 - a. Locate the following line and comment it out.

```
@if exist %IWDA_DATA%\tmp\rep.sql del %IWDA_DATA%\tmp\rep.sql
```
 - b. Locate the following line and comment it out.

```
@ %cmd% >%IWDA_DATA%\tmp\rep.sql 2>&1
```

Important: The output file must be named `rep.sql` and must be placed in the `%IWDA_DATA%\tmp` directory.

3. Open a command line window.

- a. Change directories to the subdirectory for the data source you are using (for example, 55be_ext).
- b. Type the following command:

```
rep TSPATH WCSSHEMA TS >%IWDA_DATA%\tmp\rep.sql
```

Where:

- *TSPATH* is the path to the tablespace location.

Note: The *TSPATH*-value must end with a file separator.

- *WCSSHEMA* is the WebSphere Commerce schema name.
 - *TS* is the sizing value.
4. Change the directory to %IWDA_DATA%\tmp.
 5. Modify the rep.sql script as required before starting the Replication Setup for Source Databases configuration step.

Starting the Capture process

The only task you have to do for replication on the WebSphere Commerce server is to start the Capture program just before the initial extraction is done. Note that this is a one-time activity.

Refer to the procedure for setting up and starting the Capture program in the *IBM WebSphere Commerce Analyzer Installation and Configuration Guide, Version 5.5*.

Registering the replication process

A script to register replication for a new table is generated using the IBM DB2 Replication Center. The WCA Configuration Manager tool registers all replication Capture tables in the WebSphere Commerce database.

Running the replication steps

The WCA replication steps have been designed using the IBM DB2 Warehouse Center replication tool. The replication steps are chained and processed by the link order. The replication steps populate data into the staging tables that are the replica of the WebSphere Commerce data source. Then, the staging tables make use of the ETL (Extract, Transform, Move, and Load) steps to populate data into the datamart of the WCA system.

The **Start Replication** step is the first step of the WCA ETL processing. WCA runs either continuous replication processing or ETL processing. Continuous replication processing avoids moving large volumes of data at one time because it moves the data continuously as the data becomes available.

The **Replication History** step provides information to monitor the replication Capture server. When the latest synch time of the replication is less than the previous replication start time, either the Capture server is down or no new data was in the WebSphere Commerce system.

The WCA.PARAMETERS table includes replication information, such as:

replication_start

The replication starting time. This parameter is a controller step that helps make replication processing continuous.

replication_end

The replication ending time. This parameter is a controller step that helps make replication processing continuous.

replication_succeed

Set to 1 when the replication starts. Set to 0 when the replication ends.

request_etl_process

Set to 1 when the replication steps are processing to indicate that ETL processing should wait until all of the replication steps are processed. Set to 0 when the ETL step starts. If the ETL step is processing, replication processing will wait until all the ETL steps are processed. If no ETL step is processing, the replication processing starts immediately. This parameter indicates that ETL processing waits until all the replication steps are completed.

Scheduling replication and extraction

If there is no replication step processing, the ETL process that corresponds to a user-defined schedule starts. You can schedule the processes to run as frequently as you want, but it might take some time for them to run.

The first time replication and extraction are run, all existing data is replicated and extracted. During subsequent runs, only new data is replicated and extracted.

Refer to the instructions for scheduling replication and extraction in the *IBM WebSphere Commerce Analyzer Installation and Configuration Guide, Version 5.5*.

Chapter 13. Customizing data mining

Professional > Business The WCA Data Mining component handles the development and application of data mining models to the WCA datamart. To develop a custom data mining base to complement the WCA supplied data mining base, you can use the **Professional > Business** IBM DB2 Intelligent Miner for Data tool.

Overview of data mining

WCA provides data mining functions through the **Professional > Business** DB2 Intelligent Miner for Data tool. The DB2 Intelligent Miner for Data tool enhances the capability of the WCA server for organizing, aggregating, summarizing, and mining the data.

The following diagram shows the data mining features of WCA:

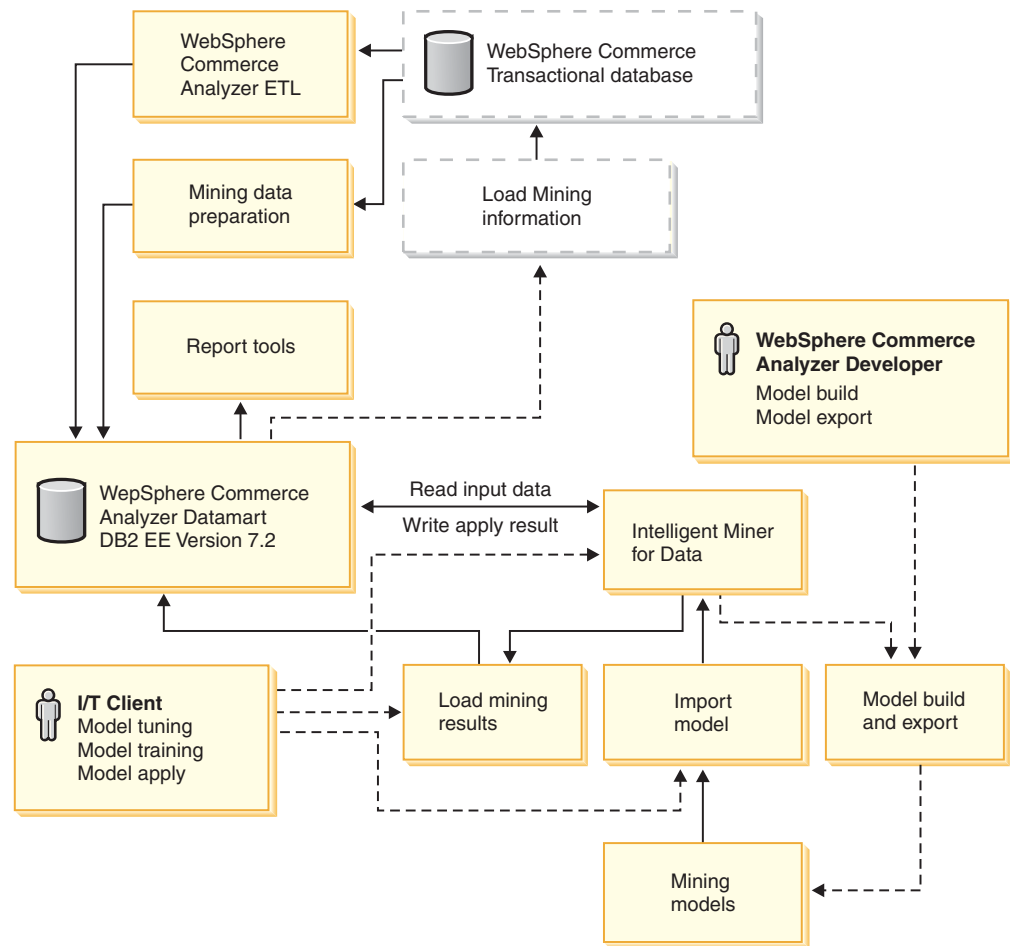


Figure 4. The WCA data mining features

WCA provides the following data mining features:

- Support for answering a set of data mining-related business questions. WCA provides mining models based on these questions.

- Setup of the environment for data mining during WCA configuration. This setup includes data preparation, table setup, and mining model setup.
- **Professional > Business** Support for users to perform data mining operations through the DB2 Intelligent Miner for Data GUI. These data mining operations generate the information necessary for answering the data mining-related business questions from the data.
- Utilities for loading the **Professional > Business** DB2 Intelligent Miner for Data mining results into the WCA datamart.
- Reports that incorporate the data mining results.

The WCA installation installs the **Professional > Business** DB2 Intelligent Miner for Data server on the same computer as where it installs the WCA server. You create the data mining-related database tables in the WCA datamart by using the WCA schema. **Professional > Business** DB2 Intelligent Miner for Data reads input data from the WCA datamart. After you run the data mining sequences, use the utilities provided with WCA to load the data mining results back into the WCA datamart. You can incorporate these data mining results into the business reports.

Examples of database model for clustering

The following information answers the question "What is the relationship between the shoppers and total order values":

Input data table:	MEMBSUMS
Sample data set:	MEMBSUBSAMP
Model name:	wcamemtord61
Result file:	wcamemtord61.xml
Apply output table:	MEMAPPLY61
Assigned model ID for the model	90000-99999



Register data tables in DATATABS table

data_id	data_name	data_type	usage_type	data_desc	data_location	sample_pct
1001	MEMBSUMS	0	0	Summary table for member usage activities	wca	100
1002	MEMBAPPLY61	0	1	Output table for model apply	wca	100
1101	wcamemtotord61.xml	2	1	Result from model_id 1051		
1011	MEMBSUMSAMP	0	1	Sampled data set of MEMBSUMS	wca	50

You must type data in the **data_name** and **data_location** columns on one line. If it is broken here, it is because of the page-layout limitations.

