IBM WebSphere Commerce

Technical Operations: Online Help Files

Version 54

IBM WebSphere Commerce

Technical Operations: Online Help Files

Version 54

Note!

Before using this information and the product it supports, be sure to read the general information in the Notices section.

Contents

Chapter 1. Technical Operations roles

Technical operations roles

The following technical operations roles are supported by WebSphere Commerce:

- Site Administrator (page 1)
- Store Administrator

Site Administrator

The Site Administrator installs, configures, and maintains WebSphere Commerce and the associated software and hardware. The Administrator responds to system warnings, alerts, and errors, and diagnoses and resolves system problems. This role typically controls access and authorization (creating and assigning members to the appropriate role), manages the Web site, monitors performance, and manages load balancing tasks. The Site Administrator may also be responsible for establishing and maintaining several server configurations for different stages of development such as testing, staging, and production. This role also handles critical system backups and resolves performance problems.

- Installs, configures, and maintains WebSphere Commerce
- Manages users, organizations, roles, and member groups (Organizational Administrators are primarily responsible for these tasks. Site Administrators set up the minimum IDs to get the site operational.)
- Manages access control
- Defines transports and message types for the site
- Monitors performance of the site and resolves performance problems
- Handles critical system backups
- Specifies Payment Manager settings
- Configures logging and tracing
- Enables and disables WebSphere Commerce components
- Schedules jobs to be run for the site
- Updates registry components
- May access all Commerce Accelerator functions

Store Administrator

The Store Administrator manages the store assets and updates and publishes changes to taxes, shipping and store information. The Store Administrator, usually the lead on the store development team, is the only role on the team with the authority to publish a store archive (the Site Administrator can also publish a store archive). The Store Administrator is usually web-literate and has a thorough knowledge of the store's business procedures.

- Configures messages
- Publishes the store archive
- Manages store updates and assets
- Administers Blaze rules (Does not apply to the Start edition of WebSphere Commerce)
- Manages payments

- Publishes changes to tax, shipping, and store information
- Schedules jobs to be run for a store

Chapter 2. Site Administrator

Administration Console

The Administration Console allows you to control your site or store by completing administrative operations and configuration tasks. If you are authorized to work with multiple stores and languages, you select the store and language with which you want to work when you log on to the Administration Console. The tasks that you are authorized to perform display on the Administration Console home page through various menus. These tasks are based on the user group names (roles) and authority levels.

Note: User group names are not translated in the Administration Console. All names will be displayed in English only.

To return to the Administration Console home page, click the **Home** link at the top right of the screen. To change the store or language you are working with at any time, click the

icon, found in the upper left corner.

Configuring messaging for a site

Activating a transport method for a site

To activate an existing transport method for your site, do the following:

1. Open the Administration Console and log on as a Site Administrator.
2. From the **Configuration** menu, select **Transports**. The Transport Configuration page displays.
3. Click the check box next to the transport you want to activate.
4. Click **Change Status**. The page reloads, indicating that the status of the transport is now Active.

Deactivating a transport method for a site

To deactivate an existing transport method for your site, do the following:

1. Open the Administration Console and log on as a Site Administrator.
2. From the **Configuration** menu, select **Transports**. The Transport Configuration page displays.
3. Click the check box next to the transport you want to deactivate.
4. Click **Change Status**. The page reloads, and the status changes.

Configuring a transport method for a site

To configure a transport method for your site, do the following:

1. Open the Administration Console and log on as a Site Administrator.
2. From the **Configuration** menu, select **Transports**. The Transport Configuration page displays.
3. Select the check box beside the method you want to configure.

4. Click **Configure**. The Transport Configuration Parameters dialog opens. The name of the transport method you selected appears at the top-left of the parameter table.
5. Type the values to be used by the transport method.
6. Click **OK** to accept the changes, or click **Cancel** to return to the Transport Configuration page.

Assigning a transport method to a message type for a site

To assign a transport method to be used for specific message types for a site, do the following:

1. Open the Administration Console and log on as a Site Administrator.
2. From the **Configuration** menu, select **Message Types**. The Message Type Configuration page displays.
3. Click the check box next to the message type to which you want to assign a transport, and click **Change**. If the message type is not in the list, click **New**. The Message Transport Assignment page opens.
4. If this is a new transport assignment, select the message type to which a transport is to be assigned from the **Message Type** drop-down list.
5. Type the transport configuration values in the appropriate fields. In general, a **message severity** of 0,0, and a Standard Device Format is recommended.
6. Click **Next** to configure the transport parameters for the specified message type.
7. Type the attributes for the transport you have chosen for this message type.
8. Click **Finish** or **Cancel** to return to the Message Type Configuration page.

Activating notification

Activating notification

Activating notification

Once you have set the system default settings for the e-mail transport method, activate notification by doing the following:

1. Edit the <*instance_name*>.xml by setting:

```
<Notification display="false">
<order enabled="true" />
<error enabled="true" />
<pwdreset enabled="true" />
</Notification>
```

2. Restart WebSphere Commerce Server.

Note: If you encounter problems when sending e-mail for error notification the following actions may be required:

- Check that configuration values are correct.
- Clean-up all configuration data by deleting all entries that configure transport at the store level. The following SQL statement will delete all entries:

```
delete from cseditatt where store_id is not null
```

- Open the Administration Console and check that all message types are configured properly and do not configure transport at the store level.

Enabling broadcast messages

To send a broadcast e-mail message, do the following:

1. Create a JSP file called `BroadcastMessage.jsp`.
2. Place the file in the site or store directory.
3. Open the Administration Console and log on as a Site or Store Administrator.
   a. From the **Configuration** menu, select **Message Types**. The Message Type Configuration page displays.
   b. Click **New**. The Message Transport Assignment page opens.
      1) On the **Message Type** drop-down list select A broadcast message.
      2) On the **Message Severity** field specify 0 0.
      3) From the **Transport** drop-down list, select E-mail.
      4) From the **Device Format** drop-down list, select Standard Device Format.
   c. Click **Next**. The Message Transport Assignment parameter page displays.
      1) On the **Host** field, type the fully qualified name of your mail server.
      2) On the **Protocol** field, type `smtp`.
      3) On the **Recipient** field, specify a default recipient. If multiple recipients are specified, separate recipient names with commas. The recipient name will be replaced by the customers e-mail address at run-time.
      4) On the **Sender** field, specify the sender of the message. This text appears as the content of the From line in the e-mail.
      5) On the **Subject** field, specify the subject of the message. This text appears as the content of the Subject line in the e-mail.
   d. Click **Finish**.
   e. To send the message, follow the directions in the BroadcastMessage command reference file.
      For example, to send a message to all customers who have purchased part number “sku1234” from any store in site type the following on the address line of your browser:

      ```
      BroadcastMessage?subject=testing&messageContent=this+is+a+test

      &sender=example%40ca.ibm.com&mode=2&partNumber=sku1234&URL=BroadcastMessage.jsp
      ```

Enabling messages to be sent from the Administration Console

If you have based your store on the InFashion sample store, you can enable Customer Service Representatives to send messages to customers using the WebSphere Commerce Administration Console. To enable messages to be sent from the WebSphere Commerce Administration Console, do the following:

1. Open the Administration Console and log on as a Site Administrator or Store Administrator.
2. From the **Configuration** menu, select **Message Types**. The Message Type Configuration page displays.
3. Click **New**. The Message Transport Assignment page opens.
   a. On the **Message Type** drop-down list select Order related message sent by customer service representative.
   b. On the **Message Severity** field specify 0 0.
   c. From the **Transport** drop-down list, select E-mail.
   d. From the **Device Format** drop-down list, select Standard Device Format.

4. Click **Next**. The Message Transport Assignment parameter page displays.
   a. On the **Host** field, type the fully qualified name of your mail server.
   b. On the **Protocol** field, type `smtp`.
   c. On the **Recipient** field, specify a default recipient. If multiple recipients are specified, separate recipient names with commas. The recipient name will be replaced by the customer e-mail address at run-time.
   d. On the **Sender** field, specify the sender of the message.
   This text appears in the Sender field of the e-mail message. (This value is overriden by the E-mail address value entered in Store Services.)
   e. On the **Subject** field, specify the subject of the message.
   This text appears in the Subject field of the e-mail message. (This value is overriden by the values entered in Store Services.)
5. Click **Finish**.

Enabling password reset messages

To enable password reset messages, do the following:

1. If desired, modify the default text in the following password reset messages:
   - `PasswordNotify.jsp` for stores based on the InFashion sample
   - `PasswordResetNotification.jsp` for the default template
2. Activate notification.
3. Open the Administration Console.
   - To configure password reset messages on a site level, log on as a Site Administrator.
   - To configure password reset messages on a store level, log on as a Store Administrator.
4. From the **Configuration** menu, select **Message Types**. The Message Type Configuration page displays.
5. Click **New**. The Message Transport Assignment page displays.
   a. On the **Message Type** drop-down list select Notification message for password reset.
   b. On the **Message Severity** field specify 0 0.
   c. From the **Transport** drop-down list, select E-mail.
   d. From the **Device Format** drop-down list, select Standard Device Format.
6. Click **Next**. The Message Transport Assignment parameter page displays.
   a. On the **Host** field, type the fully qualified name of your mail server.
   b. On the **Protocol** field, type `smtp`.
   c. On the **Recipient** field, specify a default recipient. If multiple recipients are specified, separate recipient names with commas. The recipient name will be replaced by the customer e-mail address at run-time.
   d. On the **Sender** field, specify the sender of the message.
   This text appears in the Sender field of the e-mail message. This value is overriden by the values entered in Store Services.
   e. On the **Subject** field, specify the subject of the message.
   This text appears in the Subject field of the e-mail message. This value is overriden by the values entered in Store Services.
7. Click **Finish**.

Enabling error notification

To enable e-mail error notification, do the following:

1. Ensure that the system default settings for the e-mail transport method have been set.
2. Activate notification for error messages.
3. Assign the error condition message type to a transport.
4. Assign a transport method to message types for a site.

   On the Message Transport Assignment page enter the following values:

   a. On the **Message Type** drop-down list select Description of an Error Condition occurring in WebSphere Commerce.
   b. On the **Message Severity** field specify 0 0.
   c. From the **Transport** drop-down list, select E-mail.
   d. From the **Device Format** drop-down list, select Standard Device Format.

   On the Message Transport Assignment parameter page use the following values:

   a. On the **Host** field, type the fully qualified name of your mail server.
   b. On the **Protocol** field, type `smtp`.
   c. On the **Recipient** field, specify the administrator who should receive error notification messages. If multiple recipients are specified, separate recipient names with commas.
   d. On the **Sender** field, specify the sender of the message. This text appears in the Sender field of the e-mail message.
   e. On the **Subject** field, specify the subject of the message. This text appears in the Subject field of the e-mail message.

Note: If you encounter problems when sending e-mail for error notification the following actions may be required:

- Check that configuration values are correct.
- Clean-up all configuration data by deleting all entries that configure transport at the store level. The following SQL statement will delete all entries:

```
delete from cseditatt where store_id is not null
```

- Open the Administration Console and check that all message types are configured properly and do not configure transport at the store level.

Checking the system settings for the e-mail transport method

Ensure that the default settings for the e-mail transport method have been set by doing the following:

1. Launch the Configuration Manager.
2. Select the **Instance**, then open the **transports** —> **Outbound**—> **JavaMail** —> **ConnectionSpec** folder.
3. Click the **Advanced** tab in the right-hand frame.
4. Set the value of **host**, to your SMTP mail server.
5. Set the value of **protocol**, to `smtp`.
6. Click **Apply**.
7. Restart the WebSphere Commerce Server.

Note: This provides a system wide default setting for the e-mail transport. This setting will be overridden by those created in the Administration Console.

Chapter 3. Store Administrator

Adding a transport method to a store

To add a new transport method to your store, do the following:

1. Open the Administration Console and log on as a Store Administrator.
2. From the **Configuration** menu, select **Transports**. The Transport Configuration page displays.
3. Click **Add**. The Add Transport dialog opens.
4. Select the check box next to the transport you want to add to the store. You can select all transports by selecting the check box at the top-left. If there are no transports available, then you have already added all of the transports made available by the Site Administrator.
5. Click **Add** to add the transport, or click **Cancel** to return to the Transport Configuration page.

Activating a transport method for a store

To activate an existing transport method for your store, do the following:

1. Open the Administration Console and log on as a Store Administrator.
2. From the **Configuration** menu, select **Transports**. The Transport Configuration page displays.
3. Click the check box next to the transport you want to activate.
4. Click **Change Status**. The page reloads and the status changes.

Deactivating a transport method for a store

To deactivate an existing transport method for your store, do the following:

1. Open the Administration Console and log on as a Store Administrator.
2. From the **Configuration** menu, select **Transports**. The Transport Configuration page displays.
3. Click the check box for the transport you want to deactivate.
4. Click **Change Status**. The page reloads, indicating that the transport status is now Inactive.

Configuring a transport method for a store

To configure a transport method for your store, do the following:

1. Open the Administration Console and log on as a Store Administrator.
2. From the **Configuration** menu, select **Transports**. The Transport Configuration page displays.
3. Select the check box beside the transport you want to configure.
4. Click **Configure**. The Transport Configuration Parameters page opens. The name of the transport you selected displays at the top-left of the parameter table.
5. Provide the information for the transport in the appropriate fields.
6. Click **OK** to accept the changes, or click **Cancel** to return to the Transport Configuration menu without making changes.

Note: Do not click **OK** unless you want the settings to take effect. Once you have clicked **OK**, changes made to the configuration of this transport by the Site Administrator will no longer affect this store. If you have not made any changes, or do not want the settings to take effect , click **Cancel**.

Assigning a transport method to a message type for a store

A Store Administrator can accept the transport method assignments made by the Site Administrator, or override them for their store. To assign transport methods to be used for specific message types for a store, do the following:

1. Open the Administration Console and log on as a Store Administrator.
2. From the **Configuration** menu, select **Message Types**. The Message Type Configuration page displays.
3. Click the check box next to the message type to which you want to assign a transport, and click **Change**. If the message type is not in the list, click **New**. The Message Transport Assignment page opens.
4. If this is a new transport assignment, select the message type to which a transport is to be assigned from the **Message Type** drop-down list.
5. Type the transport configuration values in the appropriate fields. In general, a **message severity** of 0,0, and a Standard Device Format is recommended.
6. Click **Next** to configure the transport parameters for the specified message type.
7. Type the attributes for the transport you have chosen for this message type.
8. Click **Finish** to save your changes or **Cancel** to return to the Message Type Configuration page.

Note: Do not click **Finish** unless you want the settings to take effect. Once you have clicked **Finish**, changes made to the configuration of this transport by the Site Administrator will no longer affect this store. If you have not made any changes, or do not want the settings to take effect effect, click **Cancel**.

Chapter 4. Setting up outbound message composition

Setting up outbound message composition

To set up and use the composition service for an outbound message, do the following:

1. Assign the transports to the appropriate message type, using either of these methods:
   - Assigning a transport method to a message type for a site
   - Assigning a transport method to a message type for a store
     A valid device format, as specified in the DEVICEFMT table, must be specified for each transport to be used.
2. Referring to the information in the topic Outbound messaging system interface, create a messaging system object using the SendMsgCmd task command. Use the setMsgType() and setStoreId() initialization services.
3. Invoke the messaging system compose method.

Chapter 5. Using the database cleanup utility

Database Cleanup utility

The Database Cleanup utility allows you to delete many objects from the database at the same time. Go to cleaning the database to find out which objects you can delete. To learn more about each type of object, refer to the object types.

When the Database Cleanup utility deletes an object, the records in the object's tables are deleted to preserve the referential integrity of the database. The Database Cleanup utility command cleans the database in one of two ways: top-down or bottom-up. Top-down deletes all rows from the child tables with a delete cascade. If a delete restrict is specified in the referential integrity, the delete cascade will fail and you will have to use the bottom-up method. To use the bottom-up method, specify `yes` for the

force

`FORCE` parameter in the command syntax, which first deletes the child tables, followed by the parent table.

Another way to trigger the bottom-up method is to specify the

loglevel

`LOGLEVEL` parameter as `2` in the command syntax. Specifying `0` logs nothing, and `1` only logs the delete statements from the top table.

loglevel

`LOGLEVEL 2` logs the delete statements from each deleted child table until the top table. Although selecting `2` triggers the bottom-up method, it cannot guarantee a successful deletion if there is a delete restrict in the referential integrity. To delete

records with a delete restrict, specify the

Windows

AIX

Solaris

force

400

`FORCE`parameter as `yes`.

You can expect a longer response time with the bottom-up method if the table contains many child tables. For example, the MEMBER table contains more than 500 child tables. For performance reasons, we recommend using the top-down method.

The Database Cleanup utility is configurable, extensible, and adaptable. Aside from the preset cleanup configurations, you can add new objects to the CLEANCONF database table to define which tables and rows to clean. Refer to adding a new configuration to the Database Cleanup utility.

If you have extended your database schema by creating new tables, you can use the Database Cleanup utility to clean your new tables. If you have changed your database schema (such as adding new columns to one table, changing the foreign key primary key relationship, or adding a new child table to the referential integrity path), the Database Cleanup utility will automatically adapt to the changes. If you change the column names, update the configuration data in the CLEANCONF table.

The Database Cleanup utility deletes records in child tables based on the delete rule of the referential integrity definition in the database schema. You can set the delete rule to `on delete cascade`, `on delete set null`, or `on delete restrict`. If you add new tables, ensure that the referential integrity and delete rule is properly defined. Otherwise, the Database Cleanup utility cannot work with your new tables.

Note: You should only run the Database Cleanup utility on a staging server to clean the staglog object. The staging database is different from the production database. The staging database only has configuration data without the operation data. Deleting configuration data might cause a delete cascade on the operation data. When the Stage Propagate utility propagates the deletion to the production database, this might cause a cascade delete to the operation data (which you want to keep). To clean configuration data, run the Database Cleanup utility on the production database.

Cleaning the database

To delete unused or obsolete objects from the WebSphere Commerce database using the Database Cleanup utility, select one of the following:

- account objects
- address objects
- available to promise (atp) inventory objects
- attachment objects
- auction objects

- auction log objects
- autobid log objects
- base item objects
- bid log objects
- cached objects
- calculation code objects
- campaign objects
- campaign statistic objects
- catalog entry objects
- catalog group objects
- contract objects
- coupon objects
- expected inventory record objects
- expected inventory record details object
- forum message objects
- fulfillment center objects
- inventory adjustment code objects
- inventory adjustment objects
- item specification objects
- member message objects
- message objects
- order objects
- organization objects
- Product Advisor statistic objects
- Product Comparison statistic objects
- Product Explorer statistic objects
- policy objects
- product set objects
- request for quote (RFQ) objects
- returned item objects
- rtnreason objects
- Sales Assistant statistic objects
- staged objects
- store objects
- user objects
- user traffic log objects
- vendor objects

Before you can use the Database Cleanup utility, you must follow the steps in configure the database.

Setting the PATH environment variables for WebSphere Commerce utilities

Before using the Database Cleanup utility or the staging server, ensure that you set the following in the PATH environment variables:

- NT

  DB2

  drive:\WebSphere\CommerceServer\bin and
  drive:\WebSphere\CommerceServer\sqllib\bin
- NT

  Oracle

  drive:\WebSphere\CommerceServer\bin and
  drive:\WebSphere\CommerceServer\ora81\bin
- 2000

  DB2

  drive:\Program Files\WebSphere\CommerceServer\bin and *drive*:\Program Files\WebSphere\CommerceServer\sqllib\bin
- 2000

  Oracle

  drive:\Program Files\WebSphere\CommerceServer\bin and *drive*:\Program Files\WebSphere\CommerceServer\ora81\bin
- AIX

  DB2

  /usr/WebSphere/CommerceServer/bin and /usr/lpp/db2_07_01
- AIX

  Oracle

  /usr/WebSphere/CommerceServer/bin and /usr/ora81/bin
- Solaris

  DB2

  /opt/WebSphere/CommerceServer/bin and /opt/IBMdb2/V7.1
- Solaris

  Oracle

  /opt/WebSphere/CommerceServer/bin and /opt/WebSphere/ora81/bin

- 400

  This step does not apply to OS/400 for iSeries.

Deleting account objects

To delete account objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:
   - Windows

     DB2

     ```
     dbclean -object account -type obsolete -db dbname -days
     daysold -loglevel loglevel
     ```
   - Windows

     Oracle

     ```
     dbclean -object account -type obsolete -db dbname -days daysold
     -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
     ```
   - AIX

     Solaris

     DB2

     ```
     . dbclean.sh -object account -type obsolete -db dbname -days
     daysold -loglevel loglevel
     ```
   - AIX

     Solaris

     Oracle

     ```
     . dbclean.sh -object account -type obsolete -db dbname -days
     daysold -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd
     password
     ```
   - 400

     DB2

     ```
     CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
     PASSWD(instance_password) OBJECT('account') TYPE('obsolete')
     LOGLEVEL(loglevel) DAYS(daysold)
     ```

Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on account objects, refer to examples of deleting objects.

Deleting address objects

To delete address objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:
   - Windows

     DB2

     ```
     dbclean -object address -type obsolete -db dbname -days daysold
     -loglevel loglevel
     ```
   - Windows

     Oracle

     ```
     dbclean -object address -type obsolete -db dbname -days daysold
     -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
     ```
   - AIX

     Solaris

     DB2

     ```
     . dbclean.sh -object address -type obsolete -db dbname -days daysold
     -loglevel loglevel
     ```
   - AIX

     Solaris

     Oracle

     ```
     . dbclean.sh -object address -type obsolete -db dbname -days daysold
     -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
     ```
   - 400

     DB2

     ```
     CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
     PASSWD(instance_password) OJBECT('address') TYPE('obsolete')
     LOGLEVEL(loglevel) DAYS(daysold)
     ```

Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting address objects, refer to examples of deleting objects.

