Oracle Warehouse Builder 10*g* Release 2

Integrating Packaged Applications Data

June 2006

ORACLE

Note:

This document is for informational purposes. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described in this document remains at the sole discretion of Oracle.

Oracle Warehouse Builder 10*g* Release 2

Integrating Packaged Applications Data

INTRODUCTION

In the current enterprise software market, it is not so much a question whether or not you have implemented an ERP or CRM system, but more a question of which one you chose to implement.

As a result of that decision, you are now looking into either integrating the ERP data with other data, or you are looking at how to extract valuable information for your decision processes. In both cases it is vital to have good extraction capabilities allowing you to easily and effectively move data into different systems, such as a data warehouse or a data mart.

Warehouse Builder is Oracle's tool to easily integrate this data, and in the meantime infuse data quality checks, compliance monitoring and metadata management into the integration process.

The connectors for eBusiness Suite and PeopleSoft are new, the connector for SAP is greatly enhanced for this release.

The paper gives an overview of the connectors for Oracle eBusiness Suite, SAP R/3, and PeopleSoft that are available with Warehouse Builder 10*g* Release 2. The paper goes into details on how the individual connectors work and which benefits you can derive from implementing them at your organization.

EXECUTIVE OVERVIEW

Warehouse Builder 10*g*R2 provides access, both from a metadata and data extraction perspective to the following packaged applications:

- Oracle eBusiness Suite
- PeopleSoft systems
- SAP R/3 systems

Oracle eBusiness Suite

Warehouse Builder's eBusiness Suite connector is a solution that customers can choose to create a custom reporting or warehousing environment, besides existing pre-packaged solutions such as Daily Business Intelligence. Warehouse Builder now natively understands the metadata in the eBusiness Suite and makes extracting from it much simpler and more effective. The extraction from the eBusiness Suite is a SQL based extraction mechanism

PeopleSoft

Next to the eBusiness Suite integration, PeopleSoft's packaged applications are now also added to Warehouse Builder as a connector. The PeopleSoft connector allows Warehouse Builder to natively understand the metadata structures of the PeopleSoft applications. Once you have imported the metadata you can use all the facilities in Warehouse Builder to integrate this data using the SQL based extraction mechanisms.

SAP R/3

The third connector for Warehouse Builder is the SAP R/3 connector, which allows you to natively understand the SAP business metadata. Since SAP has a proprietary language, ABAP, to do batch extractions, the connector allows you to extract data using ABAP. Using ABAP means that you are using the native language in SAP for data manipulation. However, instead of manually writing the code you now generate it from the Warehouse Builder mappings. This way Warehouse Builder allows you to work on transparent tables, pool and cluster tables giving you access to the entire R/3 system in an efficient and convenient manner.

INTEGRATING SAP DATA

SAP R/3 is a very different system compared to the SQL based eBusiness Suite and the PeopleSoft systems. First of all the native data manipulation language is ABAP, which is a proprietary SAP language. Secondly table names are, if possible, even more cryptic than in the SQL based ERP systems, and thirdly, not all tables are database tables. Some tables in SAP are called pool tables and cluster tables, which are logical tables comprised of more that one physical object.

Reading and Importing Metadata

The first step, within Warehouse Builder, is to locate and import the metadata from the packaged application. With SAP this very important, because of the not so straightforward naming of physical objects.

Within Warehouse Builder you connect to the SAP business definition metadata. This metadata describes the SAP objects in business terms rather than using technical names of for example the database columns and tables. This metadata also masks the usage within SAP of Pool and Cluster tables, and shows them as regular single objects to the ETL developer.

To get access to the business metadata on the SAP system you use an RFC call with an SAP GUI account as authentication. Ones you have a connection, you can browse the metadata showing you both the table name and more importantly the business description.

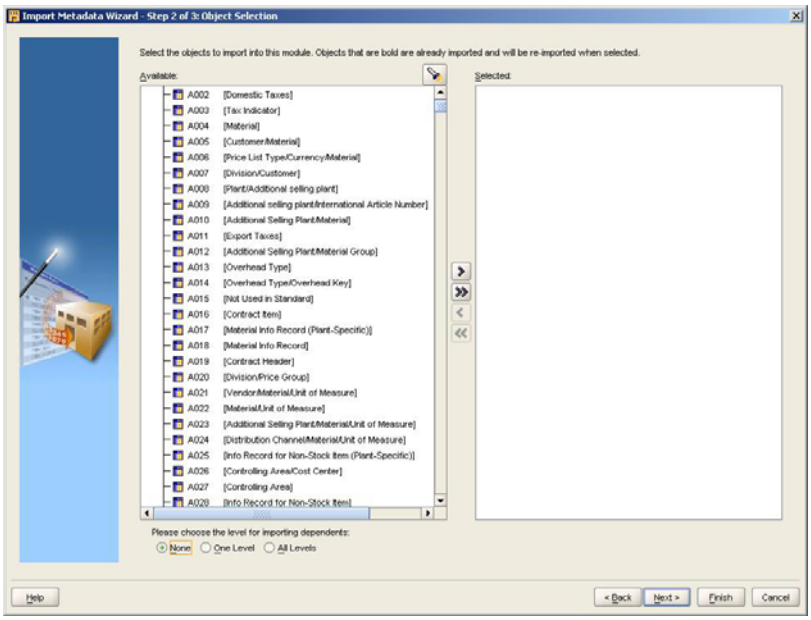


Figure 1 Business names to help you select the correct tables

Warehouse Builder retains these business names (which can also be extracted in German) throughout the product.