Register model in the MODELS table

model_id	model_name	model_type	model_alg	model_desc	data_id	file_id	res_file_name	file_location
1051	wcamem totord61	1	0	Relation- ship between shippers and total order values	1011	1101	memord val61.xml	

- You must write data in the **model_name**, **res_file_name**, and **file_location** columns on one line. If the lines are broken, it is because of the page-layout limitations.
- The **model_name** should use the setting name defined in   IBM DB2 Intelligent Miner for Data for model training.
- The **data_id** and **file_id** in the MODELS table refer to the **data_id** in the DATATABS table for the corresponding data set.

Register apply result in the APPTABS table

table_id	model_id	data_id	table_name	table_type	status
1002	1051	1001	MEMAPPLY61	2	0



- The **model_id** in the APPTABS model refers to the **model_id** in the MODELS table for the model it applied.
- The **table_id** and **data_id** in the APPTABS model refer to the **table_id** and the **data_id** in the DATATABS table for the corresponding data set in which it is used.

Data mining customization tasks

The data mining customization tasks include:

- Creating data mining models
- Registering new data mining models
- Determining the recency, frequency, and monetary value

Creating data mining models

  You can create data mining models (also known as *settings objects*) by using DB2 Intelligent Miner for Data wizards, and you can modify them using settings notebooks.

Data mining models have built-in relationships to other objects in the same mining base. For example, a mining settings object requires that you specify the input data, either by creating a new data object or by referring to an existing one. The mining settings object represents parameters that you specify for a mining function. One of these parameters is the name of the data object, which serves as a logical description of the input data.

Any user-defined input or output table uses the default data mining table schema wcamng. Any user-defined input or output table for data mining should use the

prefix *UX_* (for user extension). If you define the setting name for the model training as *ux_xxxx*, the corresponding setting for applying this model will be named *ux_xxxxa*.

To create new data mining models, refer to the *Using the Intelligent Miner for Data DB2* documentation.

Registering new data mining models

Data that is used in mining is identified and grouped into a set of tables or views. Use these tables and views as the input data for mining.

When you add a new data model, register it in the appropriate control tables. You must work with the following tables:

- DATATABS
- MODELS
- APPTABS

DATATABS

Tables for both input and output are modeled in the data control table. Register each data set with one entry in this table.

Column:

Column Name	Data Type	Note
data_id	integer NOT NULL	Unique ID for the registered data.
data_name	char(32) NOT NULL	Name of the data set. <ul style="list-style-type: none"> • The name of the table or view (for table or view). • The name of the file (for files).
data_desc	varchar(254)	Description of the data set.
data_type	integer	Type of data set. Valid values are: <ul style="list-style-type: none"> 0 table 1 view 2 file
usage_type	integer	Usage of the data set. Valid values are: <ul style="list-style-type: none"> 0 input 1 output 2 input/output
app_type	integer	Type of application with which it is associated. This column is an optional column.
data_location	varchar(254) NOT NULL	One of the following values: <ul style="list-style-type: none"> • The database name (for the table and view). • The full file path (for files).
sample_pct	integer	Sample size of the data set, if available.

Index:

Name	Column Name	Type
p_datatabs	data_id	Primary key

MODELS

The MODELS table manages the information about the models, file names, and related file locations.

Column:

Column Name	Data Type	Note
model_id	integer NOT NULL	Unique ID for a model.
model_name	char(32)	Name name of the model.
model_desc	varchar(254)	Description of the model.
model_type	integer	Type of model.
model_alg	integer	ID that indicates the algorithm used for building the model.
data_id	integer	ID of the data that was used for building the model.
per_id	integer	Date that the model was last trained.
res_file_name	varchar(32)	Name of the file that contains the Predictive Model Markup Language (PMML).
file_location	varchar(254)	Location where the file is saved.

Index:

Name	Column Name	Type
p_resfiles	model_id	Primary key

APPTABS

The APPTABS table holds information about model apply tables. Each model apply must register one entry in this table.

Column:

Column Name	Data Type	Note
table_id	integer NOT NULL	Unique ID for a registered table.
model_id	integer NOT NULL	ID for the model apply is registered in the models table.
data_id	integer	ID of the data that was used for generating the apply result table.
table_name	char(32) NOT NULL	ID of the table that holds the apply results.

Column Name	Data Type	Note
table_type	integer	Primary key of the output table. Valid values are: 1 initiative_id 2 member_id 3 member_id plus order_id 4 member_id plus product_id 5 member_id plus category_id 9 Any user-specified table
status	char(1)	Status that indicates whether the table is actively used.

Determining the recency, frequency, and monetary value

Recency, Frequency, and Monetary (RFM) is a method of ranking data based on variables that represent the most recent visits to the store, the greatest number of orders placed, and the most money spent on orders.

In WCA, Recency, Frequency, and Monetary (RFM) is applied to the orders associated with contracts and accounts. The following conditions determine RFM:

- The most recent date of the shopping activity.
- The greatest number of orders placed.
- The most amount of money spent on the orders.

The data is ranked according to each variable and then divided into five equal bins. Each contract or account is assigned a number between one and five, which corresponds to their bin assignment. The contract or account with the most recent shopping activity, the greatest number of orders, and the most money that is spent will have an RFM value of 555. The contracts or accounts with the least activity, the smallest number of orders, and the least amount spent have an RFM value of 111. RFM requires a minimum of five records (the number of bins) to run successfully.

Appendix A. WCA parameters

This section lists the pre-defined WCA parameters. It also discusses how to create or change WCA parameters and how the parameter table might look after configuration.

List of pre-defined WCA parameters

Parameters affect how the datamart looks after you perform an extraction. These parameters control the execution of the system and also control the analytic conclusions. Parameters are modified using the WCA Parameter Manager. The following are the column names of the WCA.PARAMETERS table:

Note: This parameter list was current at the time of publication.

ROW_NUM

This parameter provides the system-generated ID used with PARAM_TYPE as the primary key for the WCA.PARAMETERS table.

PARAM_TYPE

This parameter is a WCA-defined parameter that is used by the extraction steps, replication steps, and third-party reporting tools. One or more parameters enable you to tailor the behavior of the default extraction. If a PARAM_TYPE has more than one PARAM_VALUE, represent each PARAM_VALUE in a separate row. The ROW_NUM column distinguishes between multiple values for each PARAM_TYPE. For example:

Table 7. Parameter table example

ROW_NUM	PARAM_TYPE	PARAM_VALUE
1	COMPLETED_STATUS	Completed
2	COMPLETED_STATUS	Shipped

This table shows that the PARAM_TYPE COMPLETED_STATUS has two values: Completed and Shipped.

PARAM_VALUE

This parameter is a string representation of the parameter type. For integers, use the cast statement.

The following parameters are the currently defined parameters for the WCA datamart:

ABANDONED_MINUTES

This parameter defines the time period that must elapse after a member updates a pending order in order to abandon that order. The parameter ABANDONED_ORD_STATUS defines the order status values that are considered pending. This parameter affects the FACT_ORDERS.ABANDONED column as well as the FACT_ORDERITEMS.ABANDONED column.

The default value to consider an order abandoned is 60.

ABANDONED_ORD_STATUS

This parameter defines the set of ORDER.STATUS and ORDERITEMS.STATUS values from the WebSphere Commerce database that are considered pending orders. Use this parameter to determine whether to abandon an order or order

item. This parameter affects the FACT_ORDERS.ABANDONED column and the FACT_ORDERITEMS.ABANDONED column.

The following are the default values:

- A = Requires Review
- E = CSR Edit
- I = Submitted Order
- L = Low Inventory
- M = Payment Initiated
- P = Pending Order
- W = Waiting for Approval
- X = Cancelled

ANALYSIS_CATALOG_ID

This parameter has one row per catalog ID. It supplies the reporting tools with the catalog IDs of the catalogs created in WebSphere Commerce to track metrics in categories. This catalog must be a tree that has one-to-many relationships from the catalog to categories, categories to subcategories, and subcategories to products. You can refer to this type of analysis as the *Classification Analysis of Catalogs*. This parameter can have multiple values.

BROKER_STORE_ID

This parameter indicates the STORE_ID of stores which are categorized as Broker Stores. This selection is made during the Select Online Stores and the Languages and Currencies for Reports configuration step. There is one row per broker store. See "Selecting stores and languages" in the *IBM WebSphere Commerce Analyzer Installation and Configuration Guide, Version 5.5*.

CONTRACT_ACTIVE

This parameter determines which CON_STATUS_ID in the WCA.CONTRACT table indicates that a contract still is active. The descriptions of the status codes are in the WCA.CON_STATUS_REF table. The default value is 3. This parameter can have multiple values.

CONTRACT_CANCELLED

This parameter determines the CON_STATUS_IDs in the WCA.CONTRACT table that represent cancelled contracts. You can find the descriptions of the status codes in the WCA.CON_STATUS_REF table. The default value is 5. This parameter can have multiple values.

CONTRACT_IN_PREPARATION

This parameter determines which CON_STATUS_ID in the WCA.CONTRACT table indicates that a contract still is being prepared. The descriptions of the status codes are in the WCA.CON_STATUS_REF table. The default value is 0. This parameter can have multiple values.