Deleting available to promise inventory objects

To delete available to promise (ATP) inventory objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:
   - Windows DB2

     ```
     dbclean -object atp_inventory -type typename -db dbname -loglevel
     loglevel
     ```

   - Windows Oracle

     ```
     dbclean -object atp_inventory -type typename -db dbname -logLevel
     loglevel -dbtype oracle -dbuser user -dbpasswd password
     ```

   - AIX Solaris DB2

     ```
     . dbclean.sh -object atp_inventory -type typename -db dbname -loglevel
     loglevel
     ```

   - AIX Solaris Oracle

     ```
     . dbclean.sh -object atp_inventory -type typename -db dbname -loglevel
     loglevel -dbtype oracle -dbuser user -dbpasswd password
     ```

   - 400 DB2

     ```
     CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
     PASSWD(instance_password) OBJECT('atp_inventory') TYPE(typename)
     LOGLEVEL(loglevel) STATUS(orderstatus)
     ```

Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting ATP inventory objects, refer to examples of deleting objects.

Deleting attachment objects

To delete attachment objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

- Windows

  DB2

  ```
  dbclean -object attachment -type obsolete -db dbname -days daysold
  -loglevel loglevel
  ```

- Windows

  Oracle

  ```
  dbclean -object attachment -type obsolete -db dbname -days daysold
  -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
  ```

- AIX

  Solaris

  DB2

  ```
  . dbclean.sh -object attachment -type obsolete -db dbname -days
  daysold -loglevel loglevel
  ```

- AIX

  Solaris

  Oracle

  ```
  . dbclean.sh -object attachment -type obsolete -db dbname -days
  daysold -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd
  password
  ```

- 400

  DB2

```
CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
PASSWD(instance_password) OBJECT('attachment') TYPE('obsolete')
LOGLEVEL(loglevel) DAYS(daysold)
```

Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting attachment objects, refer to examples of deleting objects.

Deleting auction objects

To delete auction objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

- Windows

  DB2

  ```
  dbclean -object auction -type typename -db dbname -days daysold
  -loglevel loglevel
  ```

- Windows

  Oracle

  ```
  dbclean -object auction -type typename -db dbname -days daysold
  -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
  ```

- AIX

  Solaris

  DB2

  ```
  . dbclean.sh -object auction -type typename -db dbname -days daysold
  -loglevel loglevel
  ```

- AIX

  Solaris

  Oracle

  ```
  . dbclean.sh -object auction -type typename -db dbname -days daysold
  -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
  ```

- 400

DB2

```
CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
PASSWD(instance_password) OBJECT('auction') TYPE(typename)
LOGLEVEL(loglevel) DAYS(daysold)
```

Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`. **Note:** For the `-type` parameter, you can specify `settlement_closed` to indicate a completed auction record, or `retracted` to indicate a retracted auction.

4. Examine the `dbclean.log_yyyy.mm.dd_hh.mm.ss.zzz` file to verify that the command was successful.

For additional examples on deleting auction objects, refer to examples of deleting objects.

Deleting auction log objects

To delete auction log objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:
   - Windows

     DB2

     ```
     dbclean -object auctionlog -type obsolete -db dbname -days daysold
     -loglevel loglevel
     ```

   - Windows

     Oracle

     ```
     dbclean -object auctionlog -type obsolete -db dbname -days daysold
     -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
     ```

   - AIX

     Solaris

     DB2

     ```
     . dbclean.sh -object auctionlog -type obsolete -db dbname -days
     daysold -loglevel loglevel
     ```

   - AIX

     Solaris

     Oracle

```
. dbclean.sh -object auctionlog -type obsolete -db dbname -days
daysold -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd
password
```

- 400

  DB2

  ```
  CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
  PASSWD(instance_password) OBJECT('auctionlog') TYPE('obsolete')
  LOGLEVEL(loglevel) DAYS(daysold)
  ```

  Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting auction log objects, refer to examples of deleting objects.

Deleting auction log objects

To delete autobid log objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

   - Windows

     DB2

     ```
     dbclean -object autobidlog -type obsolete -db dbname -days daysold
     -loglevel loglevel
     ```

   - Windows

     Oracle

     ```
     dbclean -object autobidlog -type obsolete -db dbname -days daysold
     -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
     ```

   - AIX

     Solaris

     DB2

     ```
     . dbclean.sh -object autobidlog -type obsolete -db dbname -days
     daysold -loglevel loglevel
     ```

   - AIX

     Solaris

Oracle

```
. dbclean.sh -object autobidlog -type obsolete -db dbname -days
daysold -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd
password
```

- 400

  DB2

  ```
  CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
  PASSWD(instance_password) OBJECT('autobidlog') TYPE('obsolete')
  LOGLEVEL(loglevel) DAYS(daysold)
  ```

  Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting autobid log objects, refer to examples of deleting objects.

Deleting base item objects

To delete base item objects, which contain information about a general family of merchandise with the same name and description, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

   - Windows

     DB2

     ```
     dbclean -object baseitem -type obsolete -db dbname -loglevel loglevel
     ```

   - Windows

     Oracle

     ```
     dbclean -object baseitem -type obsolete -db dbname -loglevel loglevel
     -dbtype oracle -dbuser user -dbpasswd password
     ```

   - AIX

     Solaris

     DB2

     ```
     . dbclean.sh -object baseitem -type obsolete -db dbname -loglevel
     loglevel
     ```

   - AIX

Solaris

Oracle

```
. dbclean.sh -object baseitem -type obsolete -db dbname -loglevel
loglevel -dbtype oracle -dbuser user -dbpasswd password
```

- 400

DB2

```
CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
PASSWD(instance_password) OBJECT('baseitem') TYPE('obsolete')
LOGLEVEL(loglevel)
```

Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting base item objects, refer to examples of deleting objects.

Deleting bid log objects

To delete bid log objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

- Windows

DB2

```
dbclean -object bidlog -type obsolete -db dbname -days daysold
-loglevel loglevel
```

- Windows

Oracle

```
dbclean -object bidlog -type obsolete -db dbname -days daysold
-loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
```

- AIX

Solaris

DB2

```
. dbclean.sh -object bidlog -type obsolete -db dbname -days daysold
-loglevel loglevel
```

- AIX
  Solaris
  Oracle
  ```
  . dbclean.sh -object bidlog -type obsolete -db dbname -days daysold
  -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
  ```
- 400
  DB2
  ```
  CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
  PASSWD(instance_password) OBJECT('bidlog') TYPE('obsolete')
  LOGLEVEL(loglevel) DAYS(daysold)
  ```

  Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting bid log objects, refer to examples of deleting objects.

Deleting cachlog objects

To delete cachlog objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:
   - Windows
     DB2
     ```
     dbclean -object cachlog -type obsolete -db dbname -days daysold
     -loglevel loglevel
     ```
   - Windows
     Oracle
     ```
     dbclean -object cachlog -type obsolete -db dbname -days daysold
     -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
     ```
   - AIX
     Solaris
     DB2

```
. dbclean.sh -object cachlog -type obsolete -db dbname -days daysold
-loglevel loglevel
```

- AIX
  Solaris
  Oracle

```
. dbclean.sh -object cachlog -type obsolete -db dbname -days daysold
-loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
```

- 400
  DB2

```
CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
PASSWD(instance_password) OBJECT('cachlog') TYPE('obsolete')
LOGLEVEL(loglevel) DAYS(daysold)
```

Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting cached objects, refer to examples of deleting objects.

Deleting calculation code objects

To delete calculation code objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

- Windows
  DB2

```
dbclean -object calculation_code -type obsolete -db dbname -loglevel
loglevel
```

- Windows
  Oracle

```
dbclean -object calculation_code -type obsolete -db dbname -loglevel
loglevel -dbtype oracle -dbuser user -dbpasswd password
```

- AIX
  Solaris

DB2

```
. dbclean.sh -object calculation_code -type obsolete -db dbname
-loglevel loglevel
```

- AIX

  Solaris

  Oracle

  ```
  . dbclean.sh -object calculation_code -type obsolete -db dbname
  -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
  ```

- 400

  DB2

  ```
  CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
  PASSWD(instance_password) OBJECT('calculation_code') TYPE('obsolete')
  LOGLEVEL(loglevel)
  ```

  Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on calculation code objects, refer to examples of deleting objects.

Deleting campaign objects

To delete campaign objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

   - Windows

     DB2

     ```
     dbclean -object cpgnlog -type all -db dbname -loglevel loglevel
     ```

   - Windows

     Oracle

     ```
     dbclean -object cpgnlog -type all -db dbname -loglevel loglevel
     -dbtype oracle -dbuser user -dbpasswd password
     ```

   - AIX

     Solaris

DB2

```
. dbclean.sh -object cpgnlog -type all -db dbname -loglevel loglevel
```

- AIX

  Solaris

  Oracle

  ```
  . dbclean.sh -object cpgnlog -type all -db dbname -loglevel loglevel
  -dbtype oracle -dbuser user -dbpasswd password
  ```

- 400

  DB2

  ```
  CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
  PASSWD(instance_password) OBJECT('cpgnlog') TYPE('all')
  LOGLEVEL(loglevel)
  ```

  Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting campaign objects, refer to examples of deleting objects.

Deleting campaign statistic objects

To delete campaign statistic objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

   - Windows

     DB2

     ```
     dbclean -table cpgnstats -type all -db dbname -loglevel loglevel
     ```

   - Windows

     Oracle

     ```
     dbclean -table cpgnstats -type all -db dbname -loglevel loglevel
     -dbtype oracle -dbuser user -dbpasswd password
     ```

   - AIX

     Solaris

DB2

```
. dbclean.sh -object cpgnstats -type all -db dbname -loglevel loglevel
```

- AIX

  Solaris

  Oracle

```
. dbclean.sh -object cpgnstats -type all -db dbname -loglevel loglevel
-dbtype oracle -dbuser user -dbpasswd password
```

- 400

  DB2

```
CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
PASSWD(instance_password) OBJECT('cpgnstats') TYPE('all')
LOGLEVEL(loglevel)
```

Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting campaign statistic records from the CPGNSTATS table, refer to examples of deleting objects.

Deleting catalog entry objects

To delete catalog entry objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

- Windows

  DB2

```
dbclean -object catentry -type typename -db dbname -days daysold
-loglevel loglevel
```

- Windows

  Oracle

```
dbclean -object catentry -type typename -db dbname -days daysold
-loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
```

- AIX

  Solaris

DB2

```
. dbclean.sh -object catentry -type typename -db dbname -days
daysold -loglevel loglevel
```

- AIX

  Solaris

  Oracle

  ```
  . dbclean.sh -object catentry -type typename -db dbname -days daysold
  -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
  ```

- 400

  DB2

  ```
  CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
  PASSWD(instance_password) OBJECT('catentry') TYPE(typename)
  LOGLEVEL(loglevel) DAYS(daysold)
  ```

Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`. **Note:** For the `-type` parameter, you can specify `without_orderitems` to indicate catalog entries which are not referenced to any order item or `without_orderitems_iitems` to indicate catalog entries which are not referenced to any order item or interest list item.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting catalog entry objects, refer to examples of deleting objects.

Deleting catalog group objects

To delete catalog group objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

- Windows

  DB2

  ```
  dbclean -object catalog_group -type obsolete -db dbname -loglevel
  loglevel
  ```

- Windows

  Oracle

  ```
  dbclean -object catalog_group -type obsolete -db dbname -loglevel
  loglevel -dbtype oracle -dbuser user -dbpasswd password
  ```

- AIX

  Solaris

  DB2

  ```
  . dbclean.sh -object catalog_group -type obsolete -db dbname -loglevel
  loglevel
  ```

- AIX

  Solaris

  Oracle

  ```
  . dbclean.sh -object catalog_group -type obsolete -db dbname -loglevel
  loglevel -dbtype oracle -dbuser user -dbpasswd password
  ```

- 400

  DB2

  ```
  CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
  PASSWD(instance_password) OBJECT('catalog_group') TYPE('obsolete')
  LOGLEVEL(loglevel)
  ```

  Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting catalog group objects, refer to examples of deleting objects.

Deleting contract objects

To delete contract objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

- Windows

  DB2

  ```
  dbclean -object contract -type obsolete -db dbname -loglevel loglevel
  ```

- Windows

  Oracle

  ```
  dbclean -object contract -type obsolete -db database -loglevel
  loglevel -dbtype oracle -dbuser user -dbpasswd password
  ```

- AIX

  Solaris

  DB2

  ```
  . dbclean.sh -object contract -type obsolete -db dbname -loglevel
  loglevel
  ```

- AIX

  Solaris

  Oracle

  ```
  . dbclean.sh -object contract -type obsolete -db dbname -loglevel
  loglevel -dbtype oracle -dbuser user -dbpasswd password
  ```

- 400

  DB2

  ```
  CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
  PASSWD(instance_password) OBJECT('contract') TYPE('obsolete')
  LOGLEVEL(loglevel)
  ```

  Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`. Note:For the

  Windows

  AIX

  Solaris

  -dbor

  400

  `DATABASE`parameter, you must specify one of three database tables: `productset`, `tradeposn`, or `trading` .

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting contract objects, refer to examples of deleting objects.

Deleting coupon objects

To delete coupon objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

- Windows

  DB2

  ```
  dbclean -object coupon_promotion -type expired -db dbname -days
  daysold -loglevel loglevel
  ```

- Windows

  Oracle

  ```
  dbclean -object coupon_promotion -type expired -db dbname -days
  daysold -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd
  password
  ```

- AIX

  Solaris

  DB2

  ```
  . dbclean.sh -object coupon_promotion -type expired -db dbname -days
  daysold -loglevel loglevel
  ```

- AIX

  Solaris

  Oracle

  ```
  . dbclean.sh -object coupon_promotion -type expired -db dbname -days
  daysold -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd
  password
  ```

- 400

  DB2

  ```
  CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
  PASSWD(instance_password) OBJECT('coupon_promotion') TYPE('expired')
  LOGLEVEL(loglevel) DAYS(daysold)
  ```

  Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting coupon objects, refer to examples of deleting objects.

Deleting expected inventory record objects

To delete expected inventory record objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:
   - Windows

     DB2

     ```
     dbclean -object expected_inventory_records -type obsolete -db dbname
     -days daysold -loglevel loglevel
     ```
   - Windows

     Oracle

     ```
     dbclean -object expected_inventory_records -type obsolete -db dbname
     -days daysold -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd
     password
     ```
   - AIX

     Solaris

     DB2

     ```
     . dbclean.sh -object expected_inventory_records -type obsolete -db
     dbname -days daysold -loglevel loglevel
     ```
   - AIX

     Solaris

     Oracle

     ```
     . dbclean.sh -object expected_inventory_records -type obsolete -db
     dbname -days daysold -loglevel loglevel -dbtype oracle -dbuser user
     -dbpasswd password
     ```
   - 400

     DB2

     ```
     CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
     PASSWD(instance_password) OBJECT('expected_inventory_records')
     TYPE('obsolete') LOGLEVEL(loglevel) DAYS(daysold)
     ```

   Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.
4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting expected inventory record objects, refer to examples of deleting objects.

Deleting details about expected inventory record objects

To delete details about expected inventory record objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:
   - Windows

     DB2

     ```
     dbclean -object expected_inventory_record_details -type obsolete -db
     dbname -loglevel loglevel
     ```
   - Windows

     Oracle

     ```
     dbclean -object expected_inventory_record_details -type obsolete -db
     dbname -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd
     password
     ```
   - AIX

     Solaris

     DB2

     ```
     . dbclean.sh -object expected_inventory_record_details -type obsolete
     -db dbname -loglevel loglevel
     ```
   - AIX

     Solaris

     Oracle

     ```
     . dbclean.sh -object expected_inventory_record_details -type obsolete
     -db dbname -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd
     password
     ```
   - 400

     DB2

     ```
     CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
     PASSWD(instance_password) OBJECT('expected_inventory_record_details')
     TYPE('obsolete') LOGLEVEL(loglevel)
     ```

     Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log file to verify that the command was successful.

For additional examples on deleting details about expected inventory record objects, refer to examples of deleting objects.

Deleting forum message objects

To delete forum message objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

- Windows

  DB2

  ```
  dbclean -object forummsg -type obsolete -db dbname -days daysold
  -loglevel loglevel
  ```

- Windows

  Oracle

  ```
  dbclean -object forummsg -type obsolete -db dbname -days daysold
  -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
  ```

- AIX

  Solaris

  DB2

  ```
  . dbclean.sh -object forummsg -type obsolete -db dbname -days daysold
  -loglevel loglevel
  ```

- AIX

  Solaris

  Oracle

  ```
  . dbclean.sh -object forummsg -type obsolete -db dbname -days daysold
  -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
  ```

- 400

  DB2

  ```
  CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
  PASSWD(instance_password) OBJECT('forummsg') TYPE('obsolete')
  LOGLEVEL(loglevel) DAYS(daysold)
  ```

Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting forum message objects, refer to examples of deleting objects.

Deleting fulfillment center objects

To delete fulfillment center objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

- Windows

  DB2

  ```
  dbclean -object fulfillment_center -type obsolete -db dbname -loglevel
  loglevel
  ```

- Windows

  Oracle

  ```
  dbclean -object fulfillment_center -type obsolete -db dbname -loglevel
  loglevel -dbtype oracle -dbuser user -dbpasswd password
  ```

- AIX

  Solaris

  DB2

  ```
  . dbclean.sh -object fulfillment_center -type obsolete -db dbname
  -loglevel loglevel
  ```

- AIX

  Solaris

  Oracle

  ```
  . dbclean.sh -object fulfillment_center -type obsolete -db dbname
  -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
  ```

- 400

  DB2

  ```
  CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
  PASSWD(instance_password) OBJECT('fulfillment_center') TYPE('obsolete')
  LOGLEVEL(loglevel)
  ```

Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting fulfillment center objects, refer to examples of deleting objects.

Deleting inventory code adjustment objects

To delete inventory code adjustment objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:
   - Windows DB2

     ```
     dbclean -object inventory_adjustment_codes -type obsolete -db dbname
     -loglevel loglevel
     ```

   - Windows Oracle

     ```
     dbclean -object inventory_adjustment_codes -type obsolete -db dbname
     -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
     ```

   - AIX Solaris DB2

     ```
     . dbclean.sh -object inventory_adjustment_codes -type obsolete -db
     dbname -loglevel loglevel
     ```

   - AIX Solaris Oracle

     ```
     . dbclean.sh -object inventory_adjustment_codes -type obsolete -db
     dbname -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd
     password
     ```

   - 400 DB2

```
CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
PASSWD(instance_password) DBTABLE('inventory_adjustment_codes')
TYPE('obsolete') LOGLEVEL(loglevel)
```

Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting inventory code adjustment objects, refer to examples of deleting objects.

Deleting inventory adjustment objects

To delete inventory adjustment objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

- Windows

  DB2

  ```
  dbclean -object inventory_adjustments -type obsolete -db dbname -days
  daysold -loglevel loglevel
  ```

- Windows

  Oracle

  ```
  dbclean -object inventory_adjustments -type obsolete -db dbname -days
  daysold -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd
  password
  ```

- AIX

  Solaris

  DB2

  ```
  . dbclean.sh -object inventory_adjustments -type obsolete -db dbname
  -days daysold -loglevel loglevel
  ```

- AIX

  Solaris

  Oracle

  ```
  . dbclean.sh -object inventory_adjustments -type obsolete -db dbname
  -days daysold -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd
  password
  ```

- 400

  DB2

  ```
  CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
  PASSWD(instance_password) OBJECT('inventory_adjustments')
  TYPE('obsolete') LOGLEVEL(loglevel) DAYS(daysold)
  ```

  Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting inventory adjustment objects, refer to examples of deleting objects.

Deleting specified item information objects

To delete specified item information objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

- Windows

  DB2

  ```
  dbclean -object itemspecification -type obsolete -db dbname -loglevel
  loglevel
  ```

- Windows

  Oracle

  ```
  dbclean -object itemspecification -type obsolete -db dbname -loglevel
  loglevel -dbtype oracle -dbuser user -dbpasswd password
  ```

- AIX

  Solaris

  DB2

  ```
  . dbclean.sh -object itemspecification -type obsolete -db dbname
  -loglevel loglevel
  ```

- AIX

  Solaris

  Oracle

```
. dbclean.sh -object itemspecification -type obsolete -db dbname
-loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
```

- 400

  DB2

  ```
  CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
  PASSWD(instance_password) OBJECT('itemspecification') TYPE('obsolete')
  LOGLEVEL(loglevel)
  ```

  Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on specified item information objects, refer to examples of deleting objects.

Deleting member message relationship objects

To delete member message relationship objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

   - Windows

     DB2

     ```
     dbclean -object msgmemrel -type obsolete -db dbname -days daysold
     -loglevel loglevel
     ```

   - Windows

     Oracle

     ```
     dbclean -object msgmemrel -type obsolete -db dbname -days daysold
     -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
     ```

   - AIX

     Solaris

     DB2

     ```
     . dbclean.sh -object msgmemrel -type obsolete -db dbname -days daysold
     -loglevel loglevel
     ```

   - AIX

     Solaris

Oracle

```
. dbclean.sh -object msgmemrel -type obsolete -db dbname -days daysold
-loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
```

- 400

DB2

```
CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
PASSWD(instance_password) OBJECT('msgmemrel') TYPE('obsolete')
LOGLEVEL(loglevel) DAYS(daysold)
```

Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting message member relationship objects, refer to examples of deleting objects.