Apart from browsing source metadata as shown in Figure 1 you can also browse a business domain to focus on a specific area within the SAP system.

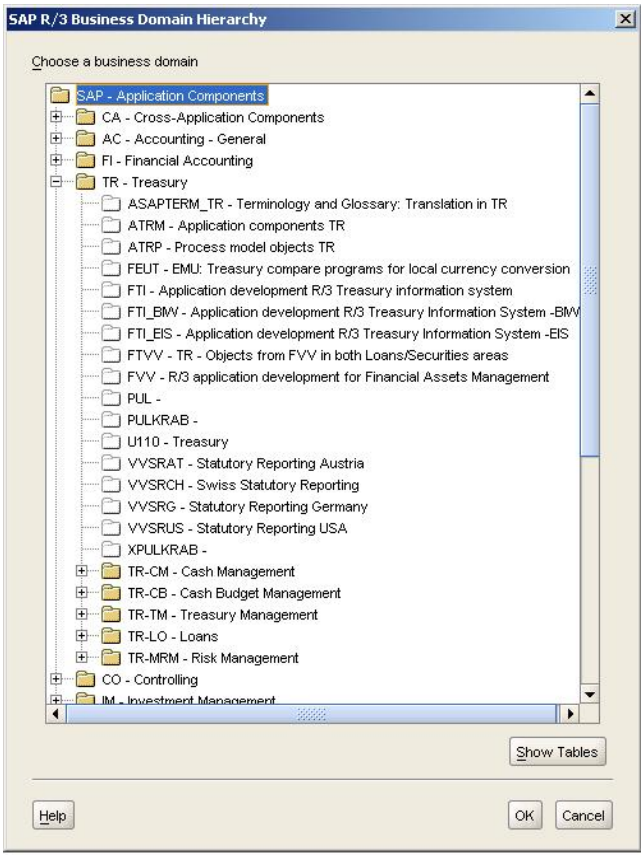


Figure 2 Application Components in a hierarchy

As you can see in Figure 2, using the business domain gives you a hierarchical view of the SAP system, making it easier to find the specific sources you are interested in.

Browsing Object Relations

The data object editor allows you to look at the SAP objects in relation to each other. You can easily see relations between objects (as defined by keys). You also have a detailed view of the properties, columns and descriptions.

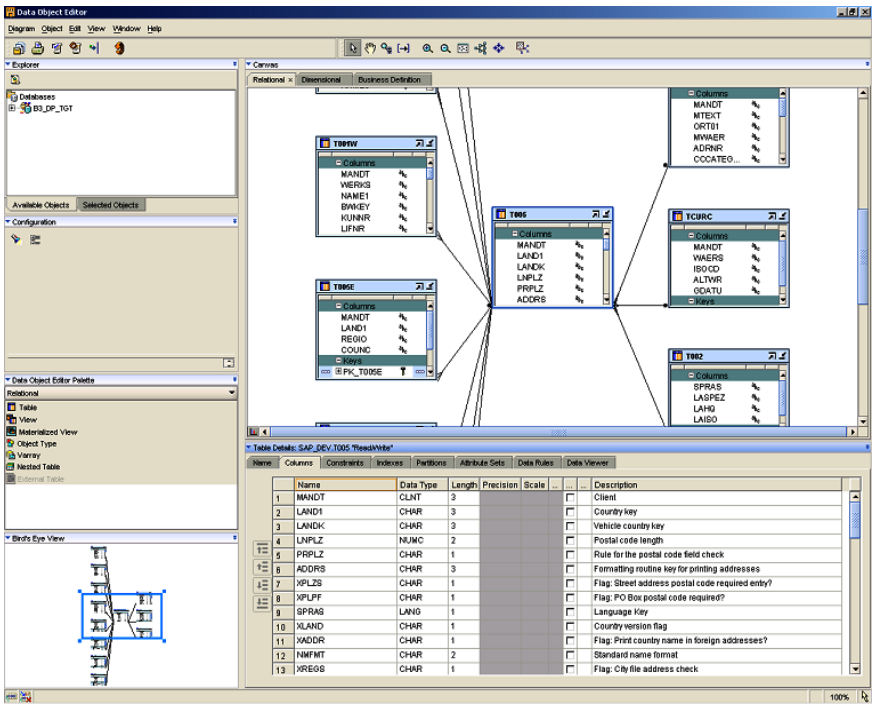


Figure 3 Relationships in a diagram

Now that you have imported the SAP metadata and understand the relationships, the meaning of the columns etc, you can start working on the ETL designs.

The ETL design and process in detail

There are a number of special additions for SAP ETL structures, which are related to the differences between SQL and PL/SQL constructs and ABAP, and between an Oracle environment and an SAP environment.

Once the metadata is imported into the Warehouse Builder repository and the relationships are understood, the ETL developers can treat SAP objects as any other data source in the graphical mapping tool.

The mapping process creates the data flow from the SAP system to the Oracle system and allows the developer to create a logical data flow using the metadata elements.