CPN_REDEEMED_ORDER_STATUS

The default values are:

- 4 = Payment Initiated
- 6 = Payment Authorized
- 7 = Order Shipped

DMT_PROSPECT_ORD_STATUS

This parameter defines a set of values for the ORDERS.STATUS and ORDERITEMS.STATUS columns from the WebSphere Commerce database. When a member owns orders in any of these states and you do not already consider the member a purchaser, then you consider that member a *prospect*. This parameter affects the value of the WCA.MEMBER.DR_MEMBER_TYPE_ID.

The following are the default values:

- A = Requires Review
- E = CSR Edit
- I = Submitted Order
- M = Payment Initiated
- P = Pending Order
- Q = Order Template
- W = Waiting for Approval
- X = Cancelled
- Y = Private Requisition List
- Z = Shareable Requisition List

DMT_PURCHASER_ORD_STATUS

This parameter defines a set of values for the ORDERS.STATUS and ORDERITEMS.STATUS columns from the WebSphere Commerce database. When a member owns orders in any of these states, that member is considered a *purchaser*. There is one row for each ORDER.STATUS value. By default, WCA defines a purchaser as a user that has at least one completed order. This parameter affects the value of the WCA.MEMBER.DR_MEMBER_TYPE_ID.

The following are the default values:

- B = Back-ordered
- C = Payment Authorized
- D = Deposited
- F = Ready For Remote Fulfillment
- G = Waiting For Remote Fulfillment
- L = Low Inventory
- R = Ready For Remote Fulfillment
- S = Order Shipped

EXTRACTION_COUNT

This parameter defines the number of extractions that have been run. This parameter is an IBM reserved parameter. *Do not* change it.

EXTRACTION_SUCCEED

This parameter indicates whether the extraction process completes successfully. If the extraction process completes successfully, this parameter carries a value of 0. If the process is unsuccessful, it carries a value of 1. This parameter is an IBM reserved parameter. *Do not* change it.

FE_EFFECTIVE_MINUTES

This parameter defines the time period in which a Suggestive Sell initiative might affect a decision by a buyers to purchase a certain product. If a member does not place the advertised product in the shopping cart within the time period defined, this initiative is not eligible to claim the revenue for that product even if it is purchased later.

This parameter affects the population of the following WCA.FACT_EVENT columns:

- SLS_VAL_CLKS
- STR_SLS_VAL_CLKS
- RPT_SLS_VAL_CLKS
- NUM_CKS_TO_ORD

The default value for a time period is 60 minutes.

FE_EFFECTIVE_ORD_STATUS

This parameter defines a set of values for the ORDERS.STATUS and ORDERITEMS.STATUS columns in the WebSphere Commerce database. When

a Suggestive Sell initiative advertises a particular product to a particular member in a time period defined by the FE_EFFECTIVE_MINUTES parameter, the member places that product in a shopping cart (pending order), and then subsequently buys that product, that initiative is eligible to claim responsibility for the revenue generated by that product. There might be many initiatives eligible for the same order item. Only one initiative gets credit for it.

This parameter affects the population of the following WCA.FACT_EVENT columns:

- SLS_VAL_CLKS
- STR_SLS_VAL_CLKS
- RPT_SLS_VAL_CLKS
- NUM_CKS_TO_ORD

The following are the default values:

- B = Back-ordered
- C = Payment Authorized
- D = Deposited
- F = Ready For Remote Fulfillment
- G = Waiting For Remote Fulfillment
- L = Low Inventory
- R = Ready For Remote Fulfillment
- S = Order Shipped

FM_EFFECTIVE_MINUTES

This parameter defines the number of minutes allowed to elapse between a user using a metaphor and a user buying a product pushed by that metaphor and still give credit for the purchase to the metaphor. Use this parameter to populate the sales value columns in the WCA.FACT_METAPHOR table. You can modify this parameter. The default value is 60 minutes.

FM_EFFECTIVE_ORD_STATUS

This parameter defines a set of values for the ORDERS.STATUS and ORDERITEMS.STATUS columns in the WebSphere Commerce database. When a product metaphor is used by a user and the user places the product in their shopping cart within the time period specified by FM_EFFECTIVE_MINUTES, then that metaphor is credited for producing the sale of that product.

The WCA.FACT_METAPHOR.STR_SLS_VAL_META and WCA.FACT_METAPHOR.RPT_SLS_VAL_META columns track the revenue generated by the metaphor. You can use many metaphors, but only the last metaphor to display the product to the user receives the credit. The values used must indicate that the revenue is realized.

The following are the default values:

- B = Back-ordered
- C = Payment Authorized
- D = Deposited
- F = Ready For Remote Fulfillment
- G = Waiting For Remote Fulfillment
- L = Low Inventory
- R = Ready For Remote Fulfillment
- S = Order Shipped

MEMBER_ADDRESS_TYPE_ID

This parameter helps WCA decide which address record to use to populate the address-based columns in the MEMBER table. You can compare this parameter to the ADDRESS_TYPE_ID field in the ADDRESS table.

MINING_APPLY_INTERVAL

This parameter defines how often the mining apply process is executed. The interval is expressed as a number of days.

MINING_APPLY_TIME

This parameter defines the time stamp when the mining apply is run successfully. If the mining apply operation is successful, this time stamp will carry the value of `TIME_CUT_OFF_LOCAL` for this extraction run. This parameter is an IBM reserved parameter. *Do not* change it.

MINING_BASE_NAME

This parameter defines the name of the mining base that carried the pre-defined mining models. You can change this parameter using the Configuration Manager.

MINING_PASSWORD

This parameter defines the password for the user who runs the mining operations. You can change this parameter using the Configuration Manager.

MINING_TRACE

This parameter has the following values:

- 0 = Minimum trace level
- 1 = Full trace

0 is the default value.

MINING_TRAINING_INTERVAL

This parameter defines how often the mining training process is executed. The interval is expressed as a number of days.

MINING_TRAINING_TIME

This parameter defines the time stamp when the mining training runs successfully. If the mining training operation is successful, this time stamp will carry the value of `TIME_CUT_OFF_LOCAL` for this extraction run. This parameter is an IBM reserved parameter.

MINING_USER_NAME

This parameter defines the name of the user who runs the mining operations.

NF_EFFECTIVE_ORDER_STATUS_ID

This parameter creates a list of `ORDER_STATUS_IDs` as compared to the `ORDER_STATUS_ID` in the `FACT_ORDERITEMS` table, which represent orderitem entries that are part of orders with realized revenue. Examples are orders that are shipped, completed, or back-ordered.

The following are the default values:

- 5 = Low Inventory
- 6 = Payment Authorization
- 7 = Order Shipped
- 13 = Back-ordered
- 14 = Released For Fulfillment
- 15 = Deposited
- 17 = Ready For Fulfillment
- 18 = Waiting for Remote Fulfillment

NF_EFFECTIVE_MINUTES

This parameter decides the number of minutes that are allowed to pass from the time that the tracked event occurred until a completed order is created. If

an order is created after this effectiveness time, then that order is not associated with that particular event. The default value is **60**

NON_PURGE_ORD_STATUS

This parameter determines how to process records after their deletion from the ORDER and ORDERITEMS tables. If a deleted record has a status value with a NON_PURGE_ORD_STATUS flag, then WCA sets DELETED_STATUS to 1 in the corresponding WCA (FACT_ORDERS or FACT_ORDERITEMS table).

This parameter tells WCA which types of records might be deleted as a result of user action and not as a result of purging the datamart. Records deleted as a result of user action have DELETED_STATUS set to 1. Records deleted as a result of a datamart purge continue to exist in the WCA datamart with the DELETED_STATUS set to 0.

The following are the default values:

- A = Requires Review
- E = CSR Edit
- I = Submitted Order
- M = Payment Initiated
- P = Pending Order
- Q = Order Template
- X = Cancelled
- Y = Private Requisition List
- Z = Shareable Requisition List

ORDER_STATUS_BILLED

This parameter determines the ORDER_STATUS_IDs in the FACT_ORDERS and FACT_ORDERITEMS tables that represent billed orders. You can find the descriptions of the status codes in the WCA.ORDER_STATUS_REF table.

The following are the default values:

- 7 = Order Shipped
- 14 = Released For Fulfillment
- 15 = Deposited
- 18 = Waiting for Remote Fulfillment

ORDER_STATUS_CANCELLED

This parameter determines the ORDER_STATUS_IDs in the FACT_ORDERS and FACT_ORDERITEMS tables that represent cancelled orders. The descriptions of the status codes are in the WCA.ORDER_STATUS_REF table. The default value is **5**. This parameter can have multiple values.

ORDER_STATUS_COLLECTED

This parameter determines the ORDER_STATUS_IDs in the FACT_ORDERS and FACT_ORDERITEMS tables that represent collected orders. You can find the descriptions of the status codes in the WCA.ORDER_STATUS_REF table. The default value is **15**. This parameter can have multiple values.

ORDER_STATUS_ID_NOREV

This parameter helps WCA determine the records in the FACT_ORDERS and FACT_ORDERITEMS tables that represent orders whose currency amount columns might not be populated. In this case, WCA sets the currency amounts to 0 and the currency description columns to NULL. The value must be an integer. You can compare this value to the ORDER_STATUS_ID field. The default value is **19**. This parameter can have multiple values.

