

# Oracle Warehouse Builder

## Quick Start Guide

Jean-Pierre Dijcks, Igor Machin,  
March 9, 2004

## What Can You Expect from this Starter Kit?

First and foremost, you can expect a helping hand in navigating through the extensive functionality of Warehouse Builder. After reading this starter kit you should understand which activities you are supposed to do - in which order - to complete a Warehouse Builder project from start to finish.

All chapters in this starter kit are a maximum of 2 pages long. Therefore, this starter kit will focus on the *main* steps you must perform to (for example) install the product. The kit is most definitely not trying to explain every possible installation screen and step.

In general, the overview of activities to complete a Warehouse Builder project is in this specific order:

- Install client and server software
- Install client and server software
- Set up the runtime platform
- Set up the design repository
- Create a Project
- Create the modules (sources and targets)
- Create mappings (data flows to transform and load data)
- Create process flows to manage the data loads
- Generate the code from the design
- Deploy the code to the database
- Execute the process flows to load data

These activities are discussed in the following chapters.

## Installation steps for a new install

This kit assumes that you are installing version 10g of Warehouse Builder.

**Step 1: Installing the software** You can install the product from a CD or from a download. In either case, launch the setup program, which launches the Universal Installer. Then use the following list of actions:

- Install Warehouse Builder in a separate Oracle home
- First install the client software; when this is completed go to the next install
- Now install the server side software (this must be done!)
- If you are installing client and server on the same machine, use the same home
- Otherwise use a new server home for the server side software
- The server side install launches the Runtime Assistant, with which you must create:
  - A runtime repository => the user that will deploy and run your jobs Suggested schema name: RTRepos
  - A runtime access user => a security user to access the runtime repository Suggested schema name: RTAccess
  - At least one target user => your actual data warehouse schema(s) Suggested schema name: RTTarget

## Setting up the Design Repository

You have now successfully installed the software and set up the runtime platform and users. The next step is to create your Design Repository.

- Go to the new Warehouse Builder program group and find the entry named OWB Repository Assistant
- Launch this and create the actual repository that you connect to with the design client software Suggested schema name: OWB10

You are now set up to start working with Warehouse Builder.

## Your First Steps in Warehouse Builder

After starting the design client using the OWB Client entry in the Warehouse Builder program group, you will perform a couple of steps to define your working environment.

### Create a Project

The first thing to do is create a Project in Warehouse Builder. This project holds the metadata you will be defining and importing for your project. The project stores metadata in modules, which you need to create as a second step.

### Create a Set of Modules

The second thing to create is a set of modules in your new project. Do this according to the following rules:

- Create one database/Oracle module for each *target* schema you are expecting to use in your data warehouse.
- Be sure to specify a 'location' while creating each module. Locations are the objects that will be registered to hold the physical schema and machine information.

For example:

- I am creating a module called WH for my warehouse.
  - I will add a location called WH\_LOC to this module, and pick Oracle 10g for database.
  - Later in the process, when registering this WH\_LOC location, I specify the following information for it:
    - Machine name where the data warehouse database is located
    - Database information (Service Name, Port)
    - Schema in which I want my target warehouse objects to reside
- 
- Create one database module for each database *source* schema from which you will be extracting metadata and data. Again, make sure to specify a location for each of these modules. For example:
    - I am creating a module called OE\_SRC that holds my order entry tables
    - I will create a location for this module called OE\_SRC\_LOC
    - When registering this location later I will use:
      - Machine name where the source database is located
      - Database information for the source database (Service name and Port)

- Schema in which these source objects are found
- Create one flat file module for each directory in which you will have source files
- You can ignore the process node for now
- You can ignore the applications node for now (if you have SAP create one of these, too)

### Creating a Runtime Repository Connection

The last step is to connect your design repository (OWB10 user) to the runtime repository so you can deploy objects into your actual warehouse. Find the Runtime Repository Connection node at the bottom of your project navigation tree (expand your project for this). Give the connection a logical name like RTRepos\_Connection.

This allows you to provide multiple runtime systems for the same metadata. So if you want development and production systems, create two of these objects.

Specify the connection information for the runtime repository and the runtime access user in the Runtime Repository Connection.

You are now ready to start working on your project!

## Working with Source Modules

A source module contains the metadata definitions for your source schema objects. This module will have the table and column names for the tables you will be extracting from.