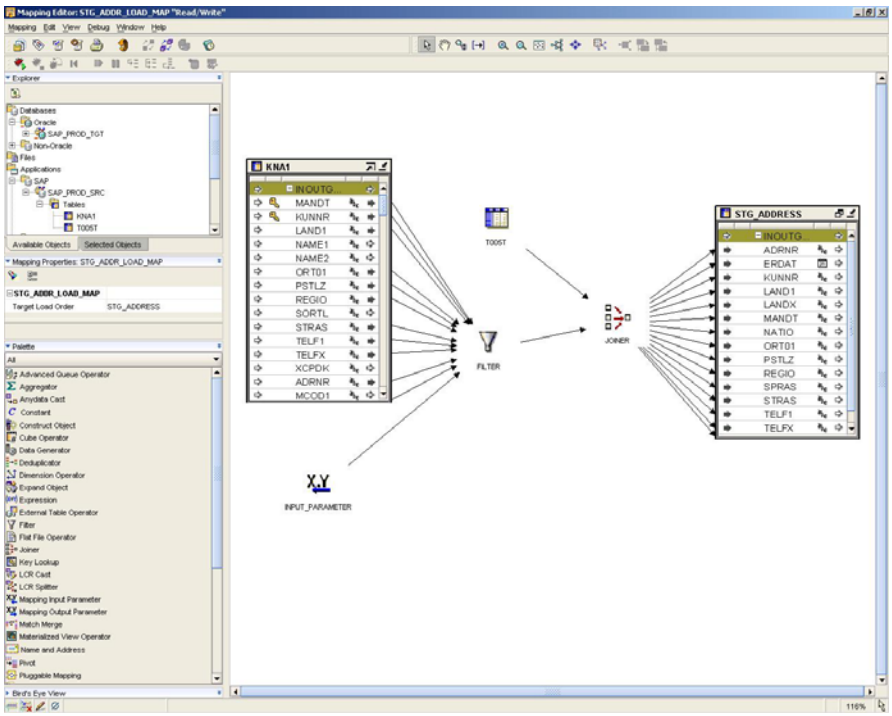


Figure 4 A data flow loading SAP data into an Oracle table

The ETL developer does the design using the Mapping Editor. He or she will drag and drop the tables into the editor and create a data flow. This data flow will in turn be used to generate the ABAP code that actually runs on the SAP system.

Warehouse Builder can automate all of the loading aspects, and you can incorporate the SAP extraction in a process that extracts data from other sources. This is typically done within a process flow. This design activity ties the entire ETL system together so that the right jobs run at the right time.

Architecture

The way Warehouse Builder works, is that the transformations (the really hard work) in ETL are done in the Oracle database using the appropriate SQL constructs. This philosophy is also applied to working on SAP data.

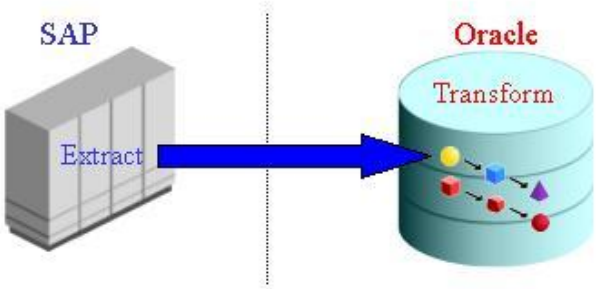


Figure 5 Extract in SAP and transform in Oracle

The idea is to extract the data from SAP in a targeted set, and then transform it within the Oracle database to fit within the data store. This means that the set of operations the integrator is intended to do is more limited than for Oracle mappings.

Extraction Steps

So how does Warehouse Builder achieve the extraction from the SAP system? If you follow the automated route, the following step are done:

1. The mapping is executed, either by hand from Warehouse Builder or automatically from a schedule (SAP) or process flow.
2. The generated code gets deployed to the SAP application server. If required mapping parameters are substituted before deployment to the SAP server.
3. The ABAP program select the data and generates a flat file on a specified directory on the SAP machine
4. Upon completion Warehouse Builder initiates an FTP command to get the data file from the SAP machine to the Oracle machine
5. Warehouse Builder initiates a SQL Loader step to load the data into the mapping target and your mapping completes

From a developer's perspective all these 5 steps are a single design, and Warehouse Builder automates all of this, so there is no real need to consider all these details.

In some cases it might not be possible to allow direct deployment onto the SAP system. In that case the SAP GUI is used to upload the ABAP and execute it within the SAP system. If that is the case, Warehouse Builder accommodates this by allowing you to save the, parameterized, ABAP to the file system. Your SAP staff can then move this code around in the SAP environment. Once the ABAP program has run, the FTP needs to be initiated and the load step must be executed. All of the latter parts can again be automated by constructing an appropriate process flow.

Capturing Changed Data

A crucial piece to ensure you are picking the exact data set you want involves capturing just the changes since the last time you extracted. For this Warehouse Builder allows you to parameterize the ABAP query and add a date or timestamp on which to filter at the moment where you run the mapping. This way you can ensure to not always extract all data but the relevant subset.