Deleting message objects

To delete message objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

- Windows

DB2

```
dbclean -object message -type obsolete -db dbname -days daysold
-loglevel loglevel
```

- Windows

Oracle

```
dbclean -object message -type obsolete -db dbname -days daysold
-loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
```

- AIX

Solaris

DB2

```
. dbclean.sh -object message -type obsolete -db dbname -days daysold
-loglevel loglevel
```

- AIX

Solaris

Oracle

```
. dbclean.sh -object message -type obsolete -db dbname -days daysold
-loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
```

- 400

DB2

```
CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
PASSWD(instance_password) OBJECT('message') TYPE('obsolete')
LOGLEVEL(loglevel) DAYS(daysold)
```

Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting message objects, refer to examples of deleting objects.

Deleting order objects

To delete order objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

- Windows

DB2

```
dbclean -object order -type typename -db dbname -days daysold
-loglevel loglevel
```

- Windows

Oracle

```
dbclean -object order -type typename -db dbname -days daysold
-loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
```

- AIX

Solaris

DB2

```
. dbclean.sh -object order -type typename -db dbname -days daysold
-loglevel loglevel
```

- AIX
  Solaris
  Oracle

  ```
  . dbclean.sh -object order -type typename -db dbname -days daysold
  -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
  ```

- 400
  DB2

  ```
  CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
  PASSWD(instance_password) OBJECT('order') TYPE(typename)
  LOGLEVEL(loglevel) DAYS(daysold) STATUS(orderstatus)
  ```

  Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`. **Note:** For the `-type` parameter, you can specify one of six different types: `completed` to indicate a completed order, `canceled` to indicate a canceled order, `shipped` to indicate a shipped order, `deposited` to indicate an order which has been deposited, `stale_guest` to indicate stale orders from guest customers, or `stale_non_guest` to indicate stale orders by customers who are not guests.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting order objects, refer to examples of deleting objects.

Deleting organization objects

To delete organization objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

- Windows
  DB2

  ```
  dbclean -object organization -type specified -db dbname -loglevel
  loglevel -name organizationid
  ```

- Windows
  Oracle

  ```
  dbclean -object organization -type specified -db dbname -loglevel
  loglevel -dbtype oracle -dbuser user -dbpasswd password -name
  organizationid
  ```

- AIX

  Solaris

  DB2

  ```
  . dbclean.sh -object organization -type specified -db dbname -loglevel
  loglevel -name organizationid
  ```

- AIX

  Solaris

  Oracle

  ```
  . dbclean.sh -object organization -type specified -db dbname -loglevel
  loglevel -dbtype oracle -dbuser user -dbpasswd password -name
  organizationid
  ```

- 400

  DB2

  ```
  CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
  PASSWD(instance_password) OBJECT('organization') TYPE('specified')
  LOGLEVEL(loglevel) NAME(organizationid)
  ```

  Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`. **Note:** For the `-type` parameter, you can specify `organization` to indicate any organization record.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting organization objects, refer to examples of deleting objects.

Deleting policy objects

To delete policy objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

- Windows

  DB2

  ```
  dbclean -object policy -type obsolete -db dbname -days daysold
  -loglevel loglevel
  ```

- Windows

Oracle

```
dbclean -object policy -type obsolete -db dbname -days daysold
-loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
```

- AIX

  Solaris

  DB2

  ```
  . dbclean.sh -object policy -type obsolete -db dbname -days
  daysold -loglevel loglevel
  ```

- AIX

  Solaris

  Oracle

  ```
  . dbclean.sh -object policy -type obsolete -db dbname -days
  daysold -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd
  password
  ```

- 400

  DB2

  ```
  CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
  PASSWD(instance_password) OBJECT('policy') TYPE('obsolete')
  LOGLEVEL(loglevel) DAYS(daysold)
  ```

  Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting policy objects, refer to examples of deleting objects.

Deleting Product Advisor statistic objects

To delete Product Advisor statistic objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

   - Windows

     DB2

     ```
     dbclean -object pastats -type all -db dbname -loglevel loglevel
     ```

- Windows

  Oracle

  ```
  dbclean -object pastats -type all -db dbname -loglevel logLevel
  -dbtype oracle -dbuser user -dbpasswd password
  ```

- AIX

  Solaris

  DB2

  ```
  . dbclean.sh -object pastats -type all -db dbname -loglevel loglevel
  ```

- AIX

  Solaris

  Oracle

  ```
  . dbclean.sh -object pastats -type all -db dbname -loglevel loglevel
  -dbtype oracle -dbuser user -dbpasswd password
  ```

- 400

  DB2

  ```
  CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
  PASSWD(instance_password) OBJECT('pastats') TYPE('obsolete')
  LOGLEVEL(loglevel)
  ```

  Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting Product Advisor statistic objects, refer to examples of deleting objects.

Deleting Product Comparison statistic objects

To delete Product Comparison (Product Advisor) statistic objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

- Windows

  DB2

  ```
  dbclean -object pcstats -type all -db dbname -loglevel loglevel
  ```

- Windows

  Oracle

  ```
  dbclean -object pcstats -type all -db dbname -loglevel loglevel
  -dbtype oracle -dbuser user -dbpasswd password
  ```

- AIX

  Solaris

  DB2

  ```
  . dbclean.sh -object pcstats -type all -db dbname -loglevel loglevel
  ```

- AIX

  Solaris

  Oracle

  ```
  . dbclean.sh -object pcstats -type all -db dbname -loglevel loglevel
  -dbtype oracle -dbuser user -dbpasswd password
  ```

- 400

  DB2

  ```
  CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
  PASSWD(instance_password) OBJECT('pcstats') TYPE('all')
  LOGLEVEL(loglevel)
  ```

  Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting Product Comparison (Product Advisor) statistic objects, refer to examples of deleting objects.

Deleting Product Explorer statistic objects

To delete Product Explorer (Product Advisor) statistic objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

   - Windows

     DB2

     ```
     dbclean -object pestats -type all -db dbname -loglevel loglevel
     ```

- Windows

  Oracle

  ```
  dbclean -object pestats -type all -db dbname -loglevel loglevel
  -dbtype oracle -dbuser user -dbpasswd password
  ```

- AIX

  Solaris

  DB2

  ```
  . dbclean.sh -object pestats -type all -db dbname -loglevel loglevel
  ```

- AIX

  Solaris

  Oracle

  ```
  . dbclean.sh -object pestats -type all -db dbname -loglevel loglevel
  -dbtype oracle -dbuser user -dbpasswd password
  ```

- 400

  DB2

  ```
  CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
  PASSWD(instance_password) OBJECT('pestats') TYPE('all')
  LOGLEVEL(loglevel)
  ```

  Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting Product Explorer (Product Advisor) statistic objects, refer to examples of deleting objects.

Deleting product set objects

To delete product set objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

- Windows

  DB2

  ```
  dbclean -object product_sets -type obsolete -db dbname -loglevel
  loglevel
  ```

- Windows

  Oracle

  ```
  dbclean -object product_sets -type obsolete -db dbname -loglevel
  loglevel -dbtype oracle -dbuser user -dbpasswd password
  ```

- AIX

  Solaris

  DB2

  ```
  . dbclean.sh -object product_sets -type obsolete -db dbname -loglevel
  loglevel
  ```

- AIX

  Solaris

  Oracle

  ```
  . dbclean.sh -object product_sets -type obsolete -db dbname -loglevel
  loglevel -dbtype oracle -dbuser user -dbpasswd password
  ```

- 400

  DB2

  ```
  CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
  PASSWD(instance_password) OBJECT('product_sets') TYPE('obsolete')
  LOGLEVEL(loglevel)
  ```

  Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on product set objects, refer to examples of deleting objects.

Deleting request for quote objects

To delete request for quote (RFQ) objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

   - Windows

     DB2

     ```
     dbclean -table rfq -type obsolete -db dbname -loglevel loglevel
     ```

- Windows

  Oracle

  ```
  dbclean -object rfq -type obsolete -db dbname -loglevel loglevel
  -dbtype oracle -dbuser user -dbpasswd password
  ```

- AIX

  Solaris

  DB2

  ```
  . dbclean.sh -object rfq -type obsolete -db dbname -loglevel loglevel
  ```

- AIX

  Solaris

  Oracle

  ```
  . dbclean.sh -object rfq -type obsolete -db dbname -loglevel loglevel
  -dbtype oracle -dbuser user -dbpasswd password
  ```

- 400

  DB2

  ```
  CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
  PASSWD(instance_password) OBJECT('rfq') TYPE('obsolete')
  LOGLEVEL(loglevel)
  ```

  Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting RFQ objects, refer to examples of deleting objects.

Deleting returned item objects

To delete returned item objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

- Windows

  DB2

  ```
  dbclean -object rma -type typename -db dbname -days daysold -loglevel
  loglevel
  ```

- Windows

  Oracle

  ```
  dbclean -object rma -type typename -db dbname -days daysold -loglevel
  loglevel -dbtype oracle -dbuser user -dbpasswd password
  ```

- AIX

  Solaris

  DB2

  ```
  . dbclean.sh -object rma -type typename -db dbname -days
  daysold -loglevel loglevel
  ```

- AIX

  Solaris

  Oracle

  ```
  . dbclean.sh -object rma -type typename -db dbname -days
  daysold -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd
  password
  ```

- 400

  DB2

  ```
  CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
  PASSWD(instance_password) OBJECT('rma') TYPE(typename)
  LOGLEVEL(loglevel) DAYS(daysold)
  ```

  Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`. **Note:** For the `-type` parameter, you can specify `abandoned` to indicate an abandoned record, `canceled` to indicate a canceled record, `not_approved` to indicate a rejected record, `approved_or_partly_approved` to indicate an approved or partly approved record, or `completed` to indicate a completed record.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting returned item objects, refer to examples of deleting objects.

Deleting return reason objects

To delete the reasons for customer dissatisfaction with merchandise, or simply return reason objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

- Windows

  DB2

  ```
  dbclean -object rtnreason -type obsolete -db dbname -loglevel loglevel
  ```

- Windows

  Oracle

  ```
  dbclean -object rtnreason -type obsolete -db dbname -loglevel loglevel
  -dbtype oracle -dbuser user -dbpasswd password
  ```

- AIX

  Solaris

  DB2

  ```
  . dbclean.sh -object rtnreason -type obsolete -db dbname -loglevel
  loglevel
  ```

- AIX

  Solaris

  Oracle

  ```
  . dbclean.sh -object rtnreason -type obsolete -db dbname -loglevel
  loglevel -dbtype oracle -dbuser user -dbpasswd password
  ```

- 400

  DB2

  ```
  CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
  PASSWD(instance_password) OBJECT('rtnreason') TYPE('obsolete')
  LOGLEVEL(loglevel)
  ```

  Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting return reason objects, refer to examples of deleting objects.

Deleting Sales Assistant statistic objects

To delete Sales Assistant (Product Advisor) statistic objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

- Windows

  DB2

  ```
  dbclean -object sastats -type all -db dbname -loglevel loglevel
  ```

- Windows

  Oracle

  ```
  dbclean -object sastats -type all -db dbname -loglevel loglevel
  -dbtype oracle -dbuser user -dbpasswd password
  ```

- AIX

  Solaris

  DB2

  ```
  . dbclean.sh -object sastats -type all -db dbname -loglevel loglevel
  ```

- AIX

  Solaris

  Oracle

  ```
  . dbclean.sh -object sastats -type all -db dbname -loglevel loglevel
  -dbtype oracle -dbuser user -dbpasswd password
  ```

- 400

  DB2

  ```
  CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
  PASSWD(instance_password) OBJECT('sastats') TYPE('all')
  LOGLEVEL(loglevel)
  ```

  Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting Sales Assistant (Product Advisor) statistic objects, refer to examples of deleting objects.

Deleting store objects

To delete store objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

- Windows

  DB2

  ```
  dbclean -object store -type specified -db dbname -loglevel loglevel
  -name storeid
  ```
- Windows

  Oracle

  ```
  dbclean -object store -type specified -db dbname -loglevel loglevel
  -dbtype oracle -dbuser user -dbpasswd password -name storeid
  ```
- AIX

  Solaris

  DB2

  ```
  . dbclean.sh -object store -type specified -db dbname -loglevel
  loglevel -name storeid
  ```
- AIX

  Solaris

  Oracle

  ```
  . dbclean.sh -object store -type specified -db dbname -loglevel
  loglevel -dbtype oracle -dbuser user -dbpasswd password -name storeid
  ```
- 400

  DB2

  ```
  CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
  PASSWD(instance_password) OBJECT('store') TYPE('specified')
  LOGLEVEL(loglevel) NAME(storeid)
  ```

  Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting store objects, refer to examples of deleting objects.

Deleting user objects

To delete user objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.

3. Type the following:
   - Windows

     DB2

     ```
     dbclean -object user -type typename -db dbname -days daysold -loglevel
     loglevel
     ```
   - Windows

     Oracle

     ```
     dbclean -object user -type typename -db dbname -days daysold -loglevel
     loglevel -dbtype oracle -dbuser user -dbpasswd password
     ```
   - AIX

     Solaris

     DB2

     ```
     . dbclean.sh -object user -type typename -db dbname -days
     daysold -loglevel loglevel
     ```
   - AIX

     Solaris

     Oracle

     ```
     . dbclean.sh -object user -type typename -db dbname -days
     daysold -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd
     password
     ```
   - 400

     DB2

     ```
     CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
     PASSWD(instance_password) OBJECT('user') TYPE(typename)
     LOGLEVEL(loglevel) DAYS(daysold)
     ```

   Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`. **Note:** For the `-type` parameter, you can specify `guest` to indicate a guest customer or `registered` to indicate a registered customer .
4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting user objects, refer to examples of deleting objects.

Deleting user traffic log objects

To delete user traffic log objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:
   - Windows

     DB2

     ```
     dbclean -object usrtraffic -type obsolete -db dbname -days
     daysold -loglevel loglevel
     ```
   - Windows

     Oracle

     ```
     dbclean -object usrtraffic -type obsolete -db dbname -days daysold
     -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd password
     ```
   - AIX

     Solaris

     DB2

     ```
     . dbclean.sh -object usrtraffic -type obsolete -db dbname -days
     daysold -loglevel loglevel
     ```
   - AIX

     Solaris

     Oracle

     ```
     . dbclean.sh -object usrtraffic -type obsolete -db dbname -days
     daysold -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd
     password
     ```
   - 400

     DB2

     ```
     CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
     PASSWD(instance_password) OBJECT('usrtraffic') TYPE('obsolete')
     LOGLEVEL(loglevel) DAYS(daysold)
     ```

   Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.
4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting user traffic log objects, refer to examples of deleting objects.

Deleting vendor objects

To delete vendor objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:
   - Windows DB2
     ```
     dbclean -object vendor -type obsolete -db dbname -loglevel loglevel
     ```
   - Windows Oracle
     ```
     dbclean -object vendor -type obsolete -db dbname -loglevel loglevel
     -dbtype oracle -dbuser user -dbpasswd password
     ```
   - AIX Solaris DB2
     ```
     . dbclean.sh -object vendor -type obsolete -db dbname -loglevel
     loglevel
     ```
   - AIX Solaris Oracle
     ```
     . dbclean.sh -object vendor -type obsolete -db dbname -loglevel
     loglevel -dbtype oracle -dbuser user -dbpasswd password
     ```
   - 400 DB2
     ```
     CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
     PASSWD(instance_password) OBJECT('vendor') TYPE('obsolete')
     LOGLEVEL(loglevel)
     ```

     Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.
4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting vendor objects, refer to examples of deleting objects.

Adding a new configuration to the Database Cleanup utility

To add a new configuration to the Database Cleanup utility, use the following syntax as a reference. For example, object o1 consists of table R1, which contains the following columns: col1, col2, lastupdate, and col3. To configure the Database Cleanup utility to delete all objects with col1 > 10, and where lastupdate is `n` days ago, do the following:

DB2

1.

Windows

AIX

Solaris

Open a DB2 command prompt.

2. Type the following:
```
db2 insert into cleanconf (objectname, type, statement, namearg,
sequence, daysarg) values ('o1', 'obsolete', 'delete from r1 where col1 >
10 and (days(CURRENT TIMESTAMP) - days(lastupdate)) > ?', 'no', 1, 'yes')
```

1.

400

Run the following SQL statement:
```
insert into cleanconf (objectname, type, statement, namearg, sequence,
daysarg) values ('o1', 'obsolete', 'delete from r1 where col1 > 10 and
(days(CURRENT TIMESTAMP) - days(lastupdate)) > ?', 'no', 1, 'yes')
```

Oracle

1. Open an SQLPlus command window.
2. Type the following:
```
insert into cleanconf (objectname, type, statement, namearg, sequence,
daysarg) values ('o1', 'obsolete', 'delete from r1 where col1 > 10 and
(sysdate - lastupdate) > ?', 'no', 1, 'yes')
```

where `?` is replaced by the `-days` parameter from the following command line. The `'no'` indicates that the name parameter is not used in the statement. The `'yes'` indicates that the `-days` parameter is used in the statement. `'obsolete'` describes the cleanup type for object o1. You can use other words, but you must use the same word in the `-type` argument when you invoke the Database Cleanup utility command.

Example

To invoke the Database Cleanup utility command to clean the records which have been in existence for two days from the new table, type the following:

- Windows

  DB2

  ```
  dbclean -object o1 -db dbname -type obsolete -days 2 -loglevel 1
  ```

- Windows

  Oracle

  ```
  dbclean -object o1 -db dbname -type obsolete -days 2 -loglevel 1 -dbtype
  oracle -dbuser user -dbpasswd password
  ```

- AIX

  Solaris

  DB2

  ```
  . dbclean.sh -object o1 -db dbname -type obsolete -days 2 -loglevel 1
  ```

- AIX

  Solaris

  Oracle

  ```
  . dbclean.sh -object o1 -db dbname -type obsolete -days 2 -loglevel 1
  -dbtype oracle -dbuser user -dbpasswd password
  ```

- 400

  DB2

  ```
  CLNWCSDB DATABASE(dbname)
  SCHEMA(schema_name) PASSWD(instance_password) OBJECT('o1')
  TYPE('obsolete') LOGLEVEL(1) DAYS(2)
  ```

Note: For the Oracle *dbname* parameter, use `host:port:sid.` For example, `myhost:1521:mydb`.

Configuring the database

Before using the Stage Copy utility, the Stage Propagate utility, or the Database Cleanup utility, do the following:

1. Set the PATH environment variables.
2. Windows

   AIX

Solaris

If you are using a DB2 database, configure the staging database and the production database. Issue the following commands:

```
db2 update db config for db_name using STMTHEAP 60000
db2 update db config for db_name using LOCKLIST 512
db2 update db config for db_name using LOGPRIMARY 80
db2 update db config for db_name using LOGBUFSZ 512
db2 update db config for db_name using DBHEAP 2048
db2 update db config for db_name using APPLHEAPSZ 2048
db2 update db config for db_name using PCKCACHESZ 8200
db2 update db config for db_name using STAT_HEAP_SZ 2048
db2 update db config for db_name using APP_CTL_HEAP_SZ 4096
```

where *db_name* is the name of your database.

3.

Windows

AIX

Solaris

Increase the buffer pool size for improved performance. Determine the optimum buffer pool size based on your DB2 database size and available memory. Run the following command to change the default buffer pool size:

```
db2 connect to db_name
db2 alter bufferpool IBMDEFAULTBP size n
db2 terminate
```

where *n* is the optimum buffer pool size.

4.

400

Log on as a user profile with secofr authority and a cssid of something other than 65535.

Database Cleanup utility objects

The Database Cleanup utility refers to the CLEANCONF table to determine which tables and which rows to delete when a particular object and object type are specified. The following table describes preconfigured deletion scenarios from the CLEANCONF table. You can configure your own deletion objects by adding similar rows to the CLEANCONF table.

