Corporate Services for the Natural Resource Sector

Information Management Branch

Procedures for Developers Working in a Corporate Versioned Repository

Last Updated:	January, 2012
Version:	1.0.0
Document:	NRS_Developers_Working_in_a_Corporate_Versioned_Repository_1.0.0.docx

Document History

Version	Description of Change, Review or Approval	Author	Date
0.9	First draft of the guidelines consolidated from Natural Resource Sector data modelling standards • Developers working in a Versioned Development Repository_(V 2.4.1) (MoE/MAL/ ILMB)	Randy Hoffner (MoE/MAL Author)	December 2011
1.0	Update standards to include new repositories.	Bill Holland	January 2012

Contents

1 INTRODUCTION

This document describes mandatory principles applied to a development environment in a shared and versioned corporate Oracle designer data model repository. It was originally developed by the Information Management Branch for Corporate Services (CSD/IMB) and by the Land and Resource Data Management Services (ILMB). It is now maintained by the Architecture Section within the Information Management Branch, Corporate Services Division for the Natural Resource Sector.

1.1 Purpose

When modelling in a versioned repository, there are principles and procedures which must be understood to develop in a coordinated manner with Data Administration (DA). This document covers principles of check out – check in rules, rules for deleting and copying objects, rules about external references, and procedures for model updates (maintenance and patches) and new model development.

1.2 Audience

The audience for this document is primarily vendors on contract with the Natural Resource Sector, who are developing data models for the ministries business clients. These principles and procedures will also apply to DA and staff who have Oracle IDS licenses with update privileges, to one of the versioned corporate oracle repository. Also the document sections 2, 3, 4, 5, and 6 are valuable to Business Portfolio Manager (BPM) directing application development.

1.3 Scope/Exclusions

- This document provides details on the procedures related to modelling in a versioned oracle repository.
- This document will not provide explanations of principles and rules that are to be used when data modelling.
- It is not meant to supplement or further expand on the process flow to get data into the databases used by ministry applications.
- This is not a training document

1.4 Assumptions

It is assumed that developers understand how to use Oracle Software Configuration Manager and have formal training in data model development. It is also assumed that developers understand application development and the SDLC.

2 VERSIONING PRINCIPLES

2.1 Major release vs. maintenance release vs. patch release (mandatory)

- Version numbering follows the current delivery standard and is used by DA to track application history within the repository by making configurations <app><version>
- For clarification, in documentation when <version> is stated it is of the format ##.##.## where the first '##' represents a major release, the second '##' represents a minor release, and the third '##' represents a patch release.
- The version number of a maintenance (minor) release or a patch **must be supplied** to CSD DA before checking out a repository Data Model data model container. A maintenance (minor) release usually spaced annually or bi-annually.

2.2 Versioned Data Models in RDRPROD1 and NRSMR1 (mandatory)

All application updates to versioned Data Models containers occur **within** the Corporate Oracle Repositories **RDRPROD1 and NRSMR1**. This means that any maintenance or patch release "change" occurs within the corporate repository. Version 4.2 going to version 4.3 or version 4.2.5 is considered a version change occurring via check out in the repository.

For maintenance or patch releases, all objects in the data model are to be brought up to current standards including the Oracle data base tab, except where application code will break.

Version 4.25 going to Version 5 is a major version change and can be considered to be done in the developer's repository or in the corporate repository. See 3.3 below

2.3 New or non versioned Data Models

New Data Models or total new versions of existing Data Models (as stated in 3.1) are developed at the discretion of the developer whether it is in their own repository or the Corporate Production Repositories. If it is a total new version (the major version number) then all objects must pass the DA approval process and conform to all standards even if objects have not changed from the previous versions. This principle holds whether the modelling is done in either the corporate or developer repository.

3 PROCEDURE for applying an update for a VERSIONED DATA MODEL in a Corporate Oracle versioned PRODUCTION repository (mandatory)

This process is for all developers that are developing in one of the versioned Oracle repositories with regards to a maintenance or emergency fix to a Data Model.

Quick Notes:

a) DA checks out versioned container on request for application patches or maintenance release. Developer is responsible for checking out objects and **providing appropriate check out notes.**

b) Developers do not check in objects, that is the responsibility of DA.

c) For small patches, logical and physical review may happen at the same time. It is advised that if new objects are created, a logical review happens before transform to physical

3.1 Developer setup in RDRPROD1 or NRSMR1

- It is the responsibility of DA to expose and to check out the Data Model folder in the work area WA_WIP_DEVELOPMENT, on request from the BPM, on which development is occurring. The DA will provide appropriate check out notes (see **Check in-checkout principles applied to versioned Data Models** below)

- The developer is given select, insert, update, delete on both the Data Model container and WA_WIP_DEVELOPMENT work area. DA's checkout notes on the container will indicate status of Data Model, developer contact and date.