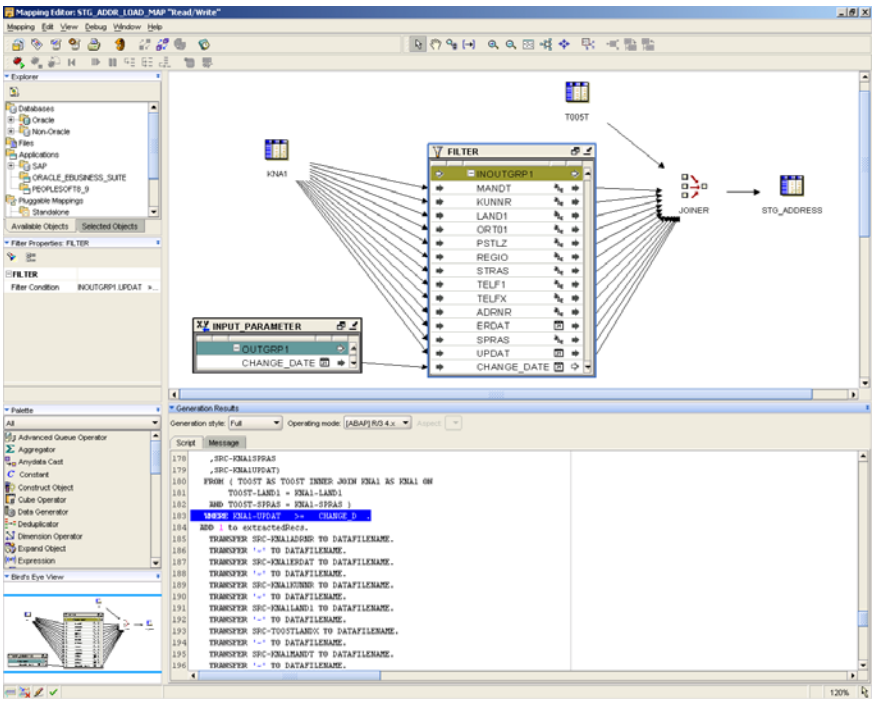


Figure 6 Parameters used to filter only new data

As with all of Warehouse Builder's mappings, you can add parameters to the mapping, which can accept outside information, like a "last loaded date" for example. Within your process flow you can then read this "last loaded date" into a variable and feed it into the mapping. The generated ABAP is then recompiled and sent to the SAP system to run and extract the data.

Configuration and Execution Options

As the execution of ABAP leads to the creation of a data file it is important that you have control over the location where this file is located. For this you can configure the mapping to determine which file name and directory is used.

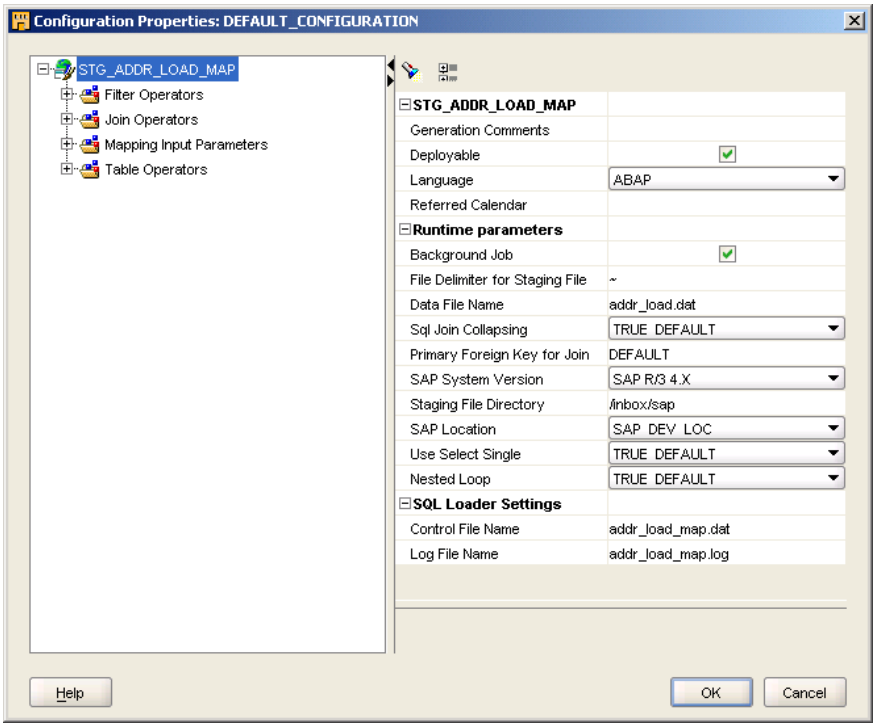


Figure 7 Configuring SAP mappings

You can decide and configure, and since this is related to the generate code manipulate these locations per system you are deploying your code to.

Background Execution

As many SAP systems do not allow long running jobs if they are not background jobs, a mapping job can be created as running in the background. Upon execution the SAP job will then become a background job allowing even long running jobs to run to completion.

Optimizing your generated ABAP

As with any data intensive language, you may sometimes want to optimize your code to behave in a certain way. To ensure you can join tables in the exact way you want it to, Warehouse Builder provides the capabilities to do join ranking.

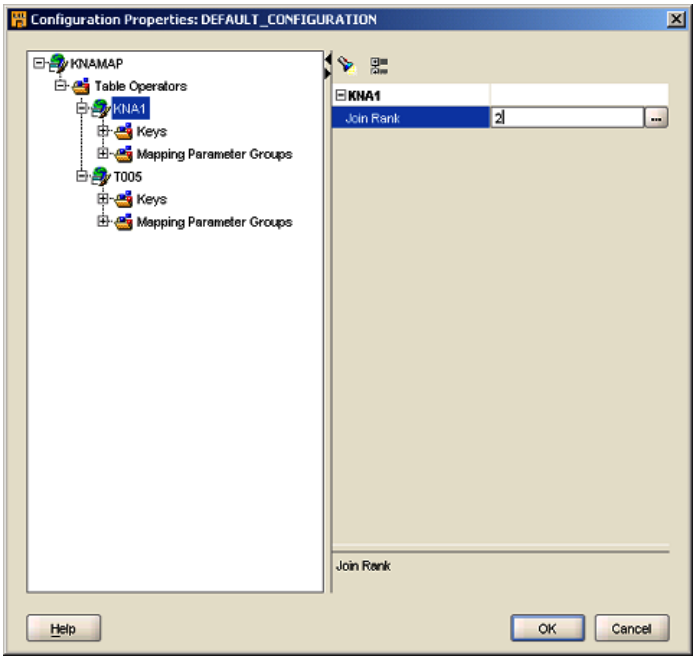


Figure 8 Ranking joins to drive query construction

In join ranking you manually override the order in which the ABAP joins the tables together. By doing so you determine which table is the driving table and you effectively force Warehouse Builder to follow your join structure.