Now that you have a module, you must first import metadata definitions into it. Do the following:

For Oracle modules:

- Use the Project > Import menu item, to invoke the wizard for finding your source tables, views, external tables etc...
- Import these *definitions* (note that you are not importing any *data*) For File modules:
  - Locate your file(s)
  - Sample the file(s) to make the file definition know to Warehouse Builder
  - Import the file(s) (note that you are not importing any *data* even as you sample data to define the file!)

You are now ready to work with sources in Warehouse Builder!

## Working with your Target Modules

Target modules are mainly used to define your warehouse structures. In these you will define the target tables, dimensions, and cubes. The target will also hold the ETL processes in the form of mappings.

Working in a target module involves defining the objects you require. Follow this typical sequence when defining objects:

1. Supporting tables, views, or external tables
2. Dimensions
3. Cubes
4. Mappings (more about these in the next chapter)
5. Materialized views for summary management

Once you have completed your design, make sure to validate the design and let Warehouse Builder take a look at the logical validity. Warehouse Builder will catch things like foreign key inconsistencies and datatype mismatches.

Once you have defined and validated your design, you are set with the target preparation!

## Working with ETL

In Warehouse Builder ETL (extraction, transformation, and load) consists of two main areas:

- Mappings (or data flow diagrams)
- Processes (or process flow diagrams – these can contain multiple mappings)

In this section we will cover mappings, which do the bulk of the work for the ETL process.

### Mapping Concepts

There are several key concepts to understand about mapping:

- Sources and targets => the target object receives data that is extracted from the source
- Operators => are used to design a mapping. These are atomic operations such as a join of several tables
- Groups => are a part of an operator and group together a row set with a single cardinality
- Attributes => are the granular objects that you move data from or into
- Configuration => allows you to change the physical implementation of the generated code
- Operating mode (or Set based vs. Row based) => the way Warehouse Builder processes your data. Set based is fast, row based is slow but controlled
- Audit level => is the level of detail you want to get from tracing the data load
- Debugger => allows you to view test data flowing through the mapping
- Generate => create the code that will be placed on the database
- Deployment => compiles the code and stores it on the database
- Execution => runs the stored packaged procedures you deployed

### Create Mappings

At a high conceptual level, these are the tasks you should perform to create mappings in the target module:

- Know what you want in your target
- Investigate/inspect your source data
- Set up the object grants or create external tables for files to guarantee access for the mapping execution
- Determine the grain of delivered data

- Might require aggregations
- Map out the transformations in your head or on paper
- Start defining the mappings using operators; as a guideline use atomic operations in operators
- Validate the mappings
- Use debugger to verify your data flow for correctness
- Do this at the end when you have a “completed” mapping
- Use test data to test, or when all source data is not available
- Fix your mappings based on debug results to ensure correct transformations
- Validate the mappings again

### Mapping Best Practices

There are a few hints or guidelines to ensure that you create scalable and logical mappings in Warehouse Builder:

- Typically you will create one mapping for one entity on the target schema. For example: one mapping will load into one dimension
- Do not join across remote databases; rather stage both in the target via separate mappings and join in the target
- Use atomic operations in an operator; do not code a lot of logic in a view as you will lose the data lineage and impact analysis that Warehouse Builder provides
- Aim for set based processing => it is faster
- Use correlated commit in multi-target situations
- Aim for explicit error handling => know your data!

After you have completed your mappings, you are ready for the next step in the design process.

## Working with Processes

Process flows are the finalizing component of the ETL process. A process flow ensures that a set of mappings is executed in the correct order and handles exceptions and notifications in an automated way.

### Process Flow Concepts

Some of the things you will need to understand in process flows are:

- Process flow module => holds all process flows (compare to target module holding all target objects)
- Process flow package => executable unit of work in Warehouse Builder
- Process flow => handles dependencies for many activities
- Activity => a single unit of work for the process, for example a mapping or the FTP activity
- Status => return status of an activity; can be success, warning, or error

### Create a Process Flow

A process flow is a chain of events modeled in a diagram. Here the user can consolidate multiple mappings and other external activities in a single processing entity, taking care of the dependencies and the sequentiality as well as the error handling of the process. So the things you conceptually do to create a process flow are:

- Know the dependency and sequence of activities
- Determine what to do for every status for each activity
- Set the statuses and ensure you cannot run into a loop
- Model atomic activities, and model activities such as FTP or FILECHECK in the process, not the mapping
- Model explicit Success, error and if need be warning END activities
- Validate the process flow

### Process Flow Best Practices

Some things to take note of when creating process flows:

- Use process flows to record and process inter-mapping dependencies
- Implement one process flow per load, for example one flow for loading staging tables and one flow for loading the warehouse

## Oracle Warehouse Builder

---

- Implement composite process flows to manage the entire warehouse load
- Use status to route the process flow and beware of SUCCESS / WARNING / ERROR processing
- Beware of AND / OR behavior in a process

If you have designed your process flows you are ready to deploy the system to the various database schemas.