- The developer is responsible for providing container property updates to DA by email or word document as required. Typically this includes title, description, summary and objectives content

- Developers are not supplied with container privileges.

3.2 Software Requirements

- If this is an update and the business has software requirements and an Object model that the business area wants to maintain, the developer must update the requirements document and the object model. Any Diagrams and the updated object model must be uploaded into the Data Model container in the Files section along with any associated reports as PDF's and distributed to BPM for sign off. A Developer must supply checkout notes if changes to any existing objects occur. (See **Check in-check out principles applied to versioned Data Models** below)

3.3 Logical Modelling

- The developer may work with the DA in developing the logical model with reference to entity naming and code table sourcing or can wait till they are finished their first DA review for any new entities. If code table referencing is required the developer must request DA to make the referencing Data Model container available in WA_WIP_DEVELOPMENT in a checked out state with the appropriate entity in a checked out state. DA will provide the container checkout note to include the text, "checked out for reference". See Code table Usage document for searching what entities are sourced and methodology for identification. The Developer must supply checkout notes if changes to any existing objects occur. (See **Check in-checkout principles applied to versioned Data Models** below)

- Before requesting the initial logical data model review the model must have updates applied, including the Entity Relationship diagram(s), then three PDFs uploaded as files to the Data Model container, either replacing the ones that exist or creating new ones if they do not. An Entity relationship diagram(s) PDF(s), Entity Definition report and a report of Entities and their Attributes are the PDFs required. These PDFs will be supplied to the business area to make sure the descriptions for entities and attributes provided are accurate and reflect meaning that is coherent to non business users. Any changes to descriptions made on direction of the business area should be forwarded to DA for review and then to the developer for input. The developer is responsible for resolving any external references in the Data Model container.

- The BPM on the project will schedule the **logical** review with the DA.

- DA now does the initial review and supplies a word document of correction requests to the developer or an email request for any corrections. The developer makes the corrections to the entities and re-submits for model approval. This might require several iterations until the logical model has been approved. It is the responsibility of DA to check-in the logical objects that have passed DA logical review. It is at the discretion of DA whether to do this now or after the physical review. However, if it is done at this time and changes need to be made to the logical model based on the physical review by DA or DBA, the logical objects may need to be checked back out with check out notes. (See **Check in-checkout principles applied to versioned Data Models** below).

3.4 Physical Modelling

- After the initial logical data model review by DA is completed and approved, the logical model can be driven to physical definitions through the use of the database design transformer (see section 11). Physical objects should be checked out before a transform is started including the Oracle database tab. Before the physical definitions of the entities are driven, the developer must update the storage definitions and at least one Oracle Database which contain the appropriate table spaces and database Roles. (These parameters can be obtained from DBA for operational datasets only). These two repository objects do not need detailed checkout notes but any sequences, PL/SQL Definitions (packages, package body's functions and procedures), triggers, synonyms, and view definitions that reference the table definitions that are updated or changed must have checkout notes.

- When the developer has finished with transforming, the developer must supply one or more Server Model diagrams, a Table Definition Report and a document specifying any changes in the physical model from the logical model. These reports will be used in final data model review by DBA. These documents need to be uploaded to the Data Model container as PDF files, with the exception of the physical change document.

- DBA will be given access rights to the Data Model container for Physical data model quality assurance if they request.

- The BPM on the project will schedule the **physical** review with the DBA and the DA.

A list of requested changes may be compiled by DA/DBA and submitted to the developer for remedy, if necessary. Developer then notifies DA of change completion. DA checks in any physical objects except for the Oracle Database tab and the PDFs. DA informs the contractor to drive the DDL which is delivered to DBA through the Application delivery process. The DDL will conform to the standard delivery architecture for DLL structure.

3.5 Model change in the application delivery process

- Changes to the DDL may occur in the application delivery process. Designer data model changes are then required to match any change to the physical implementation of the DDL. This happens by checking out objects (using checkout notes as described in section 7) and it may happen throughout the delivery and test phase. After this process of updating the repository model, the developer will request DA to check in the changes made before the DDL is migrated to production.