Manual or Automated Execution

You can choose to have Warehouse Builder manage the entire extraction process, in which case it is fully automated, or you can choose to manually load the ABAP into the SAP system. The latter could be something your SAP administrators wish to do because it gives them more control on the environment.

Either way you choose Warehouse Builder monitors the actual loading into the Oracle environment, ensuring you have all relevant audit details to review the complete load process.

INTEGRATING EBUSINESS SUITE AND PEOPLESOFT DATA

In Warehouse Builder both these systems are viewed as SQL based from an extraction perspective. Both eBusiness Suite and PeopleSoft solutions use SQL as their main access languages and this is also replicated in the ETL environment for these source systems. Because of this similarity in access we cover the systems in a single section.

Reading and Importing Metadata

As we saw before, the first step is to locate and import the metadata from the ERP system. As with SAP, a big portion of the added value of the eBusiness Suite and PeopleSoft connectors comes in right here at the beginning of the process.

The connection to the system is made using database users. These users are typically set up by the administrator to allow this type of access, making sure the user that is used here has the appropriate privileges.

Once connected, Warehouse Builder understands the metadata catalog for the ERP (or CRM) system, and rather than showing you a flat list of tables in the database schema you see an organized list.

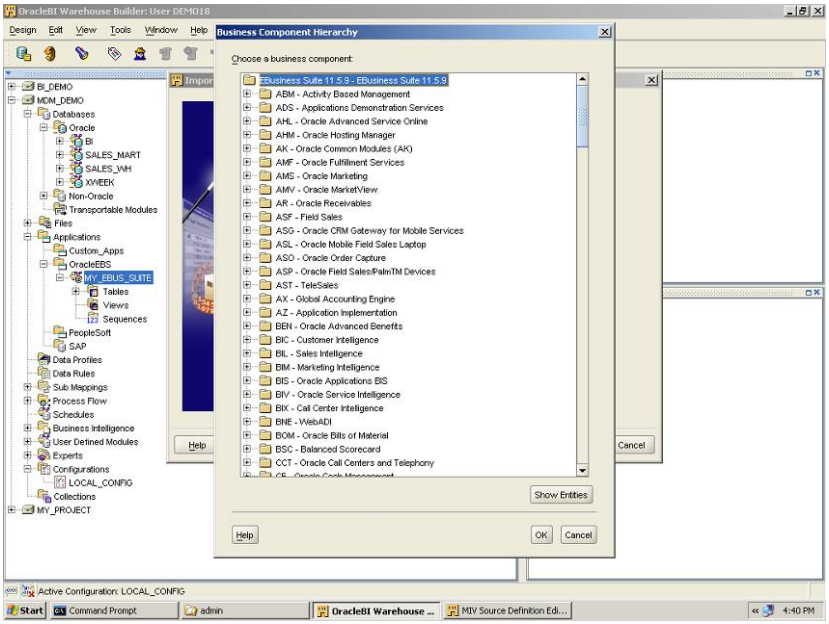


Figure 9 Importing metadata from the eBusiness Suite

As you can see in Figure 9 the applications modules are nicely presented as the highest level of the system. From there, using the business names of the modules you can drill down to the area of choice. If you are building a subject oriented data mart (like receivables for finance) you can select the entire module and all tables are marked for import into the Warehouse Builder metadata repository.

The ETL design and process in detail

As we said at the start of the eBusiness and PeopleSoft section, these systems are SQL based, and are handled as such in Warehouse Builder. Once the metadata is imported the ETL developers will work with packaged applications as they would with other SQL based systems. To ensure that Warehouse Builder can extract data from non-Oracle database applications, the locations allow for usage of Oracle's Heterogeneous Services. This will make non-Oracle database transparent for Warehouse Builder.

Understanding the Data and Structure

For ETL there is a bit more structural understanding needed. So within the nice grouping of objects the ETL developer should dig a level deeper to understand relationships. Therefore Warehouse Builder with its understanding of the metadata of the application shows the developer the set of tables that make up a logical entity.

For example, within PeopleSoft CRM, we can drill down in the hierarchy to view the logical entity contacts, then we can drill into what makes up a contact. This is shown in Figure 10.