Object	Type	Statements
account	obsolete	`delete from account where markfordelete = 1 and trdtype_id = 0 and trading_id not in (select account_id from trading) and trading_id not in (select distinct account_id from ordpaymthd)`

address	obsolete	delete from address where status = 'T' and (days(CURRENT TIMESTAMP) - days(lastcreate)) >= ? and (address_id not in (select distinct address_id from orderitems where address_id is not null)) and (address_id not in (select distinct address_id from orders where address_id is not null)) and (address_id not in (select distinct allocaddress_id from orderitems where allocaddress_id is not null))
atp_inventory	obsolete	delete from receipt where qtyonhand = 0 and qtyinkits = 0 and receipt_id not in (select distinct receipt_id from ordpickhst where receipt_id is not null) and receipt_id not in (select distinct receipt_id from ordshiphst where receipt_id is not null)
attachment	obsolete	delete from attachment where days(current timestamp) - days(timeupdated) >=? and attachment_id not in (select attachment_id from trdattach)
auction	retracted	delete from auction where austatus = 'R' and (days(CURRENT TIMESTAMP) - days(updatetime)) >= ?
auction	settlement_closed	delete from auction where austatus = 'SC' and (days(CURRENT TIMESTAMP) - days(updatetime)) >= ?
auctionlog	obsolete	delete from auctionlog where (days(CURRENT TIMESTAMP) - days(actiontime)) >= ?
autobidlog	obsolete	delete from autobidlog where (days(CURRENT TIMESTAMP) - days(actiontime)) >= ?

baseitem	obsolete	delete from baseitem where markfordelete = 1 and baseitem_id not in (select baseitem_id from catentry) and baseitem_id not in (select distinct baseitem_id from itemspc where markfordelete = 0 and itemspc_id in (select distinct itemspc_id from orderitems) or itemspc_id in (select distinct itemspc_id from oicomplist) or itemspc_id in (select distinct itemspc_id from versionspc where versionspc_id in (select distinct versionspc_id from receipt)) or itemspc_id in (select distinct itemspc_id from radetail) or itemspc_id in (select distinct itemspc_id from bkordalloc) or itemspc_id in (select distinct itemspc_id from invreserve) or itemspc_id in (select distinct itemspc_id from rmaitem) or itemspc_id in (select distinct itemspc_id from rmaitemcmp) or itemspc_id in (select distinct itemspc_id from catentry))
bidlog	obsolete	delete from bidlog where (days(CURRENT TIMESTAMP) - days(actiontime)) >= ?
cachlog	obsolete	delete from cachlog where (days(CURRENT TIMESTAMP) -days(cacstmp)) >= ?

calculation_code	obsolete	delete from calcode where published = 2 and calcode_id not in (select distinct calcode_id from ordadjust where calcode_id is not null) and calcode_id not in (select distinct calcode_id from stencalusg where calcode_id is not null) and calcode_id not in (select distinct calcode_id from ordcalcd where calcode_id is not null) and calcode_id not in (select distinct calcode_id from ordicalcd where calcode_id is not null)
catalog_group	obsolete	delete from catgroup where markfordelete = 1
catentry	without_orderitems	delete from catentry where markfordelete = 1 and(days(CURRENT TIMESTAMP) - days(lastupdate)) >= ? and catentry_id not in (select distinct catentry_id from auction) and catentry_id not in (select distinct catentry_id from orderitems where catentry_id is not null) and catentry_id not in (select distinct catentry_id from oicomplist where catentry_id is not null) and catentry_id not in (select distinct catentry_id from rmaitem where catentry_id is not null) and catentry_id not in (select distinct catentry_id from offer where offer_id in (select distinct offer_id from orderitems))

catentry	without_orderitems-iitems	delete from catentry where markfordelete = 1 and (days(CURRENT TIMESTAMP) - days(lastupdate)) >= ? and catentry_id not in (select distinct catentry_id from auction) and catentry_id not in (select distinct catentry_id from orderitems) and catentry_id not in (select distinct catentry_id from oicomplist where catentry_id is not null)and catentry_id not in (select distinct catentry_id from iitem) and catentry_id not in (select distinct catentry_id from rmaitem where catentry_id is not null) and catentry_id not in (select distinct catentry_id from offer where offer_id in (select distinct offer_id from orderitems))

contract	obsolete	1.delete from trading where markfordelete = 1 and trdtype_id = 1 and trading_id not in (select distinct trading_id from orderitems) and trading_id not in (select distinct trading_id from rma) and trading_id not in (select distinct trading_id from ordpaymthd) and trading_id not in (select distinct account_id from ordpaymthd) 2.delete from productset where markfordelete = 1 and productset_id not in (select distinct productset_id from tradeposcn where tradeposcn_id in (select distinct tradeposcn_id from offer where offer_id in (select distinct offer_id from orderitems))) 3.delete from tradeposcn where markfordelete = 1 and tradeposcn_id not in (select distinct tradeposcn_id from offer where offer_id in (select distinct offer_id from orderitems))
coupon_promotion	expired	delete from cppmn where days(current timestamp) - days(enddate) >=?
cpgnlog	obsolete	delete from cpgnlog
cpgnstats	obsolete	delete from cpgnstats
expected_inventory_ records	obsolete	delete from ra where markfordelete = 1 and ra_id not in (select distinct ra_id from receipt, radetail where receipt.radetail_id = radetail.radetail_id)
expected_inventory_ record_details	obsolete	delete from radetail where markfordelete = 1 and radetail_id not in (select distinct radetail_id from receipt)
forummsg	obsolete	delete from forummsg where msgstatus = 'D' or (days(CURRENT TIMESTAMP) - days(posttime)) >= ?

fulfillment_center	obsolete	delete from ffmcenter where markfordelete = 1 and ffmcenter_id not in (select distinct ffmcenter_id from radetail) and ffmcenter_id not in (select distinct ffmcenter_id from inventory) and ffmcenter_id not in (select distinct ffmcenter_id from rma) and ffmcenter_id not in (select distinct ffmcenter_id from orderitems) and ffmcenter_id not in (select distinct allocffmc_id from orderitems) and ffmcenter_id not in (select distinct ffmcenter_id from store) and ffmcenter_id not in (select distinct rtnffmctr_id from store) and ffmcenter_id not in (select distinct ffmcenter_id from receipt) and ffmcenter_id not in (select distinct ffmcenter_id from auction) and ffmcenter_id not in (select distinct ffmcenter_id from auctionlog)
inventory_adjustments	obsolete	delete from invadjust where days(CURRENT TIMESTAMP) - days(adjustmentdate) >= ?
inventory_adjustment_ codes	obsolete	delete from invadjcode where markfordelete = 1 and invadjcode_id not in (select distinct invadjcode_id from invadjust)

itemspecification	obsolete	delete from itemspc where markfordelete = 1 and itemspc_id not in (select distinct itemspc_id from orderitems) and itemspc_id not in (select distinct itemspc_id from oicomplist) and itemspc_id not in (select distinct itemspc_id from versionspc where versionspc_id in (select distinct versionspc_id from receipt)) and itemspc_id not in (select distinct itemspc_id from radetail) and itemspc_id not in (select distinct itemspc_id from bkordalloc) and itemspc_id not in (select distinct itemspc_id from invreserve) and itemspc_id not in (select distinct itemspc_id from rmaitem) and itemspc_id not in (select distinct itemspc_id from rmaitemcmp) and itemspc_id not in (select distinct itemspc_id from catentry)
message	obsolete	delete from message where message_id not in (select message_id from msgmemrel) or (days(CURRENT TIMESTAMP) - days(posttime)) >= ?
msgmemrel	obsolete	delete from msgmemrel where message_id in (select m.message_id from message ms, msgmemrel m where ms.message_id = m.message_id and (status = 'D' or ((status = 'O' or sendstat = 'S') and (days(CURRENT TIMESTAMP) - days(posttime)) >= ?)))
order	canceled	delete from orders where status ='X' and (days(CURRENT TIMESTAMP) - days(lastupdate)) >= ?
order	completed	delete from orders where status = 'C' and (days(CURRENT TIMESTAMP) - days(lastupdate)) >= ?

order	deposited	delete from orders where status ='D' and (days(CURRENT TIMESTAMP) - days(lastupdate)) >= ?
order	shipped	delete from orders where status ='S' and (days(CURRENT TIMESTAMP) - days(lastupdate)) >= ?
order	stale_guest	delete from orders where (status ='P' or status = 'I' or status = 'W' or status = 'N') and (days(CURRENT TIMESTAMP) - days(lastupdate)) >= ? and orders.member_id in (select distinct users_id from users where registertype = 'G') and (orders_id not in (select distinct orders_id from orderitems where inventorystatus != 'NALC' and inventorystatus is not null))
order	stale_non_guest	delete from orders where (status ='P' or status = 'I' or status = 'W' or status = 'N') and (days(CURRENT TIMESTAMP) - days(lastupdate)) >= ? and orders.member_id in (select distinct users_id from users where registertype != 'G') and (orders_id not in (select distinct orders_id from orderitems where inventorystatus != 'NALC' and inventorystatus is not null))
organization	obsolete	delete from member where member_id in (select orgentity_id from orgentity where orgentity_id = ?)
pastats	obsolete	delete from pastats
pcstats	obsolete	delete from pcstats
pestats	obsolete	delete from pestats
policy	obsolete	delete from policy where days(current timestamp) - days(endtime) > ? and policy_id not in (select distinct policy_id from ordpaymthd) and policy_id not in (select distinct policy_id from rma)

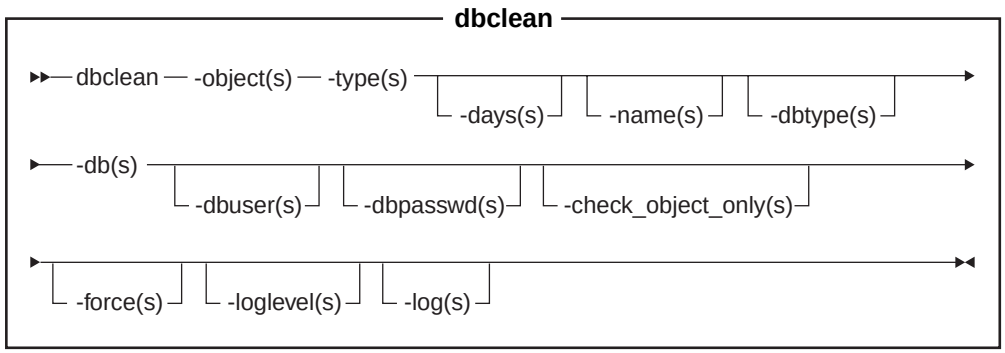
rfq	obsolete	delete from trading where markfordelete = 1 and trdtype_id in (2, 3, 4) and trading_id not in (select distinct trading_id from orderitems) and trading_id not in (select distinct trading_id from rma) and trading_id not in (select distinct trading_id from ordpaymthd) and trading_id not in (select distinct account_id from ordpaymthd)
rma	abandoned	delete from rma where status in ('PRC', 'EDT') and (days(CURRENT TIMESTAMP) - days(lastupdate)) >= ? and rma_id not in (select rma_id from rmaitem where rmaitem.status in ('APP', 'MAN')) and rma_id not in (select rma_id from rtnreceipt)
rma	approved_or_partly_ approved	delete from rma where status = 'PND' and (days(CURRENT TIMESTAMP) - days(lastupdate)) >= ? and rma_id in (select rma_id from rmaitem where rmaitem.status in ('APP', 'MAN')) and rma_id not in (select rma_id from rtnreceipt)
rma	canceled	delete from rma where status = 'CAN' and (days(CURRENT TIMESTAMP) - days(lastupdate)) >= ?
rma	completed	delete from rma where status = 'CLO' and (days(CURRENT TIMESTAMP) - days(lastupdate)) >= ? and rma_id not in (select rma_id from rtnreceipt)
rma	not_approved	delete from rma where status = 'PND' and (days(CURRENT TIMESTAMP) - days(lastupdate)) >= ? and rma_id not in (select rma_id from rmaitem where rmaitem.status in ('APP', 'MAN')) and rma_id not in (select rma_id from rtnreceipt)

rtnreasons	obsolete	delete from rtnreason where markfordelete = 1 and rtnreason_id not in (select distinct rtnreason_id from rtnrcptdsp) and rtnreason_id not in (select distinct rtnreason_id from rmaitem)
sastats	obsolete	delete from sastats
staglog	obsolete	delete from staglog where stgprocessed = 1 and (days(CURRENT TIMESTAMP) - days(stgstmp)) >= ?
store	obsolete	delete from storeent where storeent_id = ? and type='S'
users	guest	delete from member where member_id in (select users_id from users where registertype='G' and (days(CURRENT TIMESTAMP) - days(lastsession)) >= ? And (users_id not in (select member_id from orders where status != 'Q')) and (users_id > 0) and (users_id not in (select member_id from address where address_id in (select address_id from orderitems where address_id is not null and status != 'Q') or address_id in (select allocaddress_id from orderitems where allocaddress_id is not null and status != 'Q') or address_id in (select address_id from orders where address_id is not null and status !='Q'))))

users	registered	delete from member where member_id in (select users_id from users where registertype= 'R' and (days(CURRENT TIMESTAMP) - days(lastsession)) >= ? and (users_id not in (select member_id from orders where status != 'Q')) and (users_id > 0) and (users_id not in (select member_id from address where address_id in (select address_id from orderitems where address_id is not null and status != 'Q') or address_id in (select allocaddress_id from orderitems where allocaddress_id is not null and status != 'Q') or address_id in (select address_id from orders where address_id is not null and status !='Q'))))
usrtraffic	obsolete	delete from usrtraffic where (days(CURRENT TIMESTAMP) - days(stmp)) >= ?
vendor	obsolete	delete from vendor where markfordelete = 1 and vendor_id not in (select distinct vendor_id from ra) and vendor_id not in (select distinct vendor_id from receipt where vendor_id is not null)
product_set	obsolete	delete from productset where markfordelete = 1 and productset_id not in (select productset_id from tradeposcn where productset_id is not null)
product_set	obsolete	delete from productset where productset_id in (select productset_id from tradeposcn where productset_id is not NULL and markfordelete = 1 and type = 'C')
tradeposcn	obsolete	delete from tradeposcn where markfordelete = 1 and type = 'S'

Database Cleanup utility command (Windows NT, Windows 2000, AIX, and Solaris)

The Database Cleanup utility removes objects from your database. To run the Database Cleanup utility, type the following from a command line. Type the entire command on one line.



Oracle

Note:

- You must include the optional parameters, logon user ID, and password in the command even if you are currently running this utility with the same user ID.

Parameter values

`object`

The name of the object from which obsolete records will be deleted. Specify one of the following objects:

- Type `account` to delete account objects.
- Type `address` to delete address objects.
- Type `atp_inventory` to delete receipt information objects.
- Type `attachment` to delete attachment objects.
- Type `auction` to delete auction objects.
- Type `auctionlog` to delete auction log objects.
- Type `autobidlog` to delete autobids objects.
- Type `baseitem` to delete product information objects.
- Type `bidlog` to delete bid log objects.
- Type `cachlog` to delete cached objects.
- Type `calculation_code` to delete calculation code objects.
- Type `catentry` to delete catalog entry objects.
- Type `catalog_group` to delete catalog group objects.
- Type `contract` to delete contract objects.
- Type `cpgnlog` to delete campaign objects.
- Type `cpgnstats` to delete campaign statistic objects.
- Type `expected_inventory_records` to delete inventory objects.
- Type `expected_inventory_records_details` to inventory detail objects.
- Type `forummsg` to delete message objects between a Site Administrator and customers.

- Type `fulfillment_center` to delete fulfillment center objects.
- Type `inventory_adjustments` to delete inventory objects.
- Type `inventory_adjustment_codes` to delete inventory code objects.
- Type `itemspecification` to delete specified item objects.
- Type `message` to delete auction-related message objects.
- Type `msgmemrel` to delete message member relationship objects.
- Type `orders` to delete order objects.
- Type `organization` to delete organization objects.
- Type `pastats` to delete Product Advisor statistic objects.
- Type `pcstats` to delete Product Comparison statistic objects.
- Type `pestats` to delete Product Explorer statistic objects.
- Type `policy` to delete policy objects.
- Type `product_sets` to delete product set objects.
- Type `rfq` to delete request for quote objects.
- Type `rma` to delete returned item objects.
- Type `rtnreasons` to delete rtnreason objects.
- Type `sastats` to delete Sales Assistant statistic objects.
- Type `staglog` to delete staged objects.
- Type `store` to delete store objects.
- Type `user` to delete user objects.
- Type `usrtraffic` to delete user traffic log objects.
- Type `vendor` to delete vendor objects.

type The type of object you want to delete. Refer to the individual commands under cleaning the database.

days (Optional) The minimum days in existence for a record to be deleted.

name (Optional) The ID of the object to be deleted. This parameter is required if `member` was indicated as the value for the `table` parameter and `organization` was indicated as the `type` value.

dbtype
(Optional) The database type is DB2 or Oracle. The default is DB2.

db The name of the database.

Oracle

Use `host:port:sid`. For example, `myhost:1521:mydb`.

dbuser
(Optional) The logon ID of the administrator who has created the schema or Site Administrator of the database. If this parameter is not specified, the ID of the user invoking the utility is used.

dbpasswd
(Optional) The password of the logon ID that is specified by the `dbuser` parameter for the DB2 database. If not specified, the system prompts you to enter the password.

check_object_only
(Optional) The Database Cleanup utility lists all child tables, which might be impacted by the database cleanup, and delete restricts if you specify `yes`. The utility does not perform a check if you leave the parameter to `no` (the default).

force (Optional) The force option can be set to `yes` or `no`. If you specify `yes`, the utility deletes the child tables, followed by the parent table.

loglevel

(Optional) The level of logging to be performed during the database cleanup. If no value is specified, the default logging level is `0`.

- Type `0` to specify that no log activities are recorded. This is the default.
- Type `1` to specify that DELETE statements are to be recorded only for the table specified.
- Type `2` to specify that DELETE statements are to be recorded for the table specified, and for any child tables. Loglevel 2 forces the Database Cleanup utility to use the bottom-up method.

log (Optional) The path and name of the log file in which the utility records its activities. The issuer of this command must have write authority to the specified path and the path must already exist. If this parameter is not specified, a log file called `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` is created in the following log directory.

NT

drive:\WebSphere\CommerceServer\logs

2000

drive:\Program Files\WebSphere\CommerceServer\logs

AIX

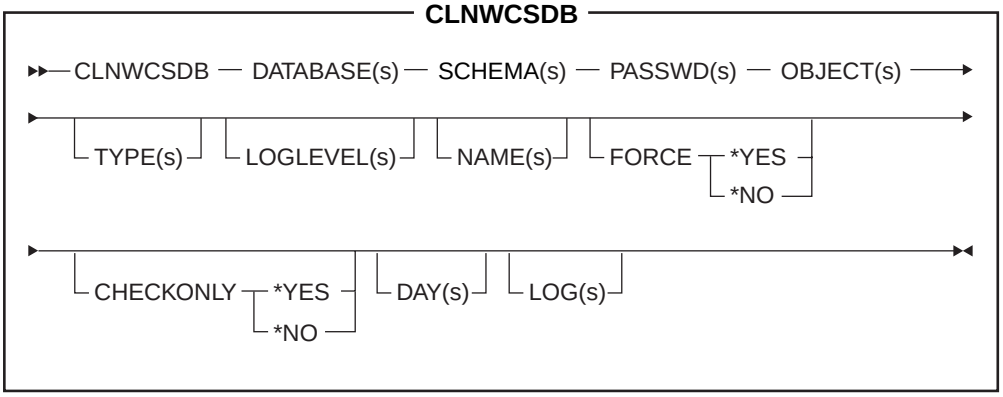
/usr/WebSphere/AppServer/logs

Solaris

/opt/WebSphere/AppServer/logs

Database Cleanup utility command (OS/400 for iSeries)

The Database Cleanup utility removes objects from your database. To run the Database Cleanup utility, type the following from a command line. Type the entire command on one line.



Parameter values

DATABASE

The name of the database. You can find the name in the relational database directory.

SCHEMA

The name of the database schema, or collection. This is the same as the instance logon ID.

PASSWD

The password of the logon ID that is specified by the SCHEMA parameter. If not specified, the system prompts you to enter the password.

OBJECT

The name of the object from which obsolete records will be deleted. Use single quotation marks (’). Specify one of the following object names:

- Type account to delete account objects.
- Type address to delete address objects.
- Type atp_inventory to delete receipt information objects.
- Type attachment to delete attachment objects.
- Type auction to delete auction objects.
- Type auctionlog to delete auction log objects.
- Type autobidlog to delete autobids objects.
- Type baseitem to delete product information objects.
- Type bidlog to delete bid log objects.
- Type cachlog to delete cached objects.
- Type calculation_code to delete calculation code objects.
- Type catentry to delete catalog entry objects.
- Type catalog_group to delete catalog group objects.
- Type contract to delete contract objects.
- Type cpgnlog to delete campaign objects.
- Type cpgnstats to delete campaign statistic objects.
- Type expected_inventory_records to delete inventory objects.
- Type expected_inventory_records_details to inventory detail objects.
- Type forummsg to delete message objects between a Site Administrator and customers.
- Type fulfillment_center to delete fulfillment center objects.
- Type inventory_adjustment to delete inventory objects.
- Type inventory_adjustment_codes to delete inventory code objects.
- Type itemspecification to delete specified item objects.
- Type message to delete auction-related message objects.
- Type msgmemrel to delete message member relationship objects.
- Type orders to delete order objects.
- Type organization to delete organization objects.
- Type pastats to delete Product Advisor statistic objects.
- Type pcstats to delete Product Comparison statistic objects.
- Type pestats to delete Product Explorer statistic objects.
- Type policy to delete policy objects.
- Type product_sets to delete product set objects.
- Type rfq to delete request for quote objects.
- Type rma to delete returned item objects.
- Type rtnreason to delete rtnreason objects.
- Type sastats to delete Sales Assistant statistic objects.

- Type `staglog` to delete staged objects.
- Type `store` to delete store objects.
- Type `user` to delete user objects.
- Type `usrtraffic` to delete user traffic log objects.
- Type `vendor` to delete vendor objects.

TYPE (Optional) The type of database object you want to delete. Use single quotation marks (’). Refer to the individual commands under clean up the database.

LOGLEVEL
(Optional) The level of logging to be performed during the database cleanup. If no value is specified, the default logging level is 0.

- Type `0` to specify that no log activities are recorded. This is the default.
- Type `1` to specify that DELETE statements are to be recorded only for the table specified.
- Type `2` to specify that DELETE statements are to be recorded for the table specified, and for any child tables. Loglevel 2 forces the Database Cleanup utility to use the bottom-up method.

NAME (Optional) The ID of the object to be deleted. This parameter is required if `member` was indicated as the value for the `table` parameter and `organization` was indicated as the `type` value. Use single quotation marks (’).

FORCE (Optional) The force option can be set to *YES or *NO (the default).

CHECKONLY
(Optional) The Database Cleanup utility checks all child tables and delete restricts. Use the following values: *YES or *NO (the default).

DAYS (Optional) The minimum days in existence for a record to be deleted.

LOG (Optional) The path and name of the log file in which the utility records its activities. The issuer of this command must have write authority to the specified path and the path must already exist. If this parameter is not specified, a log file called `dbclean_`*`SCHEMA name`*`_yyyy.mm.dd_hh.mm.ss.zzz.log` is created in the `QIBM/UserData/WebCommerce/instances/` directory.