ORDER_STATUS_ID_SUM_MEMBER

This parameter determines the set of ORDER_STATUS_ID values that a row in the FACT_ORDERITEMS table must match to populate rows in the SUM_MEMBER table.

The following are the default values:

- 5 = Low Inventory
- 6 = Payment Authorized
- 7 = Order Shipped
- 13 = Back-ordered
- 14 = Released For Fulfillment
- 15 = Deposited
- 17 = Ready For Fulfillment
- 18 = Waiting for Remote Fulfillment

ORDER_STATUS_ID_SUM_TRADING

This parameter determines the set of ORDER_STATUS_ID values that a row in the FACT_ORDERITEMS table must match to populate rows in the SUM_TRADING table.

The following are the default values:

- 5 = Low Inventory
- 6 = Payment Authorized
- 7 = Order Shipped
- 13 = Back-ordered
- 14 = Released For Fulfillment
- 15 = Deposited
- 17 = Ready For Fulfillment
- 18 = Waiting for Remote Fulfillment

ORDER_STATUS_XFERRED

The default values are:

- R = Read For Remote Fulfillment
- G = Waiting For Remote Fulfillment
- F = Ready For Remote Fulfillment

ORDERS_AWAITING_PAYMENT

This parameter helps WCA determine which ORDERS and ORDERITEMS are still awaiting payment. You can compare this parameter with ORDER_STATUS_ID in the FACT_ORDERS and FACT_ORDERITEMS tables. Reports can use this parameter to show metrics that only apply to orders that are awaiting payment. The default value is 4. This parameter can have multiple values.

ORG_BUSINESS_TYPE

This parameter applies to business-to-business transactions only. The default value is **OrgEntityBusinessType**. This parameter determines how WCA populates the WCA.ORGANIZATION.BUSINESS_TYPE column. WCA uses this string against the WebSphere Commerce MBRATTR table to determine the MBRATTR_ID that is used against the WebSphere Commerce MBRATTRVAL table to find the string to put into the WCA.ORGANIZATION.BUSINESS_TYPE column.

ORG_INDUSTRY_TYPE

This parameter applies to business-to-business transactions only. The default value is **OrgEntityIndustryType**. This parameter determines how WCA populates the WCA.ORGANIZATION.INDUSTRY_TYPE column. WCA uses this string against the WebSphere Commerce MBRATTR table to determine the MBRATTR_ID that is used against the WebSphere Commerce MBRATTRVAL

table to find the string to put into the WCA.ORGANIZATION.INDUSTRY_TYPE column.

PRODUCT_PRICE_AGGREGATE

This parameter applies to both business-to-business and business-to-customer transactions. The default is **MIN**, but there are three possible values: **MIN**, **MAX**, and **AVG**. This parameter controls the method that determines the following columns in the WCA database.

Table 8. Transaction Methods

<p>Transaction Type: Business-to-Business</p> <p>Columns: WCA.PRODUCT.RPT_EST_LIST_PRICE WCA.OFFER.EST_MQ_VALUE</p>	<p>Description: Because there can be multiple prices available for a single product in multiple currencies as represented in the WCA.OFFER_PRICE table, it is necessary to use an aggregation function to pick only one estimated price per product. This parameter allows the customer to choose the minimum, maximum, or average price to use in this column based on the reporting currency.</p>
<p>Transaction Type: Business-to-Customer</p> <p>Column: WCA.PRODUCT.RPT_EST_LIST_PRICE</p>	<p>Description: In WebSphere Commerce Version 5.5, the source is the WebSphere Commerce LISTPRICE table.</p> <p>Because there can be multiple list prices for a product, this parameter allows the customer to populate this column with the minimum, maximum, and average price based on the reporting currency to be used in this column. In WebSphere Commerce Professional Edition, Version 5.5, the source is the WebSphere Commerce OFFERPRICE table.</p> <p>Because there can be multiple prices available for a single product in multiple currencies as represented in the WebSphere Commerce OFFERPRICE table, it is necessary to use an aggregation function to pick only one estimated price per product. This parameter allows the customer to choose the minimum, maximum or average price based on the reporting currency to be used in this column.</p>

REPORT_CURRENCY

This parameter defines the currency to convert all of the RPT_XXXX columns. You select the currency by using the WCA configuration tool. All of the stores that WCA reports on must be able to convert all of their supported currencies to this report currency.

REPORT_LANGUAGE

This parameter defines the language that is used by all of the reports. Reporting tools use this parameter to determine how to display language-specific names and descriptions. Primarily, this parameter defines the common language used in XXXX_REF tables. The WCA Configuration program determines the value, and the value depends on languages that are supported by the operating system that is currently installed.

REPLICATION_METHOD

This parameter controls the replication process. If the parameter is set to **Y**, continuous replication is used. If the parameter is set to **N**, scheduled

replication is used. For a discussion of replication methods, see "Replication options" in the *IBM WebSphere Commerce Analyzer Installation and Configuration Guide, Version 5.5*.

REPLICATION_START

This parameter is an IBM reserved parameter. Do not change it.

REPLICATION_SUCCEED

This parameter is an IBM reserved parameter. Do not change it.

REQUEST_ETL_PROCESS

This parameter is an IBM reserved parameter. Do not change it.

RFM_BINS

This parameter applies to business-to-business transactions only. A customer-defined field that designates the number of bins into which to divide the data. The default value is 5. After the records are ranked and divided equally into 5 bins, each record is assigned a corresponding R, F, or M value of 1, 2, 3, 4, or 5 depending upon the bin that they are associated with. A composite value is all of the variations from 111 through 555. The ideal value for this field is 3, 4, or 5.

RFM_INTERVAL

This parameter is a definable field that determines the frequency in days of running Recency, Frequency, and Monetary (RFM). This parameter is an IBM reserved parameter. *Do not* change it.

RFM_LAST_RUN

This parameter indicates the last time that RFM was run. This parameter is an IBM reserved parameter. *Do not* change it.

RFQSP_OUTSTANDING_ORDERS

This parameter determines the ORDER_STATUS_IDs in the FACT_ORDERS and FACT_ORDERITEMS tables that represent the outstanding orders. You can consider orders related to RFQSPs with these flags as outstanding because the customer has not initiated payment. The descriptions of the status codes are in the WCA.ORDER_STATUS_REF table.

The following are the default values:

- 1 = Pending Order
- 3 = Submitted Order
- 4 = Payment Initiated
- 9 = Waiting for Approval
- 11 = Requires Review
- 12 = CSR Edit

RFQ_RESPONSE_IN_PREPARATION

This parameter helps WCA determine which RFQ_RSP records still are in the prepared state. You can compare this multi-valued parameter with the RFQ_RSP.RSP_STATUS_ID field. The default value is 1.

RFQ_WINNING_RESPONSES

This parameter determines which RSP_STATUS_ID represents RFQ responses that are winning responses. The descriptions of the status codes are in the WCA.RSP_STATUS_REF table.

The following are the default values:

- 8 = Won
- 10 = Won Completed

SRF_EFFECTIVE_ORDER_STATUS_ID

This parameter creates a list of ORDER_STATUS_IDs as compared to the

ORDER_STATUS_ID in the FACT_ORDERITEMS table, which represent orderitem entries that are part of orders with realized revenue. Examples are orders that are shipped, completed, or back-ordered.

The following are the default values:

- 5 = Low Inventory
- 6 = Payment Authorized
- 7 = Order Shipped
- 13 = Back-ordered
- 14 = Released For Fulfillment
- 15 = Deposited
- 17 = Ready For Fulfillment
- 18 = Waiting for Remote Fulfillment

SRF_EFFECTIVE_MINUTES

This parameter decides the number of minutes that are allowed to pass from the time that the tracked event occurred until a completed order is created. If an order is created after this effectiveness time, that order is not associated with that particular event. The default value is **60**

STORE

This parameter indicates the STORE_ID of all the stores reported on during this instance of WCA. The configuration tool sets this parameter. There is one row per store.

TIME_CUT_OFF

This parameter defines the end of the extraction window, which was set at the beginning of the extraction and replication process. The WebSphere Commerce server supplies this parameter. This parameter is an IBM reserved parameter. *Do not* change it.

TIME_CUT_OFF_LOCAL

This parameter defines the end of the extract window. It is set at the beginning of the extraction and replication processes in the time stamp relative to the local WCA database. This parameter is an IBM reserved parameter. *Do not* change it.

TIME_CUT_OFF_PREV

This parameter defines the beginning of the extract window. It is set at the beginning of the extraction and replication processes. The WebSphere Commerce server supplies this parameter. This parameter is an IBM reserved parameter. *Do not* change it.

WCS_SOURCE

This parameter defines the version of the WebSphere Commerce database from which the WCA data is extracted. Currently, the parameter carries the value 55be_ext. This parameter is an IBM reserved parameter. *Do not* change it.

The following table is a list of the IBM Tivoli Web Site Analyzer columns:

wsa_category

The default value is catGroupId.

wsa_coupon

The default value is couponIds

Creating or changing parameters

For quick access to parameters, use the WCA Parameter Manager.