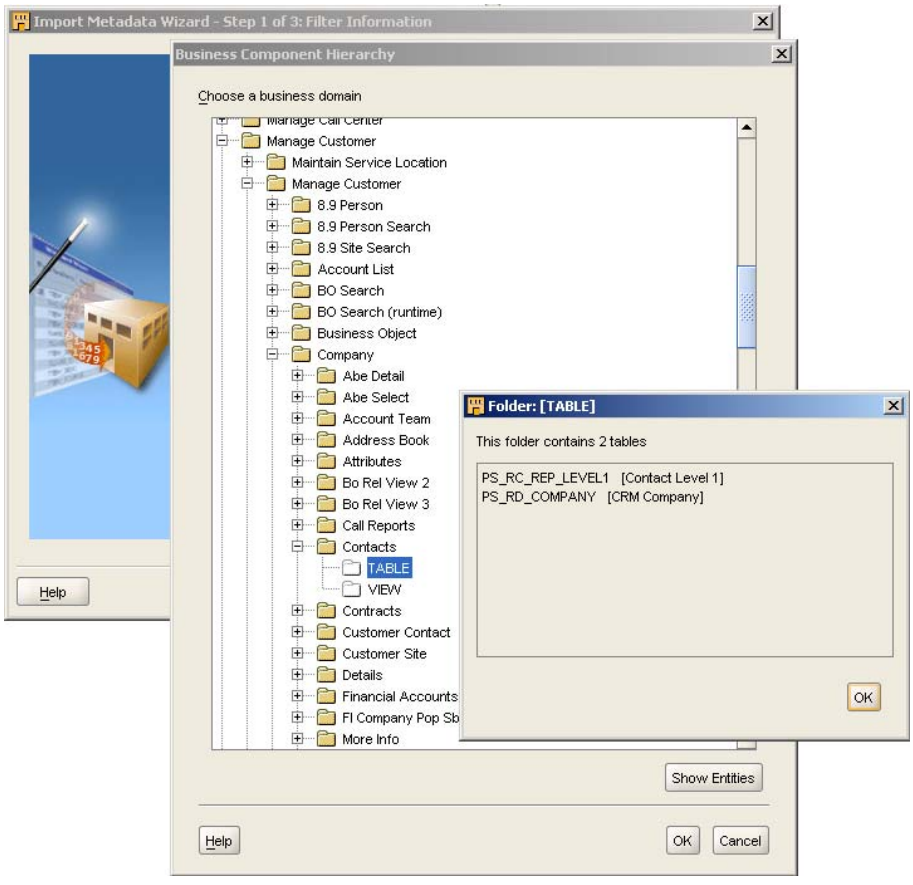


Figure 10 Discover the actual tables making up a logical entity

Once the metadata import is completed, you can browse the objects to understand the structure and the contents directly from within Warehouse Builder.

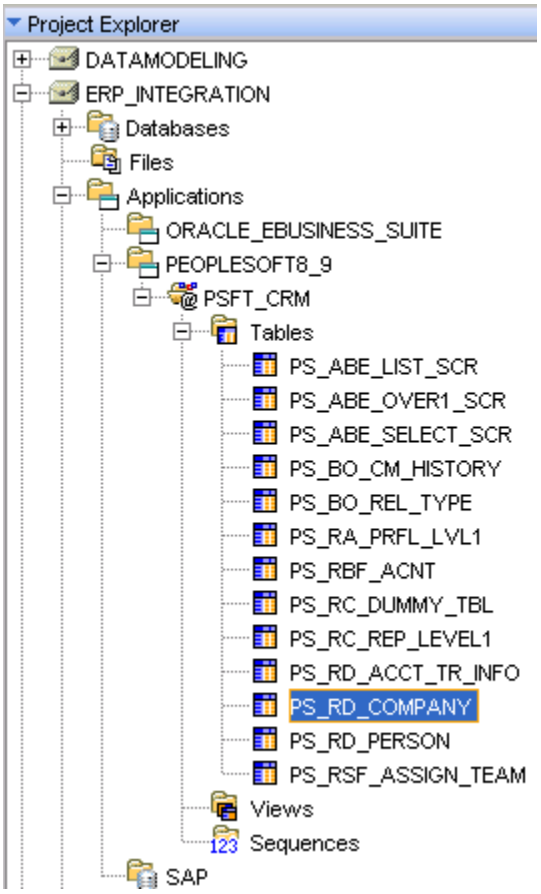
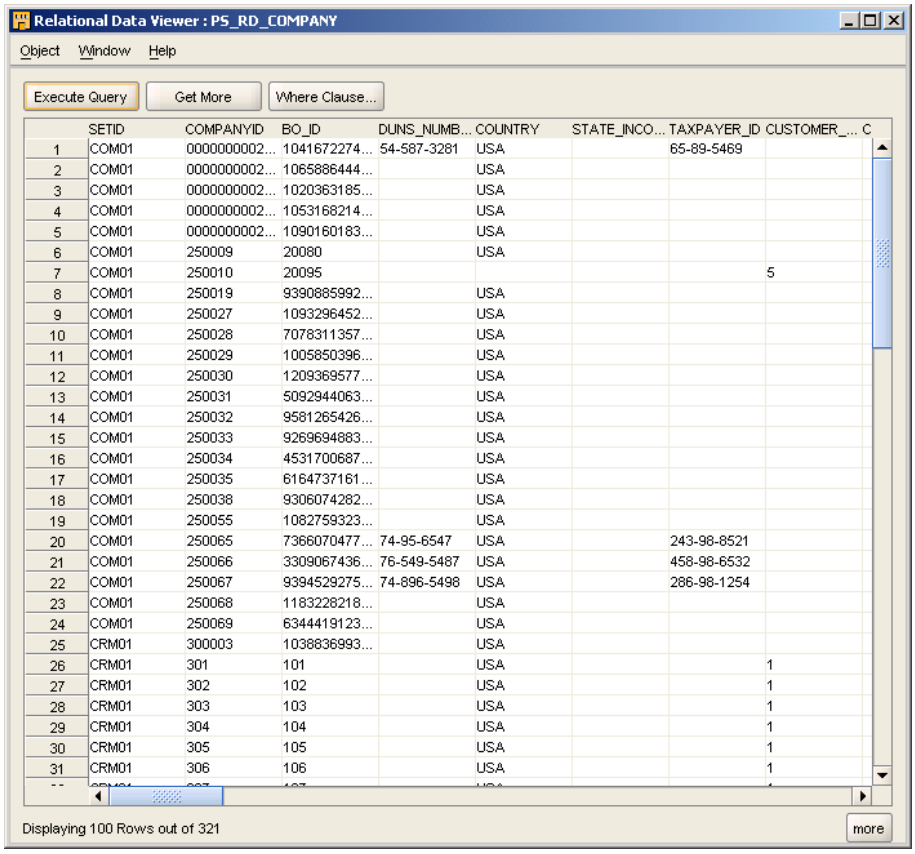