Examples of deleting objects

The following are examples of deleting objects from the database tables using optional parameters from the Database Cleanup utility command. Refer the Database Cleanup utility command for detailed parameter information.

Example 1
To verify which tables specify the delete restrict parameter, type the following:

•

Windows

DB2

```
dbclean -object objectname -type typename -db database -check_object_only
yes
```

- Windows

  Oracle

  ```
  dbclean -object objectname -type typename -db host:port:sid
  -check_object_only yes -dbtype oracle -dbuser user -dbpasswd <password>
  ```

- AIX

  Solaris

  DB2

  ```
  . dbclean.sh -object objectname -type typename -db database
  -check_object_only yes
  ```

- AIX

  Solaris

  Oracle

  ```
  . dbclean.sh -object objectname -type typename -db host:port:sid
  -check_object_only yes -dbtype oracle -dbuser user -dbpasswd password
  ```

- 400

  DB2

  ```
  CLNWCSDB DATABASE(dbname)
  SCHEMA(schema_name) PASSWD(instance_password) OBJECT(objectname)
  TYPE(typename) CHECKONLY(*YES)
  ```

Example 2
To use the force option on tables which specified the delete restrict parameter, type the following:

- Windows

  DB2

  ```
  dbclean -object objectname -type typename -db database -days daysold
  -loglevel loglevel -force yes
  ```

- Windows

  Oracle

  ```
  dbclean -object objectname -type typename -db host:port:sid -days daysold
  -loglevel loglevel -force yes -dbtype oracle -dbuser user -dbpasswd
  <password>
  ```

- AIX
  Solaris
  DB2
  ```
  . dbclean.sh -object objectname -type typename -db database -days daysold
  -loglevel loglevel -force yes
  ```
- AIX
  Solaris
  Oracle
  ```
  . dbclean.sh -object objectname -type typename -db host:port:sid -days
  daysold -loglevel loglevel -force yes -dbtype oracle -dbuser user
  -dbpasswd <password>
  ```
- 400
  DB2
  ```
  CLNWCSDB DATABASE(dbname)
  SCHEMA(schema_name) PASSWD(instance_password) OBJECT(objectname)
  TYPE(typename)  FORCE(*YES) DAYS(daysold) LOGLEVEL(loglevel)
  ```

Example 3
The default log file name is always a variation of `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log`.To specify the log file name, type the following:

- Windows
  DB2
  ```
  dbclean -object objectname -type typename -db database -days daysold
  -loglevel loglevel -log logfilename
  ```
- Windows
  Oracle
  ```
  dbclean -object objectname -type typename -db host:port:sid -days daysold
  -loglevel loglevel -log logfilename -dbtype oracle -dbuser user -dbpasswd
  <password>
  ```
- AIX
  Solaris

DB2

```
. dbclean.sh -object objectname -type typename -db database -days daysold
-loglevel loglevel -log logfilename
```

- AIX

  Solaris

  Oracle

  ```
  . dbclean.sh -object objectname -type typename -db host:port:sid -days
  daysold -loglevel loglevel -log logfilename -dbtype oracle -dbuser user
  -dbpasswd <password>
  ```

- 400

  DB2

  ```
  CLNWCSDB DATABASE(dbname)
  SCHEMA(schema_name) PASSWD(instance_password) OBJECT(objectname)
  TYPE(typename)  DAYS(daysold) LOGLEVEL(loglevel) LOG(logfilename)
  ```

Chapter 6. Administering the staging server

Staging server components

Most online stores operate 24 hours a day, 365 days of the year, making it difficult to perform maintenance or test changes to the system. The WebSphere Commerce staging server allows Site Administrators to copy their production database to a staging database in order to test updates without affecting customers. This is useful for testing updates to the product catalog, but it is also important for testing new shopping process commands.

The staging server consists of the following components:

A WebSphere Commerce instance
Tests and modifies your data.

Database schema scripts
Creates the staging tables and triggers for the staging database. The staging database contains the same schema and tables as the production database, plus a set of tables used for staging purposes and a set of triggers to log changes made in the staging database. The staging tables are not used by production. The staging database schema scripts add triggers to the database.

Changes are logged to the STAGLOG table (a staging table) using database triggers. Whenever you change a database table record in the staging database, the STAGLOG table records this change.

The Stage Copy utility
Allows an administrator to copy data from the production database to the staging database. You can copy the data into site-related tables, merchant-related tables, or individual tables.

The Stage Propagate utility
Allows an administrator to propagate changes from the staging database to the production database. The information in the STAGLOG table identifies the records in the staging database that must be inserted, updated, or deleted in the production database. The identified records are then updated in the production database. Processed records are indicated in the STAGLOG table by a 1 in the STGPROCESSED column.

The Stage Check utility
Allows an administrator to check for potential unique index key conflicts between two tables on a staging server and a production server.

Staging server

When you are ready to copy data from the production server to the staging server, use the Stage Copy utility. Tables that can be copied are listed in staging tables. Any new tables that you have created to store custom data can also be copied to the staging server after configuring the staging server for customized tables.

The STAGLOG table acts as an internal log. Whenever you change a record in a table on the staging server, a trigger records this change in the STAGLOG table. For each modified record, a trigger records the type of modification (insert, delete, or update), the name of the table where the record resides, and the record's

primary key or unique index. After changing and testing database records on the staging server, propagate the changes back to the production server using the Stage Propagate utility.

The database tables covered by the staging server should never be updated on the production database during a staging session. A staging session begins when you use the Stage Copy utility to copy the production database to the staging database. A staging session ends when you use the Stage Copy utility to begin another staging session. After the Stage Copy utility copies the production database to the staging database, the staging database and the production database are synchronized in terms of the tables covered by the staging server. Once the tables are synchronized, changes to the tables on the production database are not permitted. You can only update the staging database, and then use the Stage Propagate utility to propagate the changes to the production database. If you update both databases, your propagation will likely fail due to a potential key conflict or a reference integrity violation. If you must update the production database during a staging session, use the Stage Copy utility to synchronize your databases and begin a new staging session.

To guarantee that the tables are never updated on the production database during a staging session, the tables must be under the control of a Site Administrator only. In some cases, the staging tables in your production database are updated by an individual customer or merchant after your staging copy. For example, you cannot prohibit a merchant from modifying the OFFER table in the production database after the staging copy. In this situation, you cannot use the staging server. However, RFQ objects are an exception. When you create RFQ objects on the production database, rows are inserted into trading tables in the production database. If you are creating contracts in the staging database, you are also inserting rows into the trading tables in the staging database. In this case, you are updating the same tables on both the staging database and the production database.

Note that there are some limitations for the staging server when using RFQ objects. You also need to run the Stage Check utility to find potential unique index key conflicts and correct them before running the Stage Propagate utility to propagate the changes to production database.

In a typical business-to-consumer site, tables can be divided into two groups: configuration data and operation data. The configuration tables contain data such as stores, catalogs, catalog entries, languages, taxes, and discounts. These tables are under Site Administrator control; an individual customer cannot modify the tables. The operation tables contain data such as customer information, address, orders, and SET-related data. Customers can modify operation tables. The staging server only covers configuration tables. Refer to WebSphere Commerce staging tables for a list of tables covered by the staging server.

It is also important to ensure that tables covered by the staging server do not contain any foreign key references to operation tables. Otherwise, the propagation could fail due to a potential primary key deletion from the production database. Before you use the staging server, you should ensure that only the organization owns the operational data, and not the individual user, such as a catalog administrator.

You should be aware of the following before using the staging server:

- Any new image files, HTML files, or JSP files that are referenced by the staged records must be manually copied from the staging server to the production server.
- The staging server cannot copy and propagate database schema changes, image files, HTML files or JSP files. For example, if you create a new index or table in a staging database, you must manually create the index or table in production database.
- The Stage Propagate utility cannot propagate records loaded by the Loader package (load mode) or the DB2 Load utility since both bypass the staging triggers. If you have used either utility, use the Stage Copy to resynchronize your database tables and begin a new staging session. You should never use the Loader package (load mode) or DB2 Load on a staging database or a production database during a staging session.
- After using the Stage Copy utility, you must stop and restart the staging server.
- The staging server does not support DB2 Text Extender.
- Do not run the Database Cleanup utility on the staging server except to clean the STAGLOG table.
- There are certain staging server limitations. Ensure that you understand the staging server limitations before using the staging server.

Stage Copy utility (Windows NT, Windows 2000, AIX, and Solaris)

The Stage Copy utility copies data from the production database to the staging database. You can copy data in site-related tables, merchant-related tables, or individual tables. You can also clean the staging database before the Stage Copy utility by using the `cleanup_stage_db` parameter in the command syntax. If you specify `yes`, the Stage Copy utility cleans all staging tables before copying the data. This may impact other tables with the delete cascade. If you specify `no`, the Stage Copy utility will not delete anything from the staging tables. Your copy might fail if your copy data generates a conflict or duplicates a key on the primary key or unique indexes. To use the Stage Copy utility to only clean your staging database, specify `only`.

The Stage Copy utility and the Stage Propagate utility divide database data into two scope levels: site-related and merchant-related. The site scope includes data common to all merchants in the system. For example, the language and country or region code used by the system. The merchant scope includes data related to individual merchants. For example, store information is customized for individual merchants, and rows from the store tables could be specific for each merchant. Certain database tables contain both site and merchant information. If you specify the `scope` parameter to `_all_` during the Stage Copy utility, the site data will be copied, followed by all merchant data. If you specify the scope to `_site_`, only the site data will be copied. If you specify the scope to `_merchant_`, only the merchant data will be copied. Note that you cannot copy data for an individual merchant, only all merchants. If you do not set the scope to `_all_`, copy the site data before the merchant data since the site data is used by all merchants. Otherwise, your copy will fail due to a mismatch between the foreign key and the primary key. When you use the `cleanup_stage_db` to clean the site data, note that the merchant data can be deleted because of the delete cascade. You should clean the merchant data followed by the site data, and then copy the site data followed by the merchant data if you do not set the scope to `_all_`.

Another option with the Stage Copy utility is `script_file` parameter. By specifying a script file name, the Stage Copy utility generates an SQL script file which uses export and import to copy the production database to the staging database based on the specified scope. Delete statements are also generated to

clean the staging database if you use the default value or specify cleanup_stage_db to yes. The script file is located in the directory where you start your Stage Copy utility. The script file speeds your database copy process using export and import. You can also modify the behavior of the Stage Copy utility by changing the generated script file. For example, you can change the script file to use the DB2 load utility instead of the import utility, which will further speed up the copy process. Note that the generated script exports all tables to the directory where you run the utility. Ensure that you have enough disk space.

It is important to understand the transaction scope. When cleaning the staging database, the Stage Copy utility commits the transaction after cleaning each table. When copying the data, the Stage Copy utility commits the transaction after copying each table and synchronizing the KEYS table. For generated scripts, the transaction scope is slightly different because of the DB2 import utility. The DB2 import utility automatically commits the transaction after finishing the import. The transaction is committed before synchronizing the KEYS table. Consequently, synchronizing the KEYS table is done in a separate transaction.

You can specify a table to clean or copy using the dbtable parameter. Note that when you specify which table to clean or copy, the table may not be isolated. Certain tables are related to each other by referential constraints. If you clean a specified table, you will also clean the child tables by the delete cascade. If you copy a specified table, you should first copy the parent table. Otherwise, your clean or copy will fail.

The Stage Copy utility is configurable and extensible. To handle your customized tables, there are some conditions your tables must meet and you must set up in the staging configuration tables. For details, refer to configuring the staging server for customized tables. Before you can use the Stage Copy utility, you must follow the steps in configuring the database. The Stage Copy utility deletes all records from STAGLOG table if the command is successful.

Note: You cannot use the Stage Copy command if RFQs are on your system. For more information, refer to staging server limitations.

Stage Copy utility (CPYWCSSTG) (OS/400 for iSeries)

The Stage Copy utility, or the CPYWCSSTG command, copies data from the production database to the staging database. You can copy data in site-related tables, merchant-related tables, or individual tables. You can also clean the staging database before the Stage Copy utility by using the CLEANUPparameter. If you specify *YES, the Stage Copy utility cleans all staging tables before the copy. This may impact other tables with the delete cascade. If you specify *NO, the Stage Copy utility will not delete anything from the staging tables. Your copy might fail if your copy data generates a conflict or duplicates a key on the primary key or unique indexes. To use the Stage Copy utility to clean your staging database, specify *ONLY.

The Stage Copy utility and the Stage Propagate utility divide database data into two scope levels: site-related and merchant-related. The site scope includes data common to all merchants in the system. For example, the language and country or region code used by the system. The merchant scope includes data related to individual merchants. For example, store information is customized for individual merchants, and rows from the store tables could be specific for each merchant. Certain database tables contain both site and merchant information. If you specify the SCOPE parameter to _all_ during the Stage Copy utility, the site data will be copied, followed by all merchant data. If you specify the scope to _site_, only the

site data will be copied. If you specify the scope to _merchant_, only the merchant data will be copied. Note that you cannot copy data for an individual merchant, only all merchants. If you do not set the scope to _all_, copy the site data before the merchant data since the site data is used by all merchants. Otherwise, your copy will fail due to a mismatch between the foreign key and the primary key. When you use the CLEANUP to clean the site data, note that the merchant data can be deleted because of the delete cascade. You should clean the merchant data followed by the site data, and then copy the site data followed by the merchant data if you do not set the scope to _all_.

It is important to understand the transaction scope. When cleaning the staging database, the Stage Copy utility commits the transaction after cleaning each table. When copying the data, the Stage Copy utility commits the transaction after copying each table and synchronizing the KEYS table.

You can specify a table to clean or copy using the DBTABLE parameter. Note that when you specify which table to clean or copy, the table may not be isolated. Certain tables are related to each other by referential constraints. If you clean a specified table, you will also clean the child tables by the delete cascade. If you copy a specified table, you should first copy the parent table. Otherwise, your clean or copy will fail.

The Stage Copy utility is configurable and extensible. To handle your customized tables, there are some conditions your tables must meet and you must set up in the staging configuration tables. For details, refer to configuring the staging server for customized tables. Before you can use the Stage Copy utility, you must follow the steps in configuring the database. The Stage Copy utility deletes all records from STAGLOG table if the command is successful.

Note: You cannot use the Stage Copy command if RFQs are on your system. For more information, refer to staging server limitations.

Stage Check utility (Windows NT, Windows 2000, AIX, and Solaris)

When the configuration and operation data share the same table, a unique index key conflict might occur between the staging database and the production database. Before propagating your changes to the production database, use the Stage Check command to determine any potential unique index conflicts and correct the conflicts before propagation.

When you are using RFQs on your production database and creating contracts on your staging database, you are updating the same tables on both databases. For example, a Site Administrator creates a contract on a staging database, which will insert one row in the TRADING table (and other tables) on the staging database. At the same time, a user creates an RFQ on the production database, which will insert one row in TRADING table (and other tables) on the production database. The two rows might have the same unique index value in the TRADING table. When propagating the contract from the staging database to the production database, a unique index key conflict is generated, and the propagation will fail. Before propagating, use the Stage Check utility to find the conflicted unique index key and correct them. After that, you can propagate the changes.

When you use the Stage Check utility, specify the -scope parameter as _unique_index_ to check the potential key conflicts for the delta changes in the staging database. For all insert and update operations, it will check the potential index key conflict for all tables specified in the STGUINDTAB table. For each table,

it will go over all unique indexes, and check if there is potential key conflict between production database and staging database. If there are potential key conflicts, it will report the table name, unique index, and conflicted key value.

The Stage Check command does not change your database; it reports the potential key conflict, which must be resolved. When using this command, specify the `-sourcedb` parameter as the staging database. The Stage Check command does not function properly if you specify the production database as your source database.

The staging check utility is configurable and extensible. You can add more tables or your customized tables into STGUINDTAB table and run the Stage Check command to verify any potential key conflicts.

Note: Always ensure that your configuration and operation data do not share the same table.

Stage Check utility (CHKWCSSTG) (OS/400 for iSeries)

When the configuration and operation data share the same table, a unique index key conflict might occur between the staging database and the production database. Before propagating your changes to the production database, use the Stage Check command to determine any potential unique index conflicts and correct the conflicts before propagation.

For example, a Site Administrator creates a contract on a staging database, which will insert one row in the TRADING table (and other tables) on the staging database. At the same time, a user creates an RFQ on the production database, which will insert one row in TRADING table (and other tables) on the production database. The two rows might have the same value in the TRADING table. When propagating the contract from the staging database to the production database, a unique index key conflict is generated if the value is shared, and the propagation will fail.

When you use the Stage Check utility, specify the `SCOPE` parameter as `_unique_index_` to check the delta changes in the staging database. For all insert and update operations, it will check the potential index key conflict for all tables specified in the STGUINDTAB table. For each table, it will go over all unique indexes, and check if there is potential key conflict between production database and staging database. If there are potential key conflicts, it will report the table name, unique index, and conflicted key value.

The Stage Check command does not change your database; it reports the potential key conflict, which must be resolved. When using this command, specify the `SOURCEDB` parameter as the staging database. The Stage Check command does not function properly if you specify the production database.

The staging check utility is configurable and extensible. You can add more tables or your customized tables into STGUINDTAB table and run the Stage Check command to verify for any potential key conflicts.

Note: Always ensure that your configuration and operation data do not share the same table.

Stage Propagate utility (Windows NT, Windows 2000, AIX, and Solaris)

After using the Stage Copy utility, the tables covered by the staging server are synchronized in your staging database and production database. When you

complete changing and testing the database records on the staging server, check for potential unique index key conflicts and correct them using the Stage Check utility. You are now ready to propagate the changes to the production database.

The Stage Propagate utility moves changes from the staging database to the production database. The Stage Propagate utility uses the STAGLOG table to identify the changed records in the staging database, then updates these records in the production database. Processed records are indicated in the STAGLOG table by a 1 in the STGPROCESSED column.

You can specify the `scope` parameter to select the type of data for propagation. Set to `_site_`, all changed site data is propagated from the staging database to the production database. Set to `_merchant_`, the changed data of all merchants is propagated. You cannot propagate individual merchant data. Set to `_all_`, both site and merchant data is propagated.

Using the `dbtable` parameter, you can propagate a specific table. Ensure that the parent table has been propagated before you specify a table.

The transaction scope for the Stage Propagate utility is different than the Stage Copy utility. Each run of the Stage Propagate utility command counts as one transaction. For example, if you specify the `scope` as `_site_`, the Stage Propagate utility will begin a new transaction for all modified site data and commit the transaction after a successful propagation. If the propagation fails, the propagation rolls back and the state of your production database is the same as before.

The Stage Propagate utility is configurable and extensible. Before propagating your customized tables, the tables must meet certain conditions. For details, refer to configuring the staging server for customized tables. Before you can use the Stage Propagate utility, you must follow the steps in configuring the database.

Stage Propagate utility (PRPWCSSTG) (OS/400 for iSeries)

After using the Stage Copy utility, the tables covered by the staging server are synchronized in your staging database and production database. When you complete changing and testing the database records on the staging server, check for potential unique index key conflicts and correct them using the Stage Check utility. You are now ready to propagate the changes to the production database.

The Stage Propagate utility, or the PRPWCSSTG command, moves changes from the staging database to the production database. The Stage Propagate utility uses the STAGLOG table to identify the changed records in the staging database, then updates these records in the production database. Processed records are indicated in the STAGLOG table by a 1 in the STGPROCESSED column.

You can specify the `SCOPE` parameter to select the type of data for propagation. Set to `_site_`, all changed site data is propagated from the staging database to the production database. Set to `_merchant_`, the changed data of all merchants is propagated. You cannot propagate individual merchant data. Set to `_all_`, both site and merchant data is propagated.

Using the `DBTABLE` parameter, you can propagate a specific table. Ensure that the parent table has been propagated before you specify a table.

The transaction scope for the Stage Propagate utility is different than the Stage Copy utility. Each run of the Stage Propagate utility command counts as one transaction. For example, if you specify the `SCOPE` as `_site_`, the Stage Propagate

utility will begin a new transaction for all modified site data and commit the transaction after a successful propagation. If the propagation fails, the propagation rolls back and the state of your production database is the same as before.

The Stage Propagate utility is configurable and extensible. Before handling your customized tables, they must meet certain conditions. For details, refer to configuring the staging server for customized tables. Before you can use the Stage Propagate utility, you must follow the steps in configuring the database.

Configuring the staging server for customized tables

To use the staging server with your customized database tables, do the following configuration:

1. Identify your customized table scope (site data, merchant data, or site and merchant data).
2. Create the trigger for your database table using the corresponding trigger examples based on your table's scope and index type.
3. Insert the customized tables into the STGSITETAB, STGMERTAB, and STGMRSTTAB tables.
   - For site tables, insert only into STGSITETAB.
   - For merchant tables, insert only into STGMERTAB.
   - For tables which contain both site and merchant data, insert into STGSITETAB, STGMERTAB, and STGMRSTTAB.

Note: You must ensure that all the parent tables have been inserted properly and that the TABNBR column of the parent tables is less than that of the child table. If your customized table is the parent table of a WebSphere Commerce table, you also need to ensure that your table's TABNBR column is less than that of the child tables.

Testing the site on a staging server

To test a site on a staging server, do the following:

1. Configure the staging server.
2. Create triggers for any custom tables.
3. Configure your remote database (if applicable).
4. Copy data to the staging database.
5. Test the site.
6. Run the Stage Check command to ensure there is no unique index key conflict.
7. Propagate data to the production database.
8. Copy the files to the production server.
9. Delete staged objects from the STAGLOG table.