The following table contains a list of entries in the Parameter Manager that enable you to access panels used for WCA parameter maintenance:

Table 9. WCA parameters and the panels used for their maintenance

Entry	WCA Parameters
Stores	store, REPORT_CURRENCY, REPORT_LANGUAGE, BROKER_STORE_ID
Catalogs	ANALYSIS_CATALOG_ID
Mining Models	MINING_USER_NAME, MINING_APPLY_TIME, MINING_BASE_NAME, MINING_PASSWORD, MINING_TRACE, MINING_TRAINING_INTERVAL, MINING_TRAINING_TIME, MINING_USER_ID
RFM	RFM_INTERVAL, RFM_BINS, RFM_LAST_RUN, ORDER_STATUS_ID_SUM_MEMBER, ORDER_STATUS_ID_SUM_TRADING, ORDER_STATUS_XFERRED, RFM_LAST_RUN
Abandoned Orders	ABANDONED_MINUTES, ABANDONED_ORD_STATUS
Order Properties	NON_PURGE_ORD_STATUS, ORDERS_AWAITING_PAYMENT, ORDER_STATUS_BILLED, ORDER_STATUS_CANCELLED, ORDER_STATUS_COLLECTED, ORDER_STATUS_ID_NOREV
Member Properties	ADDRESS_MEMBER_TYPE_ID, DMT_PROSPECT_ORD_STATUS, DMT_PURCHASER_ORD_STATUS
Contracts	CONTRACT_ACTIVE, CONTRACT_IN_PREPARATION, CONTRACT_CANCELLED
RFQ Properties	RFQ_RESPONSE_IN_PREPARATION, RFQ_WINNING_RESPONSE, RFQ_OUTSTANDING_ORDER
Financial Periods	
Order Associations	FE_EFFECTIVE_MINUTES, FE_EFFECTIVE_ORD_STATUS, FM_EFFECTIVE_MINUTES, FM_EFFECTIVE_ORD_STATUS, CPN_REDEEMED_ORDER_STATUS
Order Status	
Additional Properties	PRODUCT_PRICE_AGGREGATE, ORG_BUSINESS_TYPE, ORG_INDUSTRY_TYPE, HOTSPOT_LIMIT, WSA_PARM_TYPE

From the Maintain WCA Parameters window, you can create a parameter by typing in the new name of the parameter in the **Parameter Type** field. As with other customer-defined steps, functions, and externally called script file names, prefix your customer-defined parameter names with *UX* to easily identify them as

being customer-defined and not supplied with WCA. This UX prefix naming convention is only a recommendation. The location for wcaparm.properties is %IWDA_DIR%\lib.

You can type the parameter value, or multiple values, in the **Parameter Value** field to add a value without having to define it in the properties file. Or, if you want to permit a value from within a range of values or specify a default value for your own parameter, create a customer-defined wcaparm.properties file. WCA uses the definitions in this properties file to populate the Maintain WCA Parameters window.

Following are examples of some entries for the wcaparm.properties file:

```
parm1.name=UX_MyOwnParm parameter identification (PARAM_TYPE)
parm1.type=s             how is the parameter handled on the panel:
                        1=single value,
                        n=multiple values,
                        s=predefined values
parm1.values=Y;N        values for the drop-down selection box
parm1.defaults=Y        default setting
parm1.prog=generic

parm2.name=UX_MySecondParm
parm2.type=1
parm2.defaults=15
parm2.prog=generic
```

The wcaparm.properties file provided by WCA is located in the lib subdirectory. If you create another wcaparm.properties file for your own parameters, place it in the tmp directory.

Parameter table after configuration

The following is a representation of the WCA.PARAMETERS table as it might look after configuration (without the **last_update** and **last_update_ID** fields).

The REPORT_LANGUAGE and REPORT_CURRENCY parameter type values originate from the configuration tool. The TIME_CUT_OFF, TIME_CUT_OFF_PREV, TIME_CUT_OFF_LOCAL, EXTRACTION_COUNT, and EXTRACTION_SUCCEED parameter type values originate from the extraction process.

The values shown in this representation of the WCA.PARAMETERS table are default values. All of the parameters in this list can have multiple values in the WCA PARAMETER table. Also, use these parameters in .sql clauses only.

ROW_NUM	PARAM_TYPE	PARAM_VALUE
1	ABANDONED_MINUTES	60
1	ABANDONED_ORD_STATUS	A
2	ABANDONED_ORD_STATUS	E
3	ABANDONED_ORD_STATUS	I
4	ABANDONED_ORD_STATUS	P
5	ABANDONED_ORD_STATUS	L
6	ABANDONED_ORD_STATUS	M
7	ABANDONED_ORD_STATUS	W
8	ABANDONED_ORD_STATUS	X

ROW_NUM	PARAM_TYPE	PARAM_VALUE
1	ANALYSIS_CATALOG_ID	20101000000
2	ANALYSIS_CATALOG_ID	21101000000
1	BROKER_STORE_ID	211
1	CONTRACT_ACTIVE	3
1	CONTRACT_CANCELLED	5
1	CONTRACT_IN_PREPARATION	0
1	CPN_REDEEMED_ORDER_STATUS	4
2	CPN_REDEEMED_ORDER_STATUS	6
3	CPN_REDEEMED_ORDER_STATUS	7
1	DMT_PROSPECT_ORD_STATUS	A
2	DMT_PROSPECT_ORD_STATUS	E
3	DMT_PROSPECT_ORD_STATUS	I
4	DMT_PROSPECT_ORD_STATUS	W
5	DMT_PROSPECT_ORD_STATUS	M
6	DMT_PROSPECT_ORD_STATUS	P
7	DMT_PROSPECT_ORD_STATUS	Q
8	DMT_PROSPECT_ORD_STATUS	X
9	DMT_PROSPECT_ORD_STATUS	Y
10	DMT_PROSPECT_ORD_STATUS	Z
1	DMT_PURCHASER_ORD_STATUS	B
2	DMT_PURCHASER_ORD_STATUS	C
3	DMT_PURCHASER_ORD_STATUS	D
4	DMT_PURCHASER_ORD_STATUS	F
5	DMT_PURCHASER_ORD_STATUS	G
6	DMT_PURCHASER_ORD_STATUS	L
7	DMT_PURCHASER_ORD_STATUS	R
8	DMT_PURCHASER_ORD_STATUS	S
1	EXTRACTION_COUNT	2
1	EXTRACTION_SUCCEED	0
1	EXTRACTION_TRACE	0
1	FE_EFFECTIVE_MINUTES	60
1	FE_EFFECTIVE_ORD_STATUS	B
2	FE_EFFECTIVE_ORD_STATUS	C
3	FE_EFFECTIVE_ORD_STATUS	D
4	FE_EFFECTIVE_ORD_STATUS	F
5	FE_EFFECTIVE_ORD_STATUS	G
6	FE_EFFECTIVE_ORD_STATUS	L
7	FE_EFFECTIVE_ORD_STATUS	R
8	FE_EFFECTIVE_ORD_STATUS	S
1	FISCAL_PERIODS_UNTIL	2005
1	FISCAL_YEAR_START	C0101

ROW_NUM	PARAM_TYPE	PARAM_VALUE
1	FM_EFFECTIVE_MINUTES	60
1	FM_EFFECTIVE_ORD_STATUS	B
2	FM_EFFECTIVE_ORD_STATUS	C
3	FM_EFFECTIVE_ORD_STATUS	D
4	FM_EFFECTIVE_ORD_STATUS	F
5	FM_EFFECTIVE_ORD_STATUS	G
6	FM_EFFECTIVE_ORD_STATUS	L
7	FM_EFFECTIVE_ORD_STATUS	R
8	FM_EFFECTIVE_ORD_STATUS	S
1	MEMBER_ADDRESS_TYPE_ID	3
1	MINING_APPLY_INTERVAL	1
1	MINING_APPLY_TIME	1000-01-01-00.00.00
1	MINING_BASE_NAME	mine0215
1	MINING_CLOSED_LOOP	Y
1	MINING_PASSWORD	<i>variable</i>
1	MINING_TRACE	0
1	MINING_TRAINING_INTERVAL	1
1	MINING_TRAINING_TIME	1000-01-01-00.00.00
1	MINING_USER_NAME	martuser
1	NF_EFFECTIVE_MINUTES	60
1	NF_EFFECTIVE_ORD_STATUS_ID	5
2	NF_EFFECTIVE_ORD_STATUS_ID	6
3	NF_EFFECTIVE_ORD_STATUS_ID	7
4	NF_EFFECTIVE_ORD_STATUS_ID	13
5	NF_EFFECTIVE_ORD_STATUS_ID	14
6	NF_EFFECTIVE_ORD_STATUS_ID	15
7	NF_EFFECTIVE_ORD_STATUS_ID	17
8	NF_EFFECTIVE_ORD_STATUS_ID	18
1	NON_PURGE_ORD_STATUS	A
2	NON_PURGE_ORD_STATUS	E
3	NON_PURGE_ORD_STATUS	I
4	NON_PURGE_ORD_STATUS	P
5	NON_PURGE_ORD_STATUS	L
6	NON_PURGE_ORD_STATUS	M
7	NON_PURGE_ORD_STATUS	Q
8	NON_PURGE_ORD_STATUS	X
9	NON_PURGE_ORD_STATUS	Y
10	NON_PURGE_ORD_STATUS	Z
1	ORDERS_AWAITING_PAYMENT	4
1	ORDER_STATUS_BILLED	7
2	ORDER_STATUS_BILLED	14