3.6 Repository Data Model container migration

- It is the responsibility of DBA to inform DA that the DDL has been migrated to production and then request Data Model container check-in within the corporate oracle repository. Before the configuration is made and the container is checked-in, DA must supply change history notes in the NOTES properties box at the Data Model container level.
- DA then checks in the container and exports the Data Model to the delivery server in the case directory of the application. DA creates an updated configuration and replaces the existing configuration in WA_PRODUCTION. It will already be viewable by the ROB (Repository Object Browser - RDRPROD1 only) so grants need not be applied.

4 PROCEDURE for delivery of a NON VERSIONED or a VERSIONED Data Model to the Oracle DELIVERY Repository (These are full version replacements or new models)

This process is for developers who have their own Oracle repository to develop in and prefer not to develop over VPN.

Quick Notes:

a) All full version model reviews will happen in the corporate delivery repository “rdrdev1” that come into CSD from developer repositories. They will be found in a work area called WA_DELIVERY.

b) The final stage of **the** process is to import the finalized model into the corporate production repository “RDRPROD1” or “NRSMR1”.

c) Any model coming into the production repository must have all external references resolved. (See external references below)

d) The model will be versioned after import to “RDRPROD1” or “NRSMR1” and any changes in the delivery process will happen according to the process defined in step 4.4 and 4.5 below.

4.1 Developer Data Model setup in developer’s repository

- The developer will make a Data Model folder on request from the BPM that uses an Application folder (acronym) following the naming standards quoted in the Naming Standards Document. It is acquired by the BPM, often at white board sessions or requesting DBA to do a search for existing uses in corporate databases.

- The developer is responsible for providing the information for the container title, description, summary and objectives for the Data Model container. This can be acquired from the business area and is also part of metadata collection. The acronym, title and description must match the metadata for the dataset being modeled.

4.2 Logical Modelling

- The developer may work with the DA in developing the logical model with reference to entity naming and code table sourcing by scheduling a meeting or they can wait till they are finished their first go round for DA review. If code table referencing is required the developer must request DA for dumps of appropriate data models that contain the required information. See Code table Usage document for searching what entities are sourced and methodology for identification (ROB usage).

- Before submission for the initial data model review the completed logical model must include one or more Entity Relationship diagrams and three PDFs uploaded as files to the Data Model container. An Entity relationship diagram's PDF's, an Entity Definition report and a report of

Entities and their Attributes are the PDF's required. These PDF's will be supplied to the business area to make sure the descriptions for entities and attributes provided are accurate and reflect meaning that is coherent to non business users. Any changes to descriptions made on direction of the business area should be forwarded to DA for review and then to the developer for input.

- The software requirements and the object model are also uploaded to the application container for reference. If the object model is being worked on in conjunction with the logical model, it can be uploaded when signed off and the data model is brought into the production environment.

- The BPM on the project will schedule the **logical** review with the DA.
- The logical model may go through a reiterative process where the developer is requested to change the model to bring it to standard. DA, in this review process, supplies preferably a word document of corrections requests to the developer. The developer makes any corrections to the entities and resubmits for model approval as a new repository dmp file.

- DA logical review must be passed before any physical table definitions are generated.

4.3 Physical Modelling

- Once the logical data model review by DA is completed and approved, the logical model can be driven to physical definitions using the database design transformer. Before the physical definitions of the entities are driven, the developer must create storage definitions and at least one Oracle Database which contain the appropriate table spaces and database Roles. (These parameters can be obtained from DBA for operational datasets only). Any sequences, PL/SQL Definitions (packages, package body's functions and procedures), triggers, synonyms, and view definitions that reference the tables must be included with the data model. The developer is responsible for resolving any external references.

- When the developer has finished with the physical model, the developer must create one or more Server Model diagrams, a Table Definition Report and a document specifying any changes in the physical model from the logical model after database design generation. These documents are uploaded to the Data Model folder as files. The developer will then send DA a dmp file of the Data Model for review of the physical model.

- The physical model may also go through a reiterative process where the developer is requested to change the model to bring it to standard. During this process a new Oracle dmp file of the Data Model, with the changes completed, will be imported into the **delivery** repository and compared to the previous version of the Data Model.

- The BPM on the project will schedule the **physical** review with the DBA and DA. If it is a (Iterations with minor changes may not need to be scheduled).

- Upon approval of the physical model by DBA, DA informs the contractor to drive the DDL and submit it to DBA through the Application delivery process. The DDL will conform to the standard delivery architecture for DLL structure.