Configuring the staging server

Any WebSphere Commerce machine can be set up as a staging server. A staging server can be configured during or after installation. Setting up a staging server during WebSphere Commerce installation is described in the *WebSphere Commerce Installation Guide*. To set up a staging server after installation, do the following:

1. Create and configure a separate WebSphere Commerce instance to use as a staging server with Configuration Manager.
2. Ensure that you select the **Use Staging Server** checkbox on the **Database** panel to configure the instance as your staging server.
3. Ensure that caching is not enabled in the Configuration Manager **Cache** panel.

Note: You cannot enable caching triggers on the staging database.

Creating triggers for custom tables

A trigger creates an entry in the STAGLOG table which identifies database record changes. You can modify the settings of the existing triggers to new tables if the tables contain the same data scope and key characteristics.

If you have not created any new tables, you do not need to perform this step. If you have created new tables, refer to the instructions in configuring the staging server for customized tables.

Configuring a remote database

If you have setup your staging server on a machine other than your production server, the remote database must be configured. If you plan to run staging utilities from the staging server, you need to configure your production database as the remote database in your staging server. If you plan to run staging utilities from the production server, you need to configure the staging database as the remote database in your production server.

DB2

For a DB2 database, refer to the *DB2 Administration Guide*.

Oracle

For an Oracle database, refer to the product's documentation.

Copying data to the staging database

To copy data from the production database to the staging database, do the following:

1. Set the PATH environment variables.
2. Configure the database.
3. Change to the directory to which you want log files written.
4. Type the following:
   - Windows

     DB2

     ```
     stagingcopy -scope _all_ -sourcedb production_database_name -destdb
     staging_database_name
     ```
   - Windows

     Oracle

     ```
     stagingcopy -scope _all_ -sourcedb production_database_name-
     destdbstaging_database_name -dbtype oracle -sourcedb_useruser
     -sourcedb_passwdpassword-destdb_useruser -destdb_passwdpassword
     ```
   - AIX

     Solaris

DB2

```
. stagingcopy.sh -scope _all_ -sourcedb production_database_name-
destdbstaging_database_name
```

- AIX

  Solaris

  Oracle

  ```
  . stagingcopy.sh -scope _all_ -sourcedb production_database_name-
  destdbstaging_database_name dbtype oracle -sourcedb_useruser
  -sourcedb_passwdpassword-destdb_useruser -destdb_passwdpassword
  ```

- 400

  DB2

  ```
  CPYWCSSTG SOURCEDB(production_database_name)
  SRCINST(production_instance_name) DESTDB(staging_database_name)
  DESTINST(staging_instance_name) SCOPE(_all_)
  ```

  Note: Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

5. Examine the `stagingcopy_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.
6. Stop and restart the staging server instance.

For further information on copying data to the staging database, see the examples.

Running the Stage Check command

To check potential unique index key confliction between the staging database to the production database, do the following:

To run the Stage Check command to ensure there is no unique index key conflict, do the following:

1. Set the PATH environment variables.
2. Configure the database.
3. Change to the directory to which you want log files written.
4. Type the following:

- Windows

  DB2

  ```
  stagingcheck -scope _unique_index_ -sourcedb staging_database_name
  -destdb production_database_name
  ```

- Windows

  Oracle

```
stagingcheck -scope _unique_index_ -sourcedb staging_database_name
-destdb production_database_name -dbtype oracle -sourcedb_user user
-sourcedb_passwd password -destdb_user user -destdb_passwd password
```

- AIX
  Solaris
  DB2

```
. stagingcheck.sh -scope _unique_index_
-sourcedb staging_database_name -destdb production_database_name
```

- AIX
  Solaris
  Oracle

```
. stagingcheck.sh -scope _unique_index_
-sourcedb staging_database_name -destdb production_database_name
dbtype oracle -sourcedb_user user -sourcedb_passwd password
-destdb_user user -destdb_passwd password
```

- 400
  DB2

```
CHKWCSSTG SOURCEDB(staging_database_name)
SRCINST(staging_instance_name) DESTDB(production_database_name)
DESTINST(production_instance_database_name) SCOPE(_unique_index_)
```

Note: Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

5. Examine the `stagingcheck_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

Propagating data to the production database

To propagate data from the staging database to the production database, do the following:

1. Set the PATH environment variables.
2. Configure the database.
3. Change to the directory to which you want log files written.
4. Type the following:

- Windows
  DB2

```
stagingprop -scope _all_ -sourcedb staging_database_name -destdb
<production_database_name>
```

- Windows

  Oracle

  ```
  stagingprop -scope _all_ -sourcedb staging_database_name -destdb
  production_database_name -dbtype oracle -sourcedb_user user
  -sourcedb_passwd password -destdb_user user -destdb_passwd password
  ```

- AIX

  Solaris

  DB2

  ```
  . stagingprop.sh -scope _all_ -sourcedb staging_database_name -destdb
  production_database_name
  ```

- AIX

  Solaris

  Oracle

  ```
  . stagingprop.sh -scope _all_ -sourcedb staging_database_name -destdb
  production_database_name dbtype oracle -sourcedb_user user
  -sourcedb_passwd password -destdb_user user -destdb_passwd password
  ```

- 400

  DB2

  ```
  PRPWCSSTG SOURCEDB(staging_database_name)
  SRCINST(staging_instance_name) DESTDB(production_database_name)
  DESTINST(production_instance_name) SCOPE(_all_)
  ```

  Note: Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

5. Examine the `stagingprop_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For further information on propagating data to the production database, see the examples.

Copying files to the production server

If you add, change, or delete image or HTML files (as opposed to data in the database) on the staging server, you must manually copy these files to the production server.

To copy files to the production server, do the following:

1. Copy the files to the production server in the correct directory:
   a. Create a zipped file that contains all static HTML files, associated image files, and other embedded files.
   This file should contain new, updated, and unchanged files.

b. Transfer the zipped files to the production server.
c. Unzip each zipped file into the corresponding directory in the production server directory structure. If you have moved the HTML files, edit the WebSphere Commerce configuration to point to the files in the new directory.

2. Delete unused directories on the production server.

Deleting staged objects

To delete staged objects, do the following:

1. Set the PATH environment variables.
2. Change to the directory to which you want log files written.
3. Type the following:

- Windows

  DB2

  ```
  dbclean -object staglog -type obsolete -db dbname -days daysold
  -loglevel loglevel
  ```

- Windows

  Oracle

  ```
  dbclean -object staglog -type obsolete -db dbname -days daysold
  -loglevel loglevel -dbtype oracle -dbuser <user> -dbpasswd <password>
  ```

- AIX

  Solaris

  DB2

  ```
  . dbclean.sh -object staglog -type obsolete -db dbname -days
  daysold -loglevel loglevel
  ```

- AIX

  Solaris

  Oracle

  ```
  . dbclean.sh -object staglog -type obsolete -db dbname -days
  daysold -loglevel loglevel -dbtype oracle -dbuser user -dbpasswd
  password
  ```

- 400

  DB2

```
CLNWCSDB DATABASE(dbname) SCHEMA(schema_name)
PASSWD(instance_password) OBJECT('staglog') TYPE('obsolete')
DAYS(daysold) LOGLEVEL(loglevel)
```

Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

4. Examine the `dbclean_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

For additional examples on deleting staged objects, refer to examples of deleting objects.

Staging server limitations

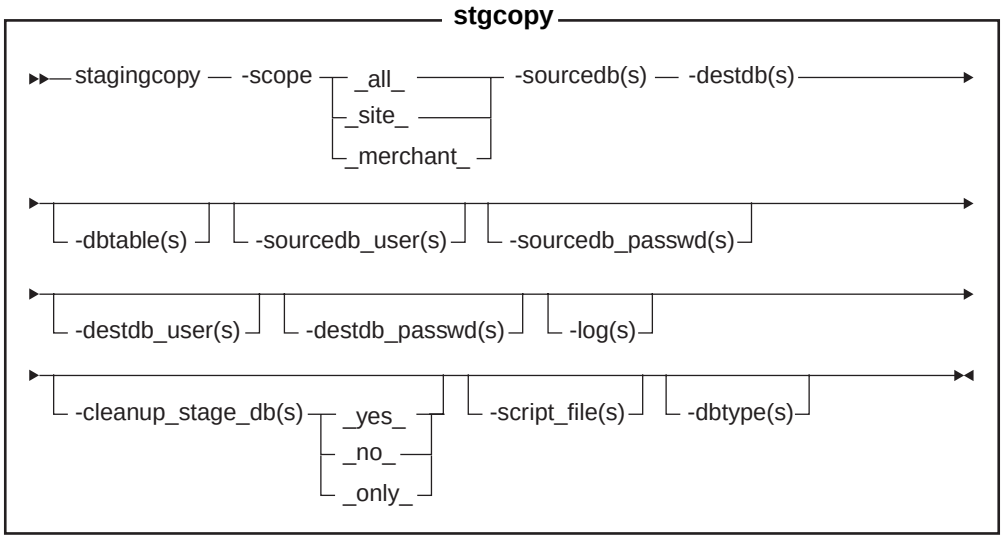
Before using the staging server, you should be aware of the following limitations:

- You cannot use the staging server with the buyer organization self-administration features.
- The `member_id` column of all staging tables (excluding MEMBER, MBRREL, MBRROLE, and MBRATTRVAL) must be organization or member groups, and not the user.
- For all site tables, the `member_id` must be `-2001` or `0`. For all tables containing both site and merchant data, the `member_id` for rows related to site data must be `0` or `-2001`.
- You cannot use the Stage Copy command if you are using RFQ features on your production system. You must use the push mode, for which the Stage Copy command is not available. Before you launch your production site, create the staging server and set up staging database. Deploy and test your data on the staging server, then push to the production server using the Stage Propagate command.
- You cannot create or update RFQs on a staging server.
- You cannot enable caching triggers on the staging database.

Stage Copy utility command (Windows NT, Windows 2000, AIX, and Solaris)

The Stage Copy utility copies data from the production database to the staging database. Note that you cannot use this command if RFQs are on your production system. For details, refer to staging server limitations.

To run the Stage Copy utility, type the following from a command line on a machine that can connect to both the staging server and the production server database. Type the entire command on one line. It is shown here on more than one line for presentation purposes only.



Oracle

Note:

- You must include the optional parameters, logon user ID and password in the command even if you are currently running this utility with the same user ID.

Parameter values

scope The level of scope for the copy to the staging server. Specify one of the following:

- `_all_`
  Type `_all_` to copy both records related to the site and to all merchants.
- `_site_`
  Type `_site_` to copy only site-related records.
- `_merchant_`
  Type `_merchant_` to copy only records related to all merchants.

sourcedb
The name of the database on the production server.

Oracle

Use `host:port:sid`. For example, `myhost:1521:mydb`.

destdb The name of the database on the staging server.

Oracle

Use `host:port:sid`. For example, `myhost:1521:mydb`.

dbtable
(Optional) The name of any specific table to be copied. All records in this table will be copied, provided the records are within the scope specified by the `scope` parameter; otherwise, no records will be copied.

sourcedb_user
(Optional) The logon ID of the database administrator who has created the source database schema. If not specified, the ID of the user currently invoking the utility is used.

sourcedb_passwd
(Optional) The password of the logon ID that is specified by the `sourcedb_user` parameter.

destdb_user
(Optional) The logon ID of the database administrator who has created the destination database schema. If not specified, the ID of the user invoking the utility is used.

destdb_passwd
(Optional) The password of the logon ID that is specified by the `destdb_user` parameter. If not specified, the system prompts you to enter the password.

log (Optional) The path and name of the file in which the Stage Copy utility records its activities and errors. If this parameter is not specified, a log file called `stagingcopy_yyyy.mm.dd_hh.mm.ss.zzz.log` is created in the following log directory.

NT

drive:\WebSphere\CommerceServer\logs

2000

drive:\Program Files\WebSphere\CommerceServer\logs

AIX

/usr/WebSphere/AppServer/logs

Solaris

/opt/WebSphere/AppServer/logs

cleanup_stage_db
(Optional) Use this parameter to clean the staging tables before using the Stage Copy utility. When you use the `-cleanup_stage_db` parameter to clean the site data, note that the merchant data can be deleted because of the delete cascade. You should clean and copy the merchant data after you clean and copy the site data. `Yes` is the default. If you specify `no`, nothing will be deleted from the staging tables. Your copy might fail if your copy data generates conflict or duplicate key on primary key or unique indexes. To use the staging copy to clean up your stage database only, without a data copy from the production database, specify the `-cleanup_stage_db` as `only`.

script file
(Optional) The name of the SQL script file generated by the Stage Copy utility command when using export and import to copy the production database to the staging database on the specified scope. The script file also generates the delete statements to clean the staging database if you use the default value or specify `-cleanup_stage_db` as `yes`. Before you run the script, verify that you have enough disk space to hold the exported tables. The script file is located in the Stage Copy utility directory where you invoke the Stage Copy utility.

DB2

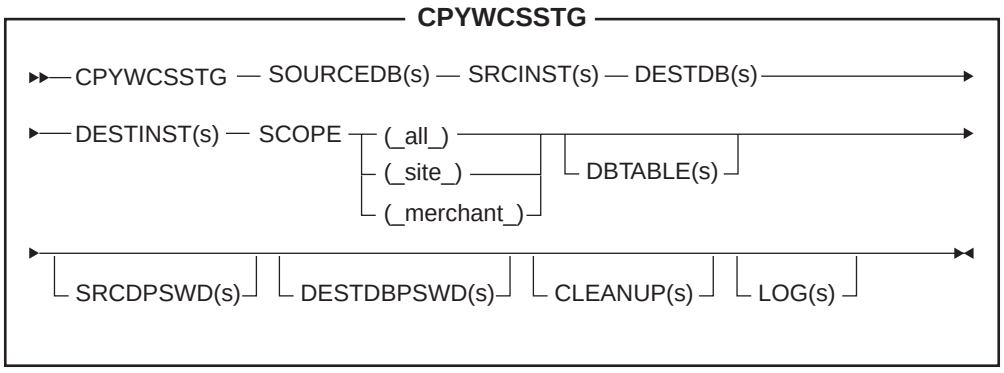
Use `db2 -vtd# -f script_file_name` to run the script file.

dbtype (Optional) The database type (DB2 or Oracle). The default is DB2.

For more information on copying the staging server, see the examples.

CPYWCSSTG command (OS/400 for iSeries)

The Stage Copy utility copies data from the production database to the staging database. To run the Stage Copy utility, type the following from a command line on a machine that can connect to both the staging server and the production server database. Type the entire command on one line. It is shown here on more than one line for presentation purposes only.



Parameter values

SOURCEDB
The name of the database on the production server. You can find this name in the relational database directory.

SRCINST
The source or production instance user ID.

DESTDB The name of the database on the staging server. You can find this name in the relational database directory.

DESTINST
The destination or staging instance user ID.

SCOPE The level of scope for the copy to the staging server. Specify one of the following:

- `_all_`
  Type `_all_` to copy both records related to the site and to all merchants.
- `_site_`
  Type `_site_` to copy only site-related records.
- `_merchant_`
  Type `_merchant_` to copy only records related to all merchants.

DBTABLE
(Optional) The name of any specific table to be copied. All records in this table will be copied, provided the records are within the scope specified by the `scope` parameter; otherwise, no records will be copied.

SRCDBPSWD
(Optional) The password of the logon ID that is specified by the `SRCINST`parameter.

DESTDBPSWD
(Optional) The password of the logon ID that is specified by the DESTINSTparameter. If not specified, the system prompts you to enter the password.

CLEANUP
(Optional) Use this parameter to clean the staging tables before using the Stage Copy utility. When you use the CLEANUP parameter to clean the site data, note that the merchant data can be deleted because of the delete cascade. You should clean and copy the merchant data after you clean and copy the site data. *YES is the default. If you specify *NO, nothing will be deleted from the staging tables. Your copy might fail if your copy data generates conflict or duplicate key on primary key or unique indexes. To use the staging copy to clean up your stage database only, without a data copy from the production database, specify the CLEANUP as *ONLY.

LOG (Optional) The path and name of the file in which the Stage Copy utility records its activities and errors. If this parameter is not specified, a log file called stagingcopy_*SRCINST_DESTINST*_yyyy.mm.dd_hh.mm.ss.zzz.log is created in the QIBM/UserData/WebCommerce/instances/ directory.

For more information on copying the staging server, see the examples.

Examples of copying data to the staging database

The following examples illustrate how you can copy tables from the production database to the staging database. It is important to remember that you cannot use the Stage Copy utility if RFQs are on your production system. For details, refer to staging server limitations.

Note that you should type the entire command in a single line. The commands are shown here on more than one line for presentation purposes only.

Example 1
After cleaning the staging database, copy the production database to the staging database with the scope set to all:

1. Set the PATH environment variables.
2. Configure the database.
3. Change to the directory to which you want log files written.
4. Type the following:
   - Windows

     DB2

     ```
     stagingcopy -scope _all_ -sourcedb production_database_name -destdb
     staging_database_name
     ```
   - Windows

     Oracle

     ```
     stagingcopy -scope _all_ -sourcedb production_database_name -destdb
     staging_database_name-dbtype oracle -sourcedb_user user
     -sourcedb_passwd password -destdb_user user -destdb_passwd password
     ```

- AIX
  Solaris
  DB2

  ```
  . stagingcopy.sh -scope _all_ -sourcedb production_database_name
  -destdb staging_database_name
  ```

- AIX
  Solaris
  Oracle

  ```
  . stagingcopy.sh -scope _all_ -sourcedb production_database_name
  -destdb staging_database_name dbtype oracle -sourcedb_user user
  -sourcedb_passwd password -destdb_user user -destdb_passwd password
  ```

- 400
  DB2

  ```
  CPYWCSSTG SOURCEDB(production_database_name)
  SRCINST(production_instance_ name) DESTDB(staging_database_name)
  DESTINST(staging_instance_name) SCOPE(_all_)
  ```

Note: Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

5. Examine the `stagingcopy_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

To specify the log file name and path, use the log file parameter:

- Windows
  DB2

  ```
  stagingcopy -scope _all_ -sourcedb production_database_name -destdb
  staging_database_name -log log_file_name
  ```

- Windows
  Oracle

  ```
  stagingcopy -scope _all_ -sourcedb production_database_name -destdb
  staging_database_name -log log_file_name -dbtype oracle -sourcedb_user
  user -sourcedb_passwd password -destdb_user user -destdb_passwd password
  ```

- AIX

Solaris

DB2

```
. stagingcopy.sh -scope _all_ -sourcedb production_database_name -destdb
staging_database_name -log log_file_name
```

- AIX

  Solaris

  Oracle

```
. stagingcopy.sh -scope _all_ -sourcedb production_database_name -destdb
staging_database_name -log log_file_name dbtype oracle -sourcedb_user
user -sourcedb_passwd password -destdb_user user -destdb_passwd password
```

- 400

  DB2

```
CPYWCSSTG SOURCEDB(production_database_name) SRCINST(production_instance_
name) DESTDB(staging_database_name) DESTINST(staging_instance_name)
SCOPE(_all_) LOG(log_file_name)
```

Note: Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

If you are using DB2 and are not logged on as the Database Administrator, you need to provide values for the `-sourcedb_user`, `-sourcedb_passwd`, `-destdb_user`, and `-destdb_passwd` options.

Example 2
After cleaning the merchant tables from staging database, copy the merchant-related tables from the production database to staging database:

1. Set the PATH environment variables.
2. Configure the database.
3. Change to the directory to which you want log files written.
4. Type the following:
   - Windows

     DB2

```
stagingcopy -scope _merchant_ -sourcedb production_database_name
-destdb staging_database_name
```

   - Windows

     Oracle

```
stagingcopy -scope _merchant_ -sourcedb production_database_name
-destdb staging_database_name -dbtype oracle -sourcedb_user user
-sourcedb_passwd password -destdb_user user -destdb_passwd password
```

- AIX
  Solaris
  DB2
  ```
  . stagingcopy.sh -scope _merchant_ -sourcedb production_database_name
  -destdb staging_database_name
  ```
- AIX
  Solaris
  Oracle
  ```
  . stagingcopy.sh -scope _merchant_ -sourcedb production_database_name
  -destdb staging_database_name -dbtype oracle -sourcedb_user user
  -sourcedb_passwd password -destdb_user user -destdb_passwd password
  ```
- 400
  DB2
  ```
  CPYWCSSTG SOURCEDB(production_database_name)
  SRCINST(production_instance_ name) DESTDB(staging_database_name)
  DESTINST(staging_instance_name) SCOPE(_merchant_)
  ```

Note: Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

5. Examine the `stagingcopy_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

To only clean the staging database, specify the `-cleanup_stage_db` parameter:

- Windows
  DB2
  ```
  stagingcopy -scope _merchant_ -sourcedb production_database_name -destdb
  staging_database_name -cleanup_stage_db only
  ```
- Windows
  Oracle
  ```
  stagingcopy -scope _merchant_ -sourcedb production_database_name -destdb
  staging_database_name -cleanup_stage_db only -dbtype oracle
  -sourcedb_user user -sourcedb_passwd password -destdb_user user
  -destdb_passwd password
  ```
- AIX

Solaris

DB2

```
. stagingcopy.sh -scope _merchant_ -sourcedb production_database_name
-destdb staging_database_name -cleanup_stage_db only
```

- AIX

  Solaris

  Oracle

  ```
  . stagingcopy.sh -scope _merchant_ -sourcedb production_database_name
  -destdb staging_database_name -cleanup_stage_db only dbtype oracle
  -sourcedb_user user -sourcedb_passwd password -destdb_user user
  -destdb_passwd password
  ```

- 400

  DB2

  ```
  CPYWCSSTG SOURCEDB(production_database_name) SRCINST(production_instance_
  name) DESTDB(staging_database_name) DESTINST(staging_instance_name)
  SCOPE(_merchant_) CLEANUP(*ONLY)
  ```

Note: Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

To only copy data, specify the `-cleanup_stage_db no` parameter:

- Windows

  DB2

  ```
  stagingcopy -scope _merchant_ -sourcedb production_database_name -destdb
  staging_database_name -cleanup_stage_db no
  ```

- Windows

  Oracle

  ```
  stagingcopy -scope _merchant_ -sourcedb production_database_name -destdb
  staging_database_name -cleanup_stage_db no -dbtype oracle -sourcedb_user
  user -sourcedb_passwd password -destdb_user user -destdb_passwd password
  ```

- AIX

  Solaris

  DB2

  ```
  . stagingcopy.sh -scope _merchant_ -sourcedb production_database_name
  -destdb staging_database_name -cleanup_stage_db no
  ```

- AIX

  Solaris

  Oracle

  ```
  . stagingcopy.sh -scope _merchant_ -sourcedb production_database_name
  -destdb staging_database_name -cleanup_stage_db no dbtype oracle
  -sourcedb_user user -sourcedb_passwd password -destdb_user user
  -destdb_passwd password
  ```

- 400

  DB2

  ```
  CPYWCSSTG SOURCEDB(production_database_name) SRCINST(production_instance_
  name) DESTDB(staging_database_name) DESTINST(staging_instance_name)
  SCOPE(_merchant_) CLEANUP(*NO)
  ```

Note: Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

Important: When copying with the scope set to merchant, ensure that you have copied the site scope data first. Otherwise, your copy will fail.