ROW_NUM	PARAM_TYPE	PARAM_VALUE
3	ORDER_STATUS_BILLED	15
4	ORDER_STATUS_BILLED	18
1	ORDER_STATUS_CANCELLED	2
1	ORDER_STATUS_COLLECTED	15
1	ORDER_STATUS_ID_NOREV	19
1	ORDER_STATUS_ID_SUM_MEMBER	5
2	ORDER_STATUS_ID_SUM_MEMBER	6
3	ORDER_STATUS_ID_SUM_MEMBER	7
4	ORDER_STATUS_ID_SUM_MEMBER	13
5	ORDER_STATUS_ID_SUM_MEMBER	14
6	ORDER_STATUS_ID_SUM_MEMBER	15
7	ORDER_STATUS_ID_SUM_MEMBER	17
8	ORDER_STATUS_ID_SUM_MEMBER	18
1	ORDER_STATUS_ID_SUM_TRADING	5
2	ORDER_STATUS_ID_SUM_TRADING	6
3	ORDER_STATUS_ID_SUM_TRADING	7
4	ORDER_STATUS_ID_SUM_TRADING	13
5	ORDER_STATUS_ID_SUM_TRADING	14
6	ORDER_STATUS_ID_SUM_TRADING	15
7	ORDER_STATUS_ID_SUM_TRADING	17
8	ORDER_STATUS_ID_SUM_TRADING	18
1	ORDER_STATUS_XFERRED	F
2	ORDER_STATUS_XFERRED	G
3	ORDER_STATUS_XFERRED	R
1	ORG_BUSINESS_TYPE	OrgEntityBusinessType
1	ORG_INDUSTRY_TYPE	OrgEntityIndustryType
1	PRODUCT_PRICE_AGGREGATE	MIN
1	REPLICATION_METHOD	N
1	REPLICATION_START	2003-02-17 16:26:19.359002
1	RELOCATION_SUCCEED	0
1	REPORT_CURRENCY	CAD
1	REPORT_LANGUAGE	- 1
1	REQUEST_ETL_PROCESS	0
1	RFM_BINS	5
1	RFM_INTERVAL	1
1	RFM_LAST_RUN	1000-01-01-00.00.00
1	RFQRSP_OUTSTANDING_ORDERS	11
2	RFQRSP_OUTSTANDING_ORDERS	12
3	RFQRSP_OUTSTANDING_ORDERS	3
4	RFQRSP_OUTSTANDING_ORDERS	1

ROW_NUM	PARAM_TYPE	PARAM_VALUE
5	RFQRSP_OUTSTANDING_ORDERS	4
6	RFQRSP_OUTSTANDING_ORDERS	9
1	RFQ_RESPONSE_IN_PREPARATION	1
1	RFQ_WINNING_RESPONSES	8
2	RFQ_WINNING_RESPONSES	10
1	SRF_EFFECTIVE_MINUTES	60
1	SRF_EFFECTIVE_ORD_STATUS_ID	5
2	SRF_EFFECTIVE_ORD_STATUS_ID	6
3	SRF_EFFECTIVE_ORD_STATUS_ID	7
4	SRF_EFFECTIVE_ORD_STATUS_ID	13
5	SRF_EFFECTIVE_ORD_STATUS_ID	14
6	SRF_EFFECTIVE_ORD_STATUS_ID	15
7	SRF_EFFECTIVE_ORD_STATUS_ID	17
8	SRF_EFFECTIVE_ORD_STATUS_ID	18
1	TIME_CUT_OFF	2003-02-17 16:26:19.359002
1	TIME_CUT_OFF_LOCAL	2003-02-17 16.51.21.109002
1	TIME_CUT_OFF_PREV	1000-01-01-00.00.00
1	WCS_SOURCE	55be_ext

Appendix B. Notices

This information was developed for products and services offered in the U.S.A. IBM might not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the information. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this information at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
J46A/G4
555 Bailey Avenue
San Jose, CA 95141-1003
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>).

Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, or other countries, or both:

DB2
DB2 Universal Database
IBM
Intelligent Miner
WebSphere

Microsoft, Windows, Windows NT[®], and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.

Glossary

A

ad (or advertisement). A piece of marketing collateral that is published or broadcast to increase awareness about a product or service. On the Web, the most common type is the banner advertisement.

B

business question. A question that is answered in a business report regarding specific information about the success of different campaigns, initiatives, and the customers who are using the store.

C

campaign. A planned series of operations that are pursued to achieve a defined set of business objectives. In retail marketing, an initiative is a common technique used to achieve campaign objectives.

column. In a relational database management system, the name for an attribute. The collection of column values that form the description of a particular entity is called a row. A column is equivalent to a field in a record in a nonrelational file system.

customer session data. Information gathered from customers during the time they visit the online store.

customization. User additions or changes to WCA to more closely fit an individual e-commerce business model.

D

Database Managed Storage (DMS). Type of data storage in which the table space is managed by the database administrator. The size of the table space is specified and the space allocated when the tables are created.

datamart. A subset of a data warehouse that contains data tailored for the specific needs of a department or team. A datamart can be a subset of a warehouse for your entire organization, such as data contained in OLAP tools.

data mining. The process of collecting critical business information from a data warehouse, correlating it and uncovering associations, patterns, and trends.

E

ETL. Extract, Transform, and Load. The functions performed when pulling data out of one database and placing it into another of a different type.

extension. User additions to WCA components (for example, new reports or new datamart tables).

extraction. Pulling data out of a database. For WCA, the process of moving data from temporary tables on the WCA server to the WCA datamart. The data in the temporary tables was replicated from the WebSphere Commerce database.

extraction time window. The window of time between the last time the WCA Extraction was run for this source and the current time. For WebSphere Commerce 5.5 sources, this is indicated by the WCA parameters `TIME_CUT_OFF` and `TIME_CUT_OFF_PREV`.

F

fulfillment. The process that occurs when an order is received. Fulfillment processes often include tasks such as order management, shipping management, returns, and status tracking.

I

impression. Represents the collaboration of a campaign, an initiative, and an e-marketing spot on a Web page to provide information to customers and allow customers to take advantage of that information by clicking on a link associated with that collaboration.

initiative. An applied technique used to encourage a specific behavior such as purchasing a product.

M

metaphor. A WebSphere Commerce feature provided as part of the Product Advisor component. It provides three usage paradigms (or metaphors) for a shopper to navigate products: Product Explorer, Sales Assistant, and Product Comparison. The Product Explorer metaphor allows the user to set some feature requirements (constraints) for a product (cost, color, type, and so on) and search for matching products. The Sales Assistant metaphor is targeted towards shoppers who are not familiar with the product details and cannot set feature requirements. This metaphor asks a series of questions from which it infers what products the customer wants. The Product Comparison

metaphor allows users to compare two or more products. For further information on metaphors, see the WebSphere Commerce documentation.

mining base. A repository where all the information about the mining run settings and the corresponding results is stored.

O

ODBC name. The Open Database Connectivity name of the database.

ODS. Operation Data Store. The working area for the ETL processing. The data from WebSphere Commerce is replicated to ODS (_r tables).

P

PMML. Predictive Model Markup Language. An XML-based language defined by the Data Mining Group that provides a way for companies to define predictive models and share models between compliant vendors' applications.

Product Comparison. Product Comparison is a metaphor that allows users to compare two or more products.

Product Explorer. Product Explorer is a metaphor that allows the user to set some feature requirements (constraints) for a product (cost, color, type, and so on) and search for matching products.

R

replication. The process of maintaining a defined set of data in more than one location. It involves copying designated changes for one location to another, and synchronizing the data in both locations. For WCA, the process of moving data from the WebSphere Commerce database into temporary tables on the WCA server.

reporting application. A program that gathers information about the customers and sales transactions of a business.

revision. User changes to existing WCA components (for example, reports or datamart tables).

RFM. Recency, frequency, monetary. A technique used to determine which customers are the best ones by examining how recently a customer has purchased (recency), how often they purchase (frequency), and how much the customer spends (monetary).

RFQ. Request for quotation. An invitation to suppliers to bid on supplying described products or services needed by a company or public agency.

S

Sales Assistant. Sales Assistant is a metaphor that is targeted towards shoppers who are not familiar with the product details and cannot set feature requirements.

System Managed Storage (SMS). The type of data storage in which the operating system manages the tables space, which is limited by the size of the hard disk. Data is stored randomly on the hard disk under the table space's directory container (the directory name in the file system).

T

table. A named data object consisting of a specific number of columns and some unordered rows.

U

User Registration properties file. A file on the WebSphere Commerce server that contains information needed to support the correct language and country for a store.

V

view. An alternative representation of data from one or more tables. A view can include all or some of the columns in the table or tables on which it is defined.

W

Warehouse Center Control Database. The Warehouse Center database that contains the control tables that are required to store Warehouse Center metadata.

WebSphere Application Server. A comprehensive Java 2 Platform, Enterprise Edition (J2EE) 1.3 and Web services technology-based application server that integrates enterprise data and transactions with the e-business world. Through a rich application deployment environment, you can build, manage and deploy dynamic e-business applications, handle high-transaction volumes and extend back-end business data and applications to the Web.