Figure 11 Imported PeopleSoft metadata in Warehouse Builder

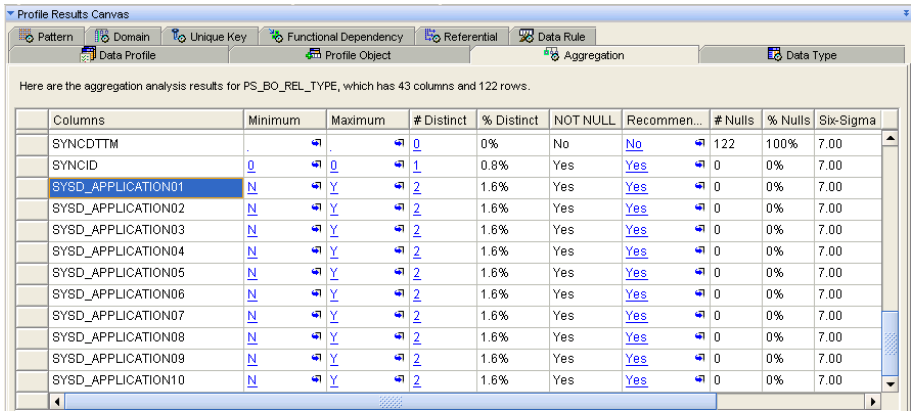
Data is viewed from within the data viewer (Figure 12), structure and descriptions on the objects can be viewed from within the data object editor. Both are launched from the Warehouse Builder Design Center.



	SETID	COMPANYID	BO_ID	DUNS_NUMB...	COUNTRY	STATE_INCO...	TAXPAYER_ID	CUSTOMER_...	C
1	COM01	0000000002...	1041672274...	54-587-3281	USA		65-89-5469		
2	COM01	0000000002...	1065886444...		USA				
3	COM01	0000000002...	1020363185...		USA				
4	COM01	0000000002...	1053168214...		USA				
5	COM01	0000000002...	1090160183...		USA				
6	COM01	250009	20080		USA				
7	COM01	250010	20095					5	
8	COM01	250019	9390885992...		USA				
9	COM01	250027	1093296452...		USA				
10	COM01	250028	7078311357...		USA				
11	COM01	250029	1005850396...		USA				
12	COM01	250030	1209369577...		USA				
13	COM01	250031	5092944063...		USA				
14	COM01	250032	9581265426...		USA				
15	COM01	250033	9269694883...		USA				
16	COM01	250034	4531700687...		USA				
17	COM01	250035	6164737161...		USA				
18	COM01	250038	9306074282...		USA				
19	COM01	250055	1082759323...		USA				
20	COM01	250065	7366070477...	74-95-6547	USA		243-98-8521		
21	COM01	250066	3309067436...	76-549-5487	USA		458-98-6532		
22	COM01	250067	9394529275...	74-896-5498	USA		286-98-1254		
23	COM01	250068	1183228218...		USA				
24	COM01	250069	6344419123...		USA				
25	CRM01	300003	1038836993...		USA				
26	CRM01	301	101		USA			1	
27	CRM01	302	102		USA			1	
28	CRM01	303	103		USA			1	
29	CRM01	304	104		USA			1	
30	CRM01	305	105		USA			1	
31	CRM01	306	106		USA			1	

Figure 12 Viewing data directly in Warehouse Builder

You can also run data profiling on the tables to really discover contents and relations. One of the areas where data profiling can be very effective in packaged applications is in the area of application extensions.



Columns	Minimum	Maximum	# Distinct	% Distinct	NOT NULL	Recommen...	# Nulls	% Nulls	Six-Sigma
SYNCDTTM			0	0%	No	No	122	100%	7.00
SYNCID	0	0	1	0.8%	Yes	Yes	0	0%	7.00
SYSD_APPLICATION01	N	Y	2	1.6%	Yes	Yes	0	0%	7.00
SYSD_APPLICATION02	N	Y	2	1.6%	Yes	Yes	0	0%	7.00
SYSD_APPLICATION03	N	Y	2	1.6%	Yes	Yes	0	0%	7.00
SYSD_APPLICATION04	N	Y	2	1.6%	Yes	Yes	0	0%	7.00
SYSD_APPLICATION05	N	Y	2	1.6%	Yes	Yes	0	0%	7.00
SYSD_APPLICATION06	N	Y	2	1.6%	Yes	Yes	0	0%	7.00
SYSD_APPLICATION07	N	Y	2	1.6%	Yes	Yes	0	0%	7.00
SYSD_APPLICATION08	N	Y	2	1.6%	Yes	Yes	0	0%	7.00
SYSD_APPLICATION09	N	Y	2	1.6%	Yes	Yes	0	0%	7.00
SYSD_APPLICATION10	N	Y	2	1.6%	Yes	Yes	0	0%	7.00

Figure 13 Detecting data in extension columns

Figure 13 shows a high level summary of the data in some of the extension columns, allowing ETL developers to easily understand required transformation logic.