## Generate and Deploy the System

After working on the design for quite a while, you are now ready to place that design onto the database. This process involves two separate logical steps:

- Generate the code
- Deploy the code

Within Warehouse Builder these steps are bundled together, so the generate step is a result of deploying if you use Warehouse Builder's Deployment Manager.

### Deployment Concepts

There is only one concept that you will need to understand and that is what deployment does:

- The deployment step (and 'generate' within it) creates code from the logical models and stores this code in the Oracle database. For objects you get DDL scripts, for mappings you get (mostly) PL/SQL and for process flows you get XPDL
- Apart from storing the code in the database, the same code is also made known in the runtime repository so you can use Warehouse Builder to execute it (see next chapter)
- Locations => the physical place where you will be deploying the code. The logical location from the design will be made physical in deployment
- Registering a location => the process of making a location physical by providing the machine and database information including the physical schema
- Connector => the way one location (or schema) is connected to another location (or schema/directory) owned by the target location. These connectors can become:
  - a. Schema reference (from target schema to source schema)
  - b. Database link (from target schema to source schema)
  - c. Oracle Directory (from target database to source directory)

### Deploying your Objects

To ensure that your deployment step goes well, follow the list below exactly in its order:

- Register all locations first (all targets and all sources)  
Note: if you do not register the source, your code will not deploy!
- Deploy all connectors in all modules
- Within a location, deploy in this sequence:

- Follow your data flow

SRC => STG => WH

Note: There is nothing to deploy for any source location! Note 2: You MUST register the source even if you do not deploy to it.

- Within a module:
  - a. Data objects go first (tables, views, dimensions, cubes, etc...)
  - b. Mapping objects
  - c. Process flows

### Dealing with Errors in Deployment

If you get an error follow these rules: (Note: you have to open up the message you see, the top error is a PL/SQL stack error!)

- Verify your privileges on the source objects. For example, if you want to select from an object in the OE schema, you must give “grant select on object” to the RT Target user
- Verify all location registrations. For example, source locations must be registered to the source machine and the schema in which the actual tables reside.
- Verify connectors are deployed :
  - a. Is the directory object for the external table deployed?
  - b. If you look into the code for the mapping, do you have schema references or dblinks => if not verify location registration and connectors

### Common Errors

This is a list of common errors in deployment, and the most common causes for them:

- ORA-00942 => You have not given explicit grants on the source object to the RT Target user => You have registered the source location incorrectly
- ORA-00902 => You are missing a dependent construct while creating a PL/SQL package; deploy the dependent object first, then redeploy this object

## Executing the Code

Execution of code is only relevant to:

- Mappings
- Processes

The reason for this is that these are components that can be run and that spawn off jobs that perform your ETL and actually load data. Executing your Code In all but the simplest cases and in test scenarios you should be executing process flows rather than mappings. Follow these rules on executing your code:

- Determine which objects must be executed (if you have more, create a process flow and embed in there)
- Choose the execution mechanism and understand what to do here
- Execute the process flows
- Verify the loads using the administration tools of either:
  - The scheduler
  - Warehouse Builder After executing your code, you are all set to run your warehouse loads on a regular basis.

## More Information

For more detailed information take a look at these additional resources:

- Oracle white papers on OTN for detailed information about specific functionality
- The OTN forum on Warehouse Builder where you can ask questions. The site is monitored by Oracle Warehouse Builder Product Management (<http://www.oracle.com/forums/forum.jsp?forum=57>)
- The Warehouse Builder documentation:
  - a. User Guide
  - b. Installation and Configuration Guide
  - c. Scripting Guide
  - d. Transformation Guide

For any feedback or enhancements to this Quick Start Guide, please do use the Warehouse Builder OTN forum as well.

## Summary

To reiterate, you have just completed a set of steps using Warehouse Builder, which have allowed you to go from nothing to a working data warehouse. These steps were:

- Installed client and server software
- Set up the runtime platform
- Set up the design repository
- Created a Project
- Created the modules (sources and targets)
- Created mappings (data flows to transform and load data)
- Created processes to manage the data load
- Generated the code from the design
- Deployed the code to the database
- Executed the process flows and loaded data

Now you are ready to create many more projects using the same flow.