Index

Special characters

.bat (batch) files 58, 60
.sh (shell) scripts 58
%IWDA_DIR%, definition v

A

ABANDONED_MINUTES column 77
ABANDONED_ORD_STATUS column 77
adding
 columns 56
 data 56
 tables 55
ANALYSIS_CATALOG_ID column 78
app_type column name 74
Apply 63
apply result, registering 73
APPTABS control table 73, 75

B

batch files 58, 60
books, WebSphere Commerce vi
BROKER_STORE_ID column 78
buffer pools, changing for DMS 23
business
 questions 7
 reports 7
business analyst, description 3
business manager, description 4
business questions
 business-to-business 7
 campaign management 8
 categories 7
 orders 8
 Product Advisor 8
 sales 8
 shoppers 8
 Web site traffic 8
business reports
 accessing 3
business-to-business transactions 7

C

CALENDAR_DATE column 21
CALENDAR_DATE field 20
campaign management 8
Capture 63
CD tables, determining size 65
change data tables, determining size 65
changing the buffer pools, tables, and table spaces for customized DMS 23
changing the default reporting currency 15
characteristics, shoppers 8
clustering 72
coding standards 58, 64

columns 85

ABANDONED_MINUTES 77
ABANDONED_ORD_STATUS 77
ANALYSIS_CATALOG_ID 78
BROKER_STORE_ID 78
CALENDAR_DATE 21
CONTRACT_ACTIVE 78
CONTRACT_CANCELLED 78
CONTRACT_IN_PREPARATION 78
CPN_REDEEMED_ORDER_STATUS 78
DAY_OF_FM 21
DAY_OF_FM_ID 21
DAY_OF_FM_REF 22
DAY_OF_FY 21
DAY_OF_FY_REF 22
DAY_OF_WK 21
DAY_OF_WK_ID 21
DAY_OF_WK_REF 22
DMT_PROSPECT_ORD_STATUS 78
DMT_PURCHASER_ORD_STATUS 79
EXTRACTION_COUNT 79
EXTRACTION_SUCCEED 79
FE_EFFECTIVE_MINUTES 79
FE_EFFECTIVE_ORD_STATUS 79
FISCAL_YR 22
FM_EFFECTIVE_MINUTES 80
FM_EFFECTIVE_ORD_STATUS 80
HOLIDAY_FLG 22
MEMBER_ADDRESS_TYPE_ID 80
MINING_APPLY_INTERVAL 81
MINING_APPLY_TIME 81
MINING_BASE_NAME 81
MINING_PASSWORD 81
MINING_TRACE 81
MINING_TRAINING_INTERVAL 81
MINING_TRAINING_TIME 81
MINING_USER_NAME 81
MON_OF_FY 21
MON_OF_FY_ID 22
MON_OF_FY_REF 22
NF_EFFECTIVE_MINUTES 81
NON_PURGE_ORD_STATUS 82
ORDER_STATUS_BILLED 82
ORDER_STATUS_CANCELLED 82
ORDER_STATUS_COLLECTED 82
ORDER_STATUS_ID_NOREV 82
ORDER_STATUS_ID_SUM_MEMBER 83
ORDER_STATUS_ID_SUM_TRADING 83
ORDER_STATUS_XFERRED 83
ORDERS_AWAITING_PAYMENT 83
ORG_BUSINESS_TYPE 83
ORG_INDUSTRY_TYPE 83
PARAM_TYPE 77
PARAM_VALUE 77
PER_AGGR_ID 21
PER_DESC_ID 21
PER_ID 21
PRODUCT_PRICE_AGGREGATE 84
QTR_OF_FY 22
QTR_OF_FY_ID 22
QTR_OF_FY_REF 22

columns (*continued*)

REPLICATION_METHOD 84
REPLICATION_START 85
REPORT_CURRENCY 84
REPORT_LANGUAGE 84
REQUEST_ETL_PROCESS 85
RFM_BINS 85
RFM_INTERVAL 85
RFM_LAST_RUN 85
RFQ_WINNING_RESPONSES 85
ROW_NUM 77
SRF_EFFECTIVE_MINUTES 86
STORE 86
TIME_CUT_OFF 86
TIME_CUT_OFF_LOCAL 86
TIME_CUT_OFF_PREV 86
WCS_SOURCE 86
WEEKDAY_FLG 22
WK_OF_FM 21
WK_OF_FM_ID 21
WK_OF_FM_REF 22
WK_OF_FQ 21
WK_OF_FQ_ID 21
WK_OF_FQ_REF 22
WK_OF_FY 21
WK_OF_FY_ID 21
commands
 REORG 27
 REORGCHK 27
 RUNSTATS 27
CommitRows, definition of 39
CommitStatement, definition of 38
completed parameter value 77
COMPLETED_STATUS parameter valued 77
configuration tool
 parameters after configuration 88
CONTRACT_ACTIVE column 78
CONTRACT_CANCELLED column 78
CONTRACT_IN_PREPARATION column 78
control tables
 APPTABS 75
 DATATABS 74
 MODELS 75
 CPN_REDEEMED_ORDER_STATUS 83
 column 78
creating
 data mining models 73
CURCONVERT, definition of 14
CURLIST, definition of 14
cursor mode, running under 43
CursorInsert 43
CursorUpdate 44
customization
 extension 50
 revision 50
customization scenarios 50
customizing
 rules and guidelines 55
customizing extraction 57

D

- data block size, updating 27
- data mining
 - creating new models 73
 - customizing 71
 - features 71
 - introducing 71
 - new model, example 72
 - new models, registering 74
- data tables, registering 72
- data_desc column name 74
- data_id column name 74, 75
- data_location column name 74
- data_name column name 74
- data_type column name 74
- database removal 14
- databases
 - backing up 13
- datamart 16
 - after extraction 77
 - business questions 7
 - customization conventions for tables 55
 - data mining 71
 - description 3
 - removing 14
 - reorganizing tables 27
- datamart (WCA) 72
- datamart customization tasks 55
- DATATABS control table 72, 74
- DAY_OF_FM column 21
- DAY_OF_FM_ID column 21
- DAY_OF_FM_REF column 22
- DAY_OF_FY column 21
- DAY_OF_FY_REF column 22
- DAY_OF_WK column 21
- DAY_OF_WK_ID column 21
- DAY_OF_WK_REF column 22
- DB2
 - Control Center 14
 - Intelligent Miner for Data 71, 72
 - Warehouse Center scripts 58, 60
- DB2 Universal Database
 - support Web site vi
- deletion of records 24
- determining language and currency properties for a store 14
- direct mappings 16
- DMT_PROSPECT_ORD_STATUS column 78
- DMT_PURCHASER_ORD_STATUS column 79
- documentation, WCA vi
- dropping versus removing 14

E

- error codes 58
- error handling 45
- ETL
 - Extract, Transform, Move, and Load process 58
 - flows 31, 33
 - modifying processes 50
 - processes, description of 29

- ETL Driver
 - error handling 45
 - supported command tags 38
 - supported global commands 38
 - supported local commands 39
 - using the 37
- examples
 - database model for clustering 72
- export/import/load mode, running under 44
- extension 50
- extension and revision, definition of 49
- Extract, Transform, Move, and Load
 - See ETML
- extraction
 - definition 5
- extraction process
 - coding standards 58
 - parameters 77
 - parameters after configuration 88
 - troubleshooting 16
- EXTRACTION_COUNT column 79
- EXTRACTION_SUCCEED column 79
- extraction, customizing 57

F

- FE_EFFECTIVE_MINUTES column 79
- FE_EFFECTIVE_ORD_STATUS column 79
- file_location column name 75
- files
 - Reference_Table.properties 20
 - reference.properties 22
 - wca_default_period.csv 20
 - wca_fill_period.sql 20
- fiscal year modifications, making 22
- FISCAL_YR column 22
- flags
 - holiday 22
 - weekday 22
- FM_EFFECTIVE_MINUTES column 80
- FM_EFFECTIVE_ORD_STATUS column 80

G

- guidelines for compatible DB2 script files 42
- guidelines for customizing 55

H

- highlighting conventions v
- HOLIDAY_FLG column 22

I

- improving performance 27
- increasing the log space 65
- index type
 - primary key 75
- insert sql statements 42
- Intelligent Miner for Data
 - data mining functions 3

- Intelligent Miner for Data (*continued*)
 - support Web site. vi
 - working with WCA 5

L

- language and currency properties for a store, determining 14
- LAST_UPDATED_REF table 56
- LoadInsert 44
- LoadInsert, definition of 42
- LoadOption, definition of 40
- LoadParam, definition of 41
- LoadUpdate 44
- log space, increasing 65