Creating ETL Routines

Once the source metadata is imported and the developer is happy with the set of objects, the next step is to create a target in which to load the data. This can be an existing or new database schema (or of course a flat file or XML output file). In Warehouse Builder this is represented as a new module.

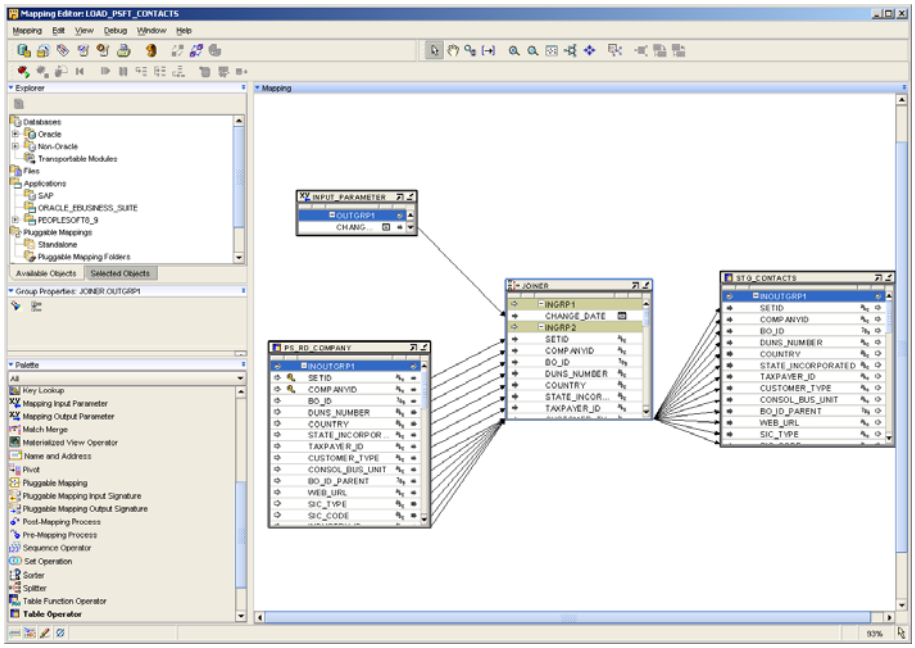


Figure 14 Loading contacts based on a change date parameter

In that module we create the mappings to move data from the source to the newly designed target.

The mapping example in Figure 14 accepts a date parameter and then limits the contacts that are loaded to fall in a range spanning a single day. This allows the mapping to be run every day and process exactly one day of new or changed data.

The code generated uses the Oracle database engine to efficiently transform and load the data. In this example the code merges records into the target table.

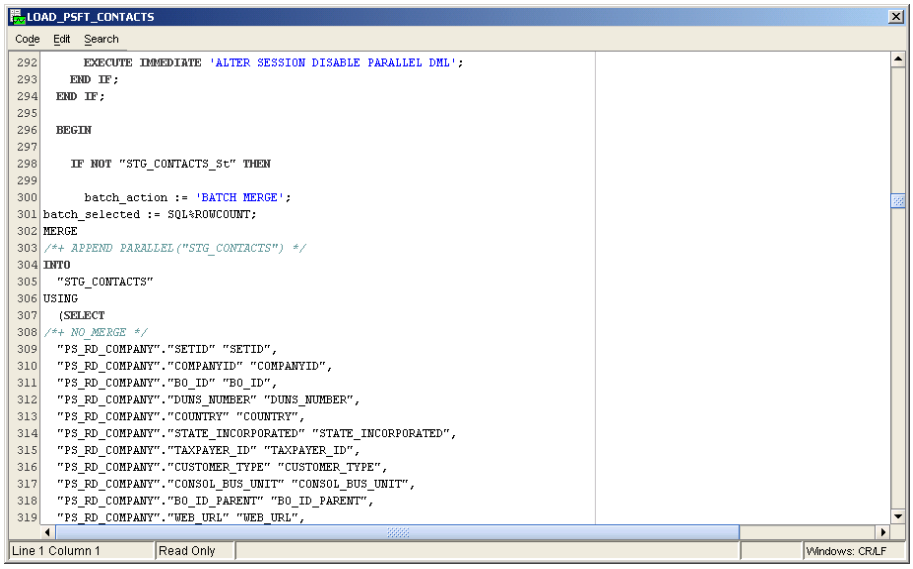


Figure 15 Generating optimized code for Oracle

Once you have the mapping created the generated code gets deployed into the target database schema and regular runs extract data from the packaged application.

To manage the environment you can use process flows and scheduling within Warehouse Builder. All of these features are more generic in nature and are not discussed as part of this section. More information is available on Oracle Technology Network (http://www.oracle.com/technology/products/warehouse/htdocs/OTN_collateral.html).

CONCLUSION

Warehouse Builder 10*g* Release 2 delivers a comprehensive set of packaged applications Connectors on the Oracle data warehouse platform.

The eBusiness Suite and PeopleSoft Connectors provide a robust and easy to use interface into the metadata and data of these commonly used packaged applications. By leveraging the SQL capabilities and transformations in the Oracle database, bulk extraction is fast and straightforward.

The SAP Connector provides a scalable, bulk extraction mechanism for SAP R/3. By generating native SAP ABAP code, Warehouse Builder fits within your SAP maintenance structures and is unobtrusive to the SAP system.

Oracle Warehouse Builder 10*g*R2 – Integrating Packaged Applications Data
June 2006
Author: Jean-Pierre Dijcks

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
www.oracle.com