Example 3
After cleaning the site tables from staging database, copy the site tables from production database to stage database.

1. Set the PATH environment variables.
2. Configure the database.
3. Change to the directory to which you want log files written
4. Type the following:
   - Windows

     DB2

     ```
     stagingcopy -scope _site_ -sourcedb production_database_name -destdb
     staging_database_name
     ```

   - Windows

     Oracle

     ```
     stagingcopy -scope _site_ -sourcedb production_database_name -destdb
     staging_database_name -dbtype oracle -sourcedb_user user
     -sourcedb_passwd password -destdb_user user -destdb_passwd password
     ```

   - AIX

     Solaris

DB2

```
. stagingcopy.sh -scope _site_ -sourcedb production_database_name
-destdb staging_database_name
```

- AIX

  Solaris

  Oracle

  ```
  . stagingcopy.sh -scope _site_ -sourcedb production_database_name
  -destdb staging_database_name - dbtype oracle -sourcedb_user user
  -sourcedb_passwd password -destdb_user user -destdb_passwd password
  ```

- 400

  DB2

  ```
  CPYWCSSTG SOURCEDB(production_database_name)
  SRCINST(production_instance_ name) DESTDB(staging_database_name)
  DESTINST(staging_instance_name) SCOPE(_site_)
  ```

Note: Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

5. Examine the `stagingcopy_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

Deleting the site tables may impact the merchant tables due to the delete cascade restriction. Clean the merchant data first, followed by the site data, and then copy the data:

- Windows

  DB2

  ```
  stagingcopy -scope _merchant_ -sourcedb production_database_name -destdb
  staging_database_name -cleanup_stage_db only
  ```

- Windows

  Oracle

  ```
  stagingcopy -scope _merchant_ -sourcedb production_database_name -destdb
  staging_database_name -cleanup_stage_db only -dbtype oracle
  -sourcedb_user user -sourcedb_passwd password -destdb_user user
  -destdb_passwd password
  ```

- AIX

  Solaris

  DB2

```
. stagingcopy.sh -scope _merchant_ -sourcedb production_database_name
-destdb staging_database_name -cleanup_stage_db only
```

- AIX

  Solaris

  Oracle

  ```
  . stagingcopy.sh -scope _merchant_ -sourcedb production_database_name
  -destdb staging_database_name -cleanup_stage_db only dbtype oracle
  -sourcedb_user user -sourcedb_passwd password -destdb_user user
  -destdb_passwd password
  ```

- 400

  DB2

  ```
  CPYWCSSTG SOURCEDB(production_database_name) SRCINST(production_instance_
  name) DESTDB(staging_database_name) DESTINST(staging_instance_name)
  SCOPE(_merchant_) CLEANUP(*ONLY)
  ```

Note: Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

- Windows

  DB2

  ```
  stagingcopy -scope _site_ -sourcedb production_database_name -destdb
  staging_database_name -cleanup_stage_db only
  ```

- Windows

  Oracle

  ```
  stagingcopy -scope _site_ -sourcedb production_database_name -destdb
  staging_database_name -cleanup_stage_db only -dbtype oracle
  -sourcedb_user user -sourcedb_passwd password -destdb_user user
  -destdb_passwd password
  ```

- AIX

  Solaris

  DB2

  ```
  . stagingcopy.sh -scope _site_ -sourcedb production_database_name -destdb
  staging_database_name -cleanup_stage_db only
  ```

- AIX

  Solaris

Oracle

```
. stagingcopy.sh -scope _site_ -sourcedb production_database_name -destdb
staging_database_name -cleanup_stage_db only dbtype oracle -sourcedb_user
user -sourcedb_passwd password -destdb_user user -destdb_passwd password
```

- 400

  DB2

  ```
  CPYWCSSTG SOURCEDB(production_database_name) SRCINST(production_instance_
  name) DESTDB(staging_database_name) DESTINST(staging_instance_name)
  SCOPE(_site_) CLEANUP(*ONLY)
  ```

Note: Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

- Windows

  DB2

  ```
  stagingcopy -scope _site_ -sourcedb production_database_name -destdb
  staging_database_name -cleanup_stage_db no
  ```

- Windows

  Oracle

  ```
  stagingcopy -scope _site_ -sourcedb production_database_name -destdb
  staging_database_name -cleanup_stage_db no -dbtype oracle -sourcedb_user
  user -sourcedb_passwd password -destdb_user user -destdb_passwd password
  ```

- AIX

  Solaris

  DB2

  ```
  . stagingcopy.sh -scope _site_ -sourcedb production_database_name -destdb
  staging_database_name -cleanup_stage_db no
  ```

- AIX

  Solaris

  Oracle

  ```
  . stagingcopy.sh -scope _site_ -sourcedb production_database_name -destdb
  staging_database_name -cleanup_stage_db no dbtype oracle -sourcedb_user
  user -sourcedb_passwd password -destdb_user user -destdb_passwd password
  ```

- 400

DB2

```
CPYWCSSTG SOURCEDB(production_database_name) SRCINST(production_instance_
name) DESTDB(staging_database_name) DESTINST(staging_instance_name)
SCOPE(_site_) CLEANUP(*NO)
```

Note: Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

- Windows

  DB2

  ```
  stagingcopy -scope _merchant_ -sourcedb production_database_name -destdb
  staging_database_name -cleanup_stage_db no
  ```

- Windows

  Oracle

  ```
  stagingcopy -scope _merchant_ -sourcedb production_database_name -destdb
  staging_database_name -cleanup_stage_db no -dbtype oracle -sourcedb_user
  user -sourcedb_passwd password -destdb_user user -destdb_passwd password
  ```

- AIX

  Solaris

  DB2

  ```
  . stagingcopy.sh -scope _merchant_ -sourcedb production_database_name
  -destdb staging_database_name -cleanup_stage_db no
  ```

- AIX

  Solaris

  Oracle

  ```
  . stagingcopy.sh -scope _merchant_ -sourcedb production_database_name
  -destdb staging_database_name -cleanup_stage_db no dbtype oracle
  -sourcedb_user user -sourcedb_passwd password -destdb_user user
  -destdb_passwd password
  ```

- 400

  DB2

  ```
  CPYWCSSTG SOURCEDB(production_database_name) SRCINST(production_instance_
  name) DESTDB(staging_database_name) DESTINST(staging_instance_name)
  SCOPE(_merchant_) CLEANUP(*NO)
  ```

Note: Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

Example 4
Generate the following script to clean and copy the production database to the stage database with scope all.

400

This example does not apply to OS/400 for iSeries.

1. Set the PATH environment variables.
2. Configure the database.
3. Change to the directory to which you want log files written.
4. Type the following:
   - Windows

     DB2

     ```
     stagingcopy -scope _all_ -sourcedb production_database_name -destdb
     staging_database_name -script_file stage_copy.sql
     ```
   - Windows

     Oracle

     ```
     stagingcopy -scope _all_ -sourcedb production_database_name -destdb
     staging_database_name -script_file stage_copy.sql -dbtype oracle
     -sourcedb_user user -sourcedb_passwd password -destdb_user user
     -destdb_passwd password
     ```
   - AIX

     Solaris

     DB2

     ```
     . stagingcopy.sh -scope _all_ -sourcedb production_database_name
     -destdb staging_database_name -script_file stage_copy.sql
     ```
   - AIX

     Solaris

     Oracle

     ```
     . stagingcopy.sh -scope _all_ -sourcedb production_database_name
     -destdb staging_database_name -script_file stage_copy.sql dbtype
     oracle -sourcedb_user user -sourcedb_passwd password -destdb_user user
     -destdb_passwd password
     ```

   Note: Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.
5. Examine the `stagingcopy_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

The Stage Copy utility generates the `stage_copy.sql` script to clean and copy the database.

DB2

If you are using DB2, run the following script:

1. Logon as the Database Administrator (DBA).
2. Open a DB2 command window.
3. Type: `db2 -vtd# -f stage_copy.sql`

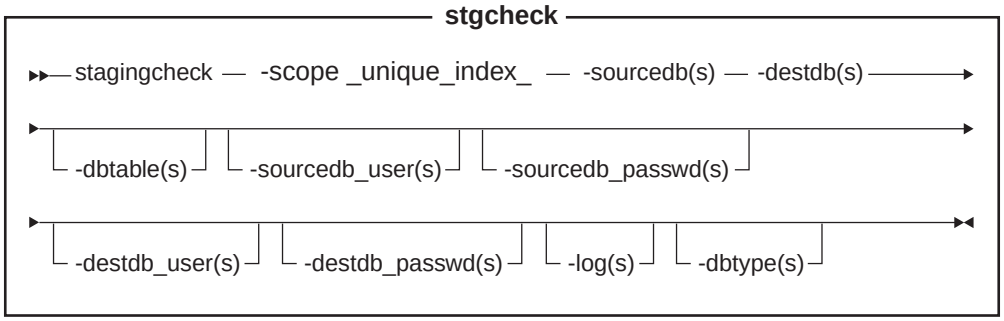
Oracle

If you are using Oracle, run the following script:

1. Open an SQLPlus window.
2. Connect as dba.
3. Type: `@stage_copy.sql`

Stage Check command (Windows NT, Windows 2000, AIX, and Solaris)

The Stage Check command determines if there is a unique index key conflict between the staging database and the production database. To run the Stage Check utility, type the following from a command line on staging server or production server. Type the entire command on one line. It is shown here on more than one line for presentation purpose only.



Oracle

Note:

- You must include the optional parameters, logon user ID and password in the command even if you are currently running this utility with the same user ID.

Parameter values

scope The level of scope for the copy to the staging server. Specify `_unique_index_`.

sourcedb
The name of the database on the staging server.

Oracle

Use `host:port:sid`. For example, `myhost:1521:mydb`.

destdb The name of the database on the production server.

Oracle

Use `host:port:sid`. For example, `myhost:1521:mydb`.

dbtable
(Optional) The name of a specific table to be checked for unique key conflicts.

sourcedb_user
(Optional) The logon ID of the database administrator who has created the schema of the staging database. If not specified, the ID of the user currently invoking the utility is used.

sourcedb_passwd
(Optional) The password of the logon ID that is specified by the `sourcedb` parameter.

destdb_user
(Optional) The logon ID of the database administrator who has created the schema of the production database. If not specified, the ID of the user invoking the utility is used.

destdb_passwd
(Optional) The password of the logon ID that is specified by the `destdb_user` parameter. If not specified, the system prompts you to enter the password.

log (Optional) The path and name of the file in which the Stage Check utility records its activities and errors. If this parameter is not specified, a log file called `stagingcheck_yyyy.mm.dd_hh.mm.ss.zzz.log` is created in the following log directory.

NT

drive`:\WebSphere\CommerceServer\logs`

2000

drive:\Program Files\WebSphere\CommerceServer\logs

AIX

/usr/WebSphere/AppServer/logs

Solaris

/opt/WebSphere/AppServer/logs

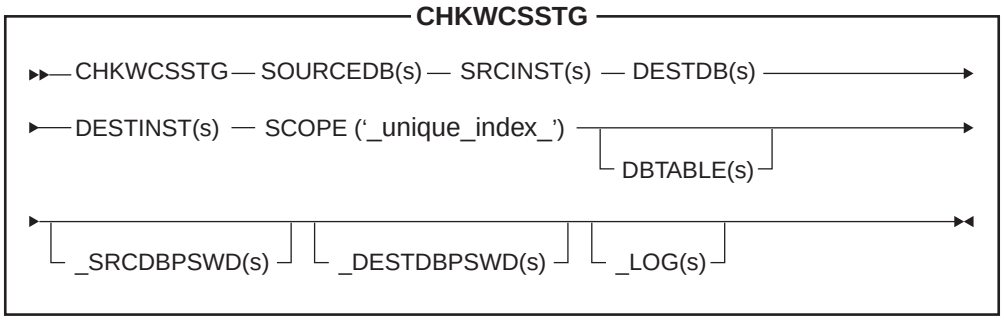
dbtype (Optional) The database type (DB2 or Oracle). The default is DB2.

For more information on checking for unique index key conflicts, see the examples.

CHKWCSSTG command (OS/400 for iSeries)

The Stage Check command determines if there is a unique index key conflict between the staging database and the production database. To run the Stage Check utility, type the following from a command line on staging server or production server. Type the entire command on one line. It is shown here on more than one

line for presentation purpose only.



Parameter values

SOURCEDB
The name of the database on the production server. You can find this name in the relational database directory.

SRCINST
The source or production instance user ID.

DESTDB The name of the database on the staging server. You can find this name in the relational database directory.

DESTINST
The destination or staging instance user ID.

SCOPE The level of scope for the check to the staging server. Specify `_unique_index_`.

DBTABLE
(Optional) The name of a specific table to check for unique key conflicts.

SRCDBPSWD
(Optional) The password of the logon ID that is specified by the `SRCINST`parameter.

DESTDBPSWD
(Optional) The password of the logon ID that is specified by the `DESTINST`parameter. If not specified, the system prompts you to enter the password.

LOG (Optional) The path and name of the file in which the Stage Check utility records its activities and errors. If this parameter is not specified, a log file called `stagingcheck_`*`SRCINST_DESTINST`*`_yyyy.mm.dd_hh.mm.ss.zzz.log` is created in the `QIBM/UserData/WebCommerce/instances/` directory.

For more information on checking for unique index key conflicts, see the examples.

Example for checking the unique index key

The following example illustrates how you can check for unique index key conflicts.

Note that you should type the entire command in a single line. The commands are shown here on more than one line for presentation purposes only.

1. Set the PATH environment variables.
2. Configure the database.
3. Change to the directory to which you want log files written.
4. Type the following:

- Windows

  DB2

  ```
  stagingcheck -scope _unique_index_ -sourcedb staging_database_name
  -destdb production_database_name
  ```

- Windows

  Oracle

  ```
  stagingcheck -scope _unique_index_ -sourcedb staging_database_ name
  -destdb production_database_name-dbtype oracle -sourcedb_user user
  -sourcedb_passwd password -destdb_user user -destdb_passwd password
  ```

- AIX

  Solaris

  DB2

  ```
  . stagingcheck.sh -scope _unique_index_
  -sourcedb staging_database_name -destdb production_database_name
  ```

- AIX

  Solaris

  Oracle

  ```
  . stagingcheck.sh -scope _unique_index_
  -sourcedb staging_database_name -destdb production_database_name
  dbtype oracle -sourcedb_user user -sourcedb_passwd password
  -destdb_user user -destdb_passwd password
  ```

- 400

  DB2

  ```
  CHKWCSSTG SOURCEDB(staging_database_name)
  SRCINST(production_instance_name) DESTDB(staging_database_name)
  DESTINST(production_instance_name) SCOPE(_unique_index_)
  ```

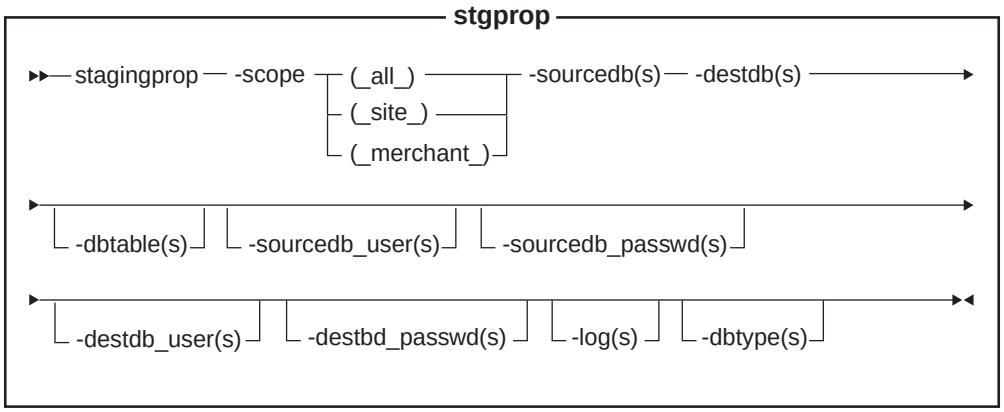
Note: Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

5. Examine the `stagingcheck_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

Stage Propagate utility command (Windows NT, Windows 2000, AIX, and Solaris)

The Stage Propagate utility copies data from the staging database to the production database. Note that you cannot create or update RFQ objects on the staging server.

To run the Stage Propagate utility, type the following from a command line on a machine that can connect to both the staging server and the production server database. Type the entire command on one line. It is shown here on more than one line for presentation purposes only.



Oracle

Note:

- You must include the optional parameters, logon user ID and password in the command even if you are currently running this utility with the same user ID.

Parameter values

scope
The scope level for the propagation to the production server. Specify one of the following:

- `_all_`
  Type `_all_` to propagate both record related to site and to all merchants.
- `_site_`
  Type `_site_` to propagate only site-related record.
- `_merchant_`
  Type `_merchant_` to propagate only records related to all merchants.

sourcedb
The name of the database on the staging server.

Oracle

Use `host:port:sid`. For example, `myhost:1521:mydb`.

destdb The name of the database on the production server.

Oracle

Use `host:port:sid`. For example, `myhost:1521:mydb`.

dbtable
(Optional) The name of any specific table to be propagated. All changed records in this table will be propagated, provided the records are within the scope specified by the `scope` parameter; otherwise, no records will be propagated.

sourcedb_user
(Optional) The logon ID of the database administrator who has created the source database schema. If not specified, the ID of the user currently invoking the utility is used.

sourcedb_passwd
(Optional) The password of the logon ID that is specified by the `sourcedb_user` parameter.

destdb_user
(Optional) The logon ID of the database administrator who has created the destination database schema. If not specified, the ID of the user invoking the utility is used. This parameter is *mandatory* when using a remote database.

destdb_passwd
(Optional) The password of the logon ID that is specified by the `destdb_user` parameter. If not specified, the system prompts you to enter the password. This parameter is mandatory when using a remote database.

log (Optional) The path and name of the file in which the Stage Propagate utility records its activities and errors. If this parameter is not specified, a log file called `stagingprop_yyyy.mm.dd_hh.mm.ss.zzz.log` is created in the following log directory.

NT

drive:\WebSphere\CommerceServer\logs

2000

drive:\Program Files\WebSphere\CommerceServer\logs

AIX

/usr/WebSphere/AppServer/logs

Solaris

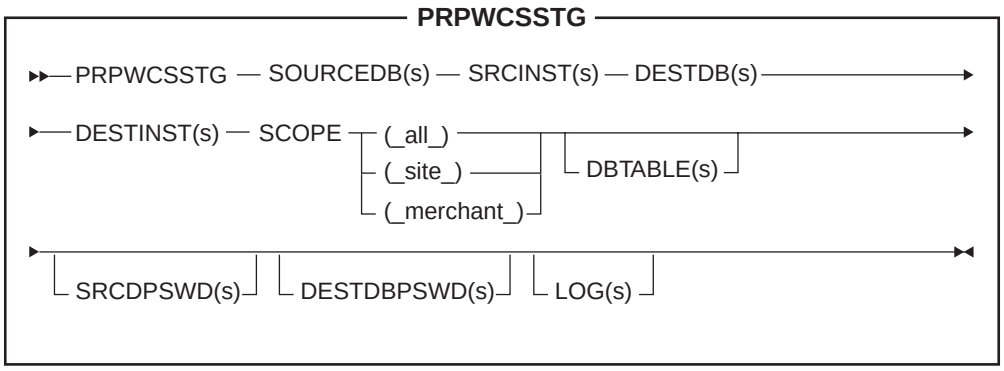
/opt/WebSphere/AppServer/logs

dbtype (Optional) The database type (DB2 or Oracle). The default is DB2.

For more information on propagating to the staging server, see the examples.

PRPWCSSTG command (OS/400 for iSeries)

The Stage Propagate utility copies data from the staging database to the production database. To run the Stage Propagate utility, type the following from a command line on a machine that can connect to both the staging server and the production server database. Type the entire command on one line. It is shown here on more than one line for presentation purposes only.



Parameter values

SOURCEDB
The name of the database on the staging server. You can find this name in the relational database directory.

SRCINST
The source or staging instance logon ID.

DESTDB The name of the database on the production server. You can find this name in the relational database directory.

DESTINST
The destination or production database.

SCOPE
The scope level for the propagation to the production server. Specify one of the following:

- `_all_`
  Type `_all_` to propagate both record related to site and to all merchants.
- `_site_`
  Type `_site_` to propagate only site-related record.
- `_merchant_`
  Type `_merchant_` to propagate only records related to all merchants.