M

- making fiscal year modifications 22
- marketing campaigns 8
- MAX_SYNCH_MINUTES, updating 27
- MEMBER_ADDRESS_TYPE_ID column 80
- metaphors 8
- MINING_APPLY_INTERVAL column 81
- MINING_APPLY_TIME column 81
- MINING_BASE_NAME column 81
- MINING_PASSWORD column 81
- MINING_TRACE column 81
- MINING_TRAINING_INTERVAL column 81
- MINING_TRAINING_TIME column 81
- MINING_USER_NAME column 81
- MNF_EFFECTIVE_ORDER_STATUS_ID column 81
- model_alg column name 75
- model_desc column name 75
- model_id column name 75
- model_name column name 75
- model_type column name 75
- model, registering 73
- MODELS control table 73, 75
- modes for combination sql statements 43
- modifying replication control tables 68
- MON_OF_FY column 21
- MON_OF_FY_ID column 22
- MON_OF_FY_REF column 22

N

- NON_PURGE_ORD_STATUS column 81, 82
- notebooks, settings 73

O

- ORDER_STATUS_BILLED column 82
- ORDER_STATUS_CANCELLED column 82
- ORDER_STATUS_COLLECTED column 82
- ORDER_STATUS_ID_NOREV column 82

ORDER_STATUS_ID_SUM_MEMBER
column 83
ORDER_STATUS_ID_SUM_TRADING
column 83
ORDER_STATUS_XFERRED column 83
orders 8
ORDERS_AWAITING_PAYMENT
column 83
ORG_BUSINESS_TYPE column 83
ORG_INDUSTRY_TYPE column 83

P

PARAM_TYPE column 77
PARAM_VALUE column 77
Parameter Manager
definition of 25
using the TraceLog Viewer tool 25
PARAMETER table 78
ABANDONED_MINUTES
column 77
ABANDONED_ORD_STATUS
column 77
ANALYSIS_CATALOG_ID
column 78
BROKER_STORE_ID column 78
CONTRACT_ACTIVE column 78
CONTRACT_CANCELLED
column 78
CONTRACT_IN_PREPARATION
column 78
DMT_PROSPECT_ORD_STATUS
column 78
DMT_PURCHASER_ORD_STATUS
column 79
EXTRACTION_COUNT column 79
EXTRACTION_SUCCEED
column 79
FE_EFFECTIVE_MINUTES
column 79
FE_EFFECTIVE_ORD_STATUS
column 79
FM_EFFECTIVE_MINUTES
column 80
FM_EFFECTIVE_ORD_STATUS
column 80
MEMBER_ADDRESS_TYPE_ID
column 80
MINING_APPLY_INTERVAL
column 81
MINING_APPLY_TIME column 81
MINING_BASE_NAME column 81
MINING_PASSWORD column 81
MINING_TRACE column 81
MINING_TRAINING_INTERVAL
column 81
MINING_TRAINING_TIME
column 81
MINING_USER_NAME column 81
NF_EFFECTIVE_MINUTES
column 81
NF_EFFECTIVE_ORDER_STATUS_ID
column 81
NON_PURGE_ORD_STATUS
column 82
ORDER_STATUS_BILLED column 82

PARAMETER table (*continued*)
ORDER_STATUS_CANCELLED
column 82
ORDER_STATUS_COLLECTED
column 82
ORDER_STATUS_ID_NOREV
column 82
ORDER_STATUS_ID_SUM_MEMBER
column 83
ORDER_STATUS_ID_SUM_TRADING
column 83
ORDER_STATUS_XFERRED
column 83
ORDERS_AWAITING_PAYMENT
column 83
ORG_BUSINESS_TYPE column 83
ORG_INDUSTRY_TYPE column 83
PARAM_TYPE column 77
PARAM_VALUE column 77
PRODUCT_PRICE_AGGREGATE
column 84
REPLICATION_METHOD
column 84
REPLICATION_START column 85
REPORT_CURRENCY column 84
REPORT_LANGUAGE column 84
REQUEST_ETL_PROCESS
column 85
RFM_BINS column 85
RFM_INTERVAL column 85
RFM_LAST_RUN column 85
RFQ_RESPONSE_IN_PREPARATION
column 85
RFQ_WINNING_RESPONSES
column 85
RFQ_RSP_OUTSTANDING_ORDERS
column 85
ROW_NUM column 77
SRF_EFFECTIVE_MINUTES
column 86
SRF_EFFECTIVE_ORDER_STATUS_IDS
column 85
STORE column 86
TIME_CUT_OFF column 86
TIME_CUT_OFF_LOCAL column 86
TIME_CUT_OFF_PREV column 86
WCS_SOURCE column 86
parameter typed 77
parameter value 77
PER_AGGR_ID column 21
PER_DESC_ID column 21
PER_ID column 21
per_id column name 75
PER_ID field 20
performance, improving 27
Predictive Model Markup Language
(PMML) 75
prefix, UX 58
primary key 76
Primary Keys, definition of 41
Product Advisor 8
PRODUCT_PRICE_AGGREGATE
column 84

Q

QTR_OF_FY column 22
QTR_OF_FY_ID column 22
QTR_OF_FY_REF column 22
questions, business 7

R

Recency, Frequency, and Monetary
See RFM
records, restrictions on deleting 24
reference tables 16
Reference_Table.properties file 20
reference.properties file 22
registering
apply result in the APPTABS
table 73
data tables in DATATABS table 72
model in the MODELS table 73
new data mining models 74
registering the replication process 69
removing
datamart 14
store fact and dimension tables 15
stores 15
removing versus dropping 14
REORG command 27
REORGCHK command 27
rep.sql file 65
replication
definition 5
replication and extraction 33
replication control tables 68
replication process
Capture and Apply 63
coding standards 64
customization tasks 65
registering 69
staging tables 64
replication source tables 66
replication table, creating a new 65
REPLICATION_METHOD column 84
REPLICATION_START column 85
REPORT_CURRENCY column 84
REPORT_LANGUAGE column 84
reporting application
description 5
Reporting Framework 5
reports, business 7
request for quote 7
REQUEST_ETL_PROCESS column 85
res_file_name column name 75
revision 50
RFM
conditions to determine value 76
definition 71
RFM_BINS column 85
RFM_INTERVAL column 85
RFM_LAST_RUN column 85
RFQ_RESPONSE_IN_PREPARATION
column 85
RFQ_WINNING_RESPONSES
column 85
RFQ_RSP_OUTSTANDING_ORDERS
column 85
ROW_NUM column 77

- rules for customizing 55
- running the replication steps 69
- running under cursor mode 43
- running under export/import/load mode 44
- RUNSTATS command 27
- Runstats, definition of 38

S

- sales evaluation 7, 8
- sample_pct column name 74
- scheduling replication and extraction 70
- schema (WCA) 72
- script files 58, 60
- SelectPrimaryKeys, definition of 41
- settings notebooks 73
- settings objects 73
- shell scripts 58
- shipped parameter value 77
- shoppers 8
- SourceKeys, definition of 40
- SourceTable, definition of 39
- SRF_EFFECTIVE_MINUTES column 86
- SRF_EFFECTIVE_ORDER_STATUS_ID column 85
- staging tables 64
- starting the Capture process 69
- status column name 76
- STORE column 86
- STORE, definition of 14
- store, removing 15
- STOREENT, definition of 14
- STORELANG, definition of 14
- support Web sites vi
- system administrator, description 3

T

- table space, calculating 65
- table spaces, changing for DMS 23
- table_id column name 75
- table_name column name 75
- table_type column name 76
- tables
 - deleting records from 24
 - LAST_UPDATED_REF 56
 - reorganizing 27
 - WCA.PARAMETERS 77, 88
- tables, changing for DMS 23
- TargetKeys, definition of 40
- TargetTable, definition of 40
- TIME_CUT_OFF column 86
- TIME_CUT_OFF_LOCAL column 86
- TIME_CUT_OFF_PREV column 86
- tools
 - IBM DB2 Intelligent Miner for Data 71
- traffic, Web site 8
- trends 8

U

- unique IDs 56
- update sql statements 43
- UpdateCondition, definition of 42

- UpdateSource, definition of 39
- UpdateType, definition of 41
- usage_type column name 74
- user types, WCA 3
- user-extended tables 55
- using the ETL Driver 37
- UX prefix 58
- UX user extended column 56
- UX user extended table 55

W

- WCA
 - backing up databases 13
 - business reports 3
 - datamart 3, 72
 - documentation vi
 - overview 3
 - reorganizing tables 27
 - schema 72
 - software provided with 3
 - user types 3
 - working with WebSphere Commerce 4
- WCA .PARAMETER table 88
- WCA columns
 - See columns
- WCA ETL sources 33
- WCA server.
 - improving performance of 27
- WCA tables
 - See tables
- wca_default_period.csv file 20
- wca_fill_period.sql file 20
- WCA.PARAMETERS table
 - column names 77
- WCS_SOURCE column 86
- Web site traffic 8
- Web sites, support vi
- WebSphere Commerce
 - books vi
 - online store 3
 - transactional database server 4
 - WebSphere Commerce Accelerator 3
 - working with WCA 4
- WebSphere Commerce Accelerator 3
- WEEKDAY_FLG column 22
- WK_OF_FM column 21
- WK_OF_FM_ID column 21
- WK_OF_FM_REF column 22
- WK_OF_FQ column 21
- WK_OF_FQ_ID column 21
- WK_OF_FQ_REF column 22
- WK_OF_FY column 21
- WK_OF_FY_ID column 21



Printed in U.S.A.