DBTABLE
(Optional) The name of any specific table to be propagated. All changed records in this table will be propagated, provided the records are within the scope specified by the `scope` parameter; otherwise, no records will be propagated.

SRCDBPSWD
(Optional) The password of the logon ID that is specified by the `SRCINST` parameter. If not specified, the system prompts you to enter the password. This parameter is *mandatory* when using a remote database.

DESTDBPSWD
(Optional) The password of the logon ID that is specified by the `DESTINST`parameter. If not specified, the system prompts you to enter the password. This parameter is *mandatory* when using a remote database.

LOG (Optional) The path and name of the file in which the Stage Propagate utility records its activities and errors. If this parameter is not specified, a log file called `stagingprop_`*`SRCINST_DESTINST`*`_yyyy.mm.dd_hh.mm.ss.zzz.log` is created in the `QIBM/UserData/WebCommerce/instances/` directory.

For more information on propagating to the staging server, see the examples.

Examples of propagating data to the production database

The following examples illustrate how you propagate changed records from a staging database to a production database.

Note that you should type the commands in a single line. The commands are shown here on more than one line for presentation purposes only.

Example 1

Propagate all changes from the staging server database to the production database.

1. Set the PATH environment variables.
2. Configure the database.
3. Change to the directory to which you want log files written
4. Type the following:
   - Windows

     DB2

     ```
     stagingprop -scope _all_ -sourcedb staging_database_name -destdb
     production_database_name
     ```
   - Windows

     Oracle

     ```
     stagingprop -scope _all_ -sourcedb staging_database_name-destdb
     production_database_name -dbtype oracle -sourcedb_user user
     -sourcedb_passwd password -destdb_user user -destdb_passwd password
     ```
   - AIX

     Solaris

     DB2

     ```
     . stagingprop.sh -scope _all_ -sourcedb staging_database_name-destdb
     production_database_name
     ```
   - AIX

     Solaris

     Oracle

     ```
     . stagingprop.sh -scope _all_ -sourcedb staging_database_name-destdb
     production_database_name dbtype oracle -sourcedb_user user
     -sourcedb_passwd password -destdb_user user -destdb_passwd password
     ```
   - 400

DB2

```
PRPWCSSTG SOURCEDB(staging_database_name)
SRCINST(staging_instance_name) DESTDB(production_database_name)
DESTINST(production_instance_name) SCOPE(_all_)
```

Note: Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

5. Examine the `stagingprop_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

The Stage Propagate utility will first propagate all site data, and then all merchant data to the production database. If an error occurs, the entire transaction will rollback.

To specify the log file name and path, use the log file parameter:

- Windows

  DB2

  ```
  stagingprop -scope _all_ -sourcedb staging_database_name-destdb
  production_database_name -loglog_file_name
  ```

- Windows

  Oracle

  ```
  stagingprop -scope _all_ -sourcedb staging_database_name-destdb
  production_database_name -loglog_file_name-dbtype oracle -sourcedb_user
  user -sourcedb_passwd password -destdb_user user -destdb_passwd password
  ```

- AIX

  Solaris

  DB2

  ```
  . stagingprop.sh -scope _all_ -sourcedb staging_database_name-destdb
  production_database_name -loglog_file_name
  ```

- AIX

  Solaris

  Oracle

  ```
  . stagingprop.sh -scope _all_ -sourcedb staging_database_name-destdb
  production_database_name -loglog_file_name-dbtype oracle -sourcedb_user
  user -sourcedb_passwd password -destdb_user user -destdb_passwd password
  ```

- 400

DB2

```
PRPWCSSTG SOURCEDB(staging_database_name) SRCINST(staging_instance_name)
DESTDB(production_database_name) DESTINST(production_instance_name)
SCOPE(_all_) LOG(log_file_name)
```

Note: Use `host:port:sid` for the Oracle database name, for example, myhost:1521:mydb.

If you are using DB2 and are not logged on as the Database Administrator, you need to provide values for the `-sourcedb_user`, `-sourcedb_passwd`, `-destdb_user`, and `-destdb_passwd` options.

Example 2

Propagate all modified site data from the staging database to the production database.

1. Set the PATH environment variables.
2. Configure the database.
3. Change to the directory to which you want log files written
4. Type the following:
   - Windows

     DB2

     ```
     stagingprop -scope _site_ -sourcedb staging_database_name-destdb
     production_database_name
     ```
   - Windows

     Oracle

     ```
     stagingprop -scope _site_ -sourcedb staging_database_name-destdb
     production_database_name -dbtype oracle -sourcedb_user user
     -sourcedb_passwd password -destdb_user user -destdb_passwd password
     ```
   - AIX

     Solaris

     DB2

     ```
     . stagingprop.sh -scope _site_ -sourcedb staging_database_name-destdb
     production_database_name
     ```
   - AIX

     Solaris

     Oracle

     ```
     . stagingprop.sh -scope _site_ -sourcedb staging_database_name-destdb
     production_database_name dbtype oracle -sourcedb_user user
     -sourcedb_passwd password -destdb_user user -destdb_passwd password
     ```

- 400

  DB2

  ```
  PRPWCSSTG SOURCEDB(staging_database_name)
  SRCINST(staging_instance_name) DESTDB(production_database_name)
  DESTINST(production_instance_name) SCOPE(_site_)
  ```

Note: Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

5. Examine the `stagingprop_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

Example 3

Propagate all modified merchant data from the staging database to production database (after propagating the site data.)

1. Set the PATH environment variables.
2. Configure the database.
3. Change to the directory to which you want log files written
4. Type the following:
   - Windows

     DB2

     ```
     stagingprop -scope _merchant_ -sourcedb staging_database_name-destdb
     production_database_name
     ```

   - Windows

     Oracle

     ```
     stagingprop -scope _merchant_ -sourcedb staging_database_name-destdb
     production_database_name -dbtype oracle -sourcedb_user user
     -sourcedb_passwd password -destdb_user user -destdb_passwd password
     ```

   - AIX

     Solaris

     DB2

     ```
     . stagingprop.sh -scope _merchant_ -sourcedb staging_database_name-
     destdb production_database_name
     ```

   - AIX

     Solaris

     Oracle

```
. stagingprop.sh -scope _merchant_ -sourcedb staging_database_name-
destdb production_database_name dbtype oracle -sourcedb_user user
-sourcedb_passwd password -destdb_user user -destdb_passwd password
```

- 400

  DB2

  ```
  PRPWCSSTG SOURCEDB(staging_database_name)
  SRCINST(staging_instance_name) DESTDB(production_database_name)
  DESTINST(production_instance_name) SCOPE(_merchant_)
  ```

Note: Use `host:port:sid` for the Oracle database name. For example, `myhost:1521:mydb`.

5. Examine the `stagingprop_yyyy.mm.dd_hh.mm.ss.zzz.log` file to verify that the command was successful.

Customized database table requirements

If you have customized your database schema by creating new tables, you must meet the following requirements to use the staging server:

- **You must define a primary key or a unique index.**
  The staging server functions based on the key. To avoid logging excessive data in the STAGLOG table, log only the key (primary key or unique index). The stage utilities will use the key for compression and to find the data to be propagated. If there is no key, the stage utilities cannot work.
- **A referential integrity (RI) constraint cycle cannot exist among the tables.**
  The staging server always propagates the parent table before the child table. If there is an RI constraint cycle, the staging server cannot distinguish between parent and child tables.
- **The database tables only contain configuration data.**
  In a business-to-consumer scenario, configuration data is under Site Administrator control, such as catalogs and catalog entries. If a table contains operation data, a customer can change the same table in a production database after a Site Administrator has copied the table to the staging database. This causes a potential key conflict or an RI constraint violation.
- **The database tables cannot contain any references to operation tables.**
  The tables to be propagated should not contain any foreign key references to the primary keys of operation tables. If there is such a reference, the data cannot be restored to the product database if a customer deletes the primary key after the stage copy.
- Windows

  AIX

  Solaris

  An insert trigger cannot exist when inserting two tables in the production database.
  For any two tables covered by the staging server (for example, R1 and R2), a trigger to insert rows into R1 or R2 cannot exist when inserting into R2 and R1 in the production database. The insert trigger creates the update in both databases and generates key problems.

- **The MEMBER table cannot have a unique index.**
- **Delete restrict on the customized database tables must be used with caution.** Delete restrict inhibits the Database Cleanup utility's performance. You can also experience difficulties when cleaning the staging database. Before you can clean the staging database, you have to manually use the Database Cleanup utility command with the force option to clean the tables. Otherwise, cleaning the staging database will fail.

To prepare the staging server for customized tables, refer to configuring the staging server for customized tables.

Staging server troubleshooting (Windows NT, Windows 2000, AIX, and Solaris)

1. While using the staging server commands `stgcopy` and `stgprop`, you may receive the following message:

```
SQLSTATE 54001: The statement is too long or too complex.
```

Ensure that you have set the `stmtheap` size of your database as suggested in configuring the database. You should verify that you have enough memory to support the increase.

2. If the Stage Copy utility fails to complete processing, the database log may be too small. From a DB2 command window issue the commands:

```
update database configuration for staging_server using logprimary 50
logfilsiz 1000
db2 terminate
db2stop
db2start
```

where *staging_server* is the name of the staging server database. If the problem persists, try using a higher value for the `logprimary` or `logfilsiz` parameters.

3. AIX

Ensure that the resource for DB2 user, such as db2inst1, has been configured properly. Entering the following command shows your resource limitation:

```
ulimit -a
```

AIX

Configure the data segment to 240MB and stack to 16MB.

For additional information, refer to the *DB2 Command Reference*.

WebSphere Commerce staging tables

You can copy WebSphere Commerce staging tables from the production server to the staging server. The tables are grouped according to whether they contain site-related data, merchant-related data, or both site and merchant-related data.

Each table group is listed at the following pages:

- Site data scope
- Merchant data scope
- Site and merchant data scope

Triggers for staging tables

The following triggers have been defined for the WebSphere Commerce staging tables. You can apply these settings to custom tables if they contain the same data scope and key characteristics. Refer to the links below to view the tables by data scope.

- Site data scope
- Merchant data scope
- Site or merchant data scope

Site data scope

The following chart lists the database tables under the site data scope.

Table name	Unique index on				
	Integer 1	Integer 2	Char 1	Integer 3	Char 2
ACACGPDESC	Yes	Yes			
ACACTACTGP	Yes	Yes			
ACACTDESC	Yes	Yes			
ACACTGRP	Yes				
ACACTION	Yes				
ACATTR	Yes				
ACATTRDESC	Yes	Yes			
ACCCMDTYPE			Yes		
ACRELATION	Yes				
ACRELDESC	Yes	Yes			
ACRESACT	Yes	Yes			
ACRESATREL	Yes	Yes			
ACRESCGRY	Yes				
ACRESGRP	Yes				
ACRESGPDES	Yes	Yes			
ACRESGPRES	Yes	Yes			
ACRESPRIM	Yes				
ACRESREL	Yes	Yes			
ATTACHUSG			Yes		
ATTRTYPE			Yes		
BUYERPOTYP	Yes				
CALUSAGE	Yes				
CATENTTYPE			Yes		
CATRELTYPE			Yes		
CHKARRANG	Yes	Yes			
CHKCMD	Yes				
COUNTCODE			Yes		Yes
COUNTRY	Yes		Yes		
DEVICEFMT	Yes				
FLCOMPOSE	Yes	Yes		Yes	

FLDOMNDESC	Yes	Yes			
FLOW	Yes				
FLOWDESC	Yes	Yes			
FLOWDOMAIN	Yes				
FLOWTYPE	Yes				
FLSTATEDCT	Yes				
FLSTATEGP	Yes				
FLSTATEREL	Yes	Yes			
FLSTDCTDSC	Yes	Yes			
FLSTGPDSC	Yes	Yes			
FLTRANSDSC	Yes	Yes			
FLTRANSITN	Yes				
FLTYPEDESC	Yes	Yes			
ICDATAREG	Yes				
INVRSRVTYP	Yes				
INVRSRVDSC	Yes	Yes			
ITEMTYPE			Yes		
LANGUAGE	Yes				
LANGUAGEDS	Yes	Yes			
MASSOC			Yes		
MASSOCTYPE			Yes		
MBRATTR	Yes				
MBRGRPTYPE	Yes				
OPERATOR	Yes				
OPERATRDSC	Yes	Yes			
ORDCHNLTYP	Yes				
OUTPUTQ	Yes				
OUTPUTQDSC	Yes	Yes			
PARTROLE	Yes				
PARTROLEDS	Yes	Yes			
PLCYTYPDSC	Yes		Yes		
POLICYTYPE	Yes				
QTYCONVERT			Yes		Yes
QTYUNIT			Yes		
QTYUNITDSC	Yes		Yes		
ROLE	Yes				
SCHCMD	Yes				
SETCURR			Yes		
SETCURRDSC	Yes		Yes		
STATECODE			Yes		Yes
STATEPROV	Yes		Yes		
STORECGRY	Yes				

TAXTYPE	Yes				
TCSUBTYPDS	Yes		Yes		
TCSUBTYPE			Yes		
TCTYPE			Yes		
TFALGOPOL	Yes				
TFALGOTYPE	Yes				
TFALGPOLDS	Yes	Yes			
TFALGTYPDS	Yes	Yes			
TFDOMAIN	Yes				
TFDOMDSC	Yes	Yes			
TFSBDOMAIN	Yes				
TFSBDOMDSC	Yes	Yes			
TRDTYPE	Yes				
TRDTYPEDSC	Yes	Yes			
TXCDSCHEME	Yes				

Merchant data scope

The following chart lists the database tables under the merchant data scope.

Table name	**Unique index on**				
	Integer 1	**Integer 2**	**Char 1**	**Integer 3**	**Char 2**
ACCCMDGRP	Yes				
ACCCUSTEXC	Yes	Yes			
ACCOUNT	Yes				
ACORGPOL	Yes	Yes			
ACPOLDESC	Yes	Yes			
ACPOLICY	Yes				
ATTACHMENT	Yes				
ATTRIBUTE	Yes	Yes			
ATTRVALUE	Yes	Yes			
BASEITEM	Yes				
BASEITEM	Yes	Yes			
BUYERPO	Yes				
CALCODE	Yes				
CALCODEDSC	Yes	Yes			
CALCODEMGP	Yes	Yes			
CALCODTXEX	Yes	Yes			
CALMETHOD	Yes				
CALRANGE	Yes				
CALRLOOKUP	Yes				
CALRULE	Yes				
CALRULEMGP	Yes	Yes			

CALSCALE	Yes				
CALSCALEDS	Yes	Yes			
CATALOG	Yes				
CATALOGDSC	Yes	Yes			
CATCNTR	Yes	Yes			
CATCONFINF	Yes				
CATENCALCD	Yes				
CATENTATTR	Yes				
CATENTDESC	Yes	Yes			
CATENTREL	Yes	Yes	Yes		
CATENTRY	Yes				
CATENTSHIP	Yes				
CATGPCALCD	Yes				
CATGPENREL	Yes	Yes		Yes	
CATGROUP	Yes				
CATGRPATTR	Yes	Yes			
CATGRPDESC	Yes	Yes			
CATGRPPS	Yes	Yes		Yes	
CATGRPREL	Yes	Yes		Yes	
CATGRPTPC	Yes	Yes		Yes	
CATTOGRP	Yes	Yes			
CHARGETYPE	Yes				
CHRGTYPDSC	Yes	Yes			
CNTRDISPLY	Yes				
CNTRNAME	Yes		Yes		
CONTRACT	Yes				
CREDITLINE	Yes				
CRULESCALE	Yes	Yes			
CURCONVERT	Yes				
CURCVLIST	Yes	Yes	Yes		Yes
CURFMTDESC	Yes	Yes	Yes		
CURFORMAT	Yes		Yes		
CURLIST	Yes		Yes		
DISPCGPREL	Yes				
DISPENTREL	Yes				
DISTARRANG	Yes				
FLOWADMIN	Yes				
INVADJCODE	Yes				
INVADJDESC	Yes	Yes			
ITEMSPC	Yes				
ITEMVERSN	Yes				
JURST	Yes				

JURSTGPREL	Yes	Yes			
JURSTGROUP	Yes				
LANGPAIR	Yes	Yes		Yes	
LISTPRICE	Yes		Yes		
MASSOCCECE	Yes				
MASSOCGPGP	Yes				
MBRATTRVAL	Yes				
MBRGRPUSG	Yes	Yes			
MBRREL	Yes	Yes			
MBRROLE	Yes	Yes		Yes	
MGPTRDPSCN	Yes	Yes			
OFFER	Yes				
OFFERDESC	Yes	Yes			
OFFERPRICE	Yes		Yes		
PARTICIPNT	Yes				
PATTRDESC	Yes	Yes			
PATTRIBUTE	Yes				
PATTRPROD	Yes	Yes			
PKGATTR	Yes				
PKGATTRVAL	Yes				
PKGITEMREL	Yes	Yes		Yes	
POLICY	Yes				
POLICYCMD	Yes		Yes		
POLICYDESC	Yes	Yes			
POLICYTC	Yes	Yes			
PRODSETDSC	Yes	Yes			
PRODUCTSET	Yes				
PRSETCEREL	Yes	Yes			
QTYFMTDESC	Yes	Yes	Yes		
QTYFORMAT	Yes		Yes		
RTNDNYDESC	Yes	Yes			
RTNDNYRSN	Yes				
RTNDSPCODE	Yes				
RTNDSPDESC	Yes	Yes			
RTNREASON	Yes				
RTNRSNDESC	Yes	Yes			
SHPARJURGP	Yes	Yes			
SHPARRANGE	Yes				
SHPJCRULE	Yes				
SHPMODEDSC	Yes	Yes			
STENCALUSG	Yes	Yes			
STORECAT	Yes	Yes			

STORECENT	Yes	Yes			
STORECGRP	Yes	Yes			
STORECNTR	Yes	Yes			
STOREDEF	Yes				
STOREITEM	Yes	Yes			
STORELANG	Yes	Yes			
STOREMBRGP	Yes	Yes			
STORITMFFC	Yes	Yes		Yes	
STORLANGDS	Yes	Yes		Yes	
TAXCGRY	Yes				
TAXCGRYDS	Yes	Yes			
TAXJCRULE	Yes	Yes		Yes	
TCDESC	Yes	Yes			
TDPSCNCNTR	Yes	Yes			
TERMCOND	Yes				
TRADEPOSCN	Yes				
TRADING	Yes				
TRDATTACH	Yes	Yes			
TRDDESC	Yes	Yes			
TXCDCLASS	Yes				
VENDOR	Yes				
VENDORDESC	Yes	Yes			
VERSIONSPC	Yes				
VIEWREG	Yes	Yes	Yes		

Site and merchant data scope

The following chart lists the database tables under the site and merchant data scope.

Table name	**Unique index on**				
	Integer 1	**Integer 2**	**Char 1**	**Integer 3**	**Char 2**
CMDREG	Yes		Yes		
FFMCENTDS	Yes	Yes			
FFMCENTER	Yes				
MBRGRP	Yes				
MBRGRPCOND	Yes				
MEMBER	Yes				
ORGENTITY	Yes				
SHIPMODE	Yes				
STADDRESS	Yes				
STORE	Yes				
STOREENT	Yes				

STOREENTDS	Yes	Yes			
STOREGRP	Yes				
URLREG	Yes		Yes		
VIEWREG	Yes	Yes	Yes		

Appendix. Learning Guides

Store Administrator Learning Guide

Learning objectives:
The Store Administrator manages the store assets and implements changes to shipping providers as well as store information.

- Learn how to manage store assets by using the Store Services.
- Publish a store
- Manage Blaze rules for a store

Prerequisite skills:
The Store Administrator is Web literate and has a thorough knowledge of business procedures.

Getting started:
From the navigation frame click **Roles -> Technical operations -> Site Administrator**.
You will see a list of applicable online help topics.

You may also want to read: *IBM WebSphere Commerce Business Edition Fundamentals*

Site Administrator Learning Guide

Learning objectives:
Learn how to install, configure, and maintain WebSphere Commerce.

Prerequisite skills:
The Site Administrator has hardware and operating system skills

Getting started:
From the navigation frame click **Roles -> Technical operations -> Site Administrator**.
You will see a list of applicable online help topics.

You may also want to read:

- *WebSphere Commerce Business Edition Installation Guide*
- *IBM WebSphere Commerce Business Edition Fundamentals*

Notices

Any reference to an IBM licensed program in this document is not intended to state or imply that only IBM's licensed program may be used. Any functionally equivalent product, program, or service that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, is the user's responsibility.

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:

Director of Licensing
Intellectual Property & Licensing
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
U.S.A.

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

Lab Director
IBM Canada Ltd. Laboratory
8200 Warden Avenue
Markham, Ontario
L6G 1C7
Canada

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

This document may contain examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples may include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

This document may contain information about other companies' products, including references to such companies' Internet sites. IBM has no responsibility for the accuracy, completeness, or use of such information.

This product is based on the SET protocol.

Note to U.S. Government Users — Documentation related to restricted rights — Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

Trademarks

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

AIX	CICS	DB2
DB2 Extenders	Encina	HotMedia
IBM	iSeries	MQSeries
SecureWay	VisualAge	WebSphere
400		

Blaze Advisor is a trademark of HNC Software, Inc. in the United States, other countries, or both.

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

Lotus and Domino are trademarks of Lotus Development Corporation in the United States, other countries, or both.

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

Oracle is a registered trademark of Oracle Corporation.

SET and the SET Logo are trademarks owned by SET Secure Electronic Transaction LLC. For further information see http://www.setco.org/aboutmark.html.

UNIX is a registered trademark of The Open Group in the United States and other countries.

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

Printed in U.S.A.