4.4 Migration and versioning (mandatory)

- After DA signs off the final delivery in the delivery repository, the Data Model is imported to the Corporate production repository (RDRPROD1 or NRSMR1) and versioned, in the WA_WIP_DEVELOPMENT work area and the contractor is given access privileges. If any more changes to the designer model are required for DBA by the developer, to match the physical implementation in the database, they will be done in RDRPROD1 or NRSMR1 by checking out the objects according to the principles stated in section 7. It is the responsibility of developer to inform DA that the changes have occurred and request DA that they are checked in.

4.5 Repository Data Model container migration

- It is the responsibility of application delivery DBA to inform DA that the DDL has been migrated to production and then request Data Model container check-in within the corporate oracle repository. Before the configuration is made and the container is checked in, DA must supply change history notes in the NOTES properties box at the Data Model container level.

- DA then checks in the container and exports the Data Model to the delivery server in the case directory of the application. DA creates an updated configuration and replaces the existing configuration in WA_PRODUCTION. It will already be viewable by the ROB (Repository Object Browser - RDRPROD1 only) so grants are not applied.

- DA will then remove insert, update, delete privileges on both the Data Model container and WA_WIP_DEVELOPMENT work area from the developer leaving only select. The privileges need not be removed from the work area if the developer is working on more than one Data Model for the ministry.

5 PROCEDURES for creating a NON VERSIONED Data Model in the Corporate PRODUCTION Repository

This process is for developers who do not have their own Oracle repository to develop in or for developers that wish to have CSD manage their application development.

Quick Notes:
a) Any model developed in the production repository must have all external references resolved. (See external references below)

b) The model will be versioned after passing full data model review and any changes in the application delivery process will happen according to the process defined in step 6.5 below. The data model may be versioned earlier on request from developer.

5.1 Developer setup in RDRPROD1 or NRSMR1

- It is the responsibility of DA to make a Data Model folder on request, from BPM. This Data Model folder will be created in the work area WA_WIP_DEVELOPMENT. Data Model folders are the responsibility of DA and are not to be deleted or renamed by the developer.

- The developer is given select, insert, update, delete on both the Data Model container and WA_WIP_DEVELOPMENT work area.

- The Application folder (acronym) following naming standards quoted in the Naming Standards Document is acquired, by the BPM, by asking DBA to do a search for existing uses in sector databases and on the apps_ux directory structure.

- The developer is responsible for providing container property updates to DA by email or word document as required. Typically this includes title, description, summary and objectives content. The acronym, title and description must be the same as the metadata for the dataset being modelled.

- Developers are not supplied with container privileges.

- If the software requirements and the object model is complete it is uploaded to the application container for reference. If the object model is being worked on in conjunction with the logical model, it can be uploaded when signed off. The finalized software requirements document should be also uploaded to the application container at this time.

5.2 Logical Modelling

- The developer may work with the DA in developing the logical model with reference to entity naming and code table sourcing or they can wait till they are finished their first DA review. If code entity or any other object referencing is required by the developer, they must request DA to make the Data Model container available in WA_WIP_DEVELOPMENT in a checked out state with the appropriate entity or object in a checked out state. DA will provide the container checkout note to include the text, "checked out for reference". See Code table Usage document for searching what entities are sourced and methodology for identification.

- Before submission for the initial data model review the completed logical model must include one or more Entity Relationship diagrams and three PDFs uploaded as files to the Data Model container. An Entity relationship diagram(s), PDFs, an Entity Definition report and a report of Entities and their Attributes are the PDFs required. These PDFs will be supplied to the business area to make sure the descriptions for entities and attributes provided are accurate and reflect meaning that is coherent by non business users. Any changes to descriptions made on direction of the business should be forwarded to DA for review and then to the developer for input.

- The BPM on the project will schedule the **logical** review with the DA.

- DA now does the logical review and supplies, preferably, a word document of correction requests. The developer makes any corrections to the entities and re-submits for model approval. This may require several iterations until the logical model has been approved.

- DA logical review must be passed before any physical table's definitions are generated.

5.3 Physical Modelling

- Once the logical data model review by DA is completed and approved, the logical model can be driven to physical definitions using the database design transformer. Before the physical definitions of the entities are driven, the developer must create storage definitions and at least one Oracle Database which contain the appropriate table spaces and database Roles. (These parameters can be obtained from DBA depending on whether this is an operational or warehouse dataset). Any sequences, PL/SQL Definitions (packages, package body functions and procedures), triggers, synonyms, and view definitions that reference the tables must be included with the data model. The developer is responsible for resolving any external references.

- When the developer has finished with the physical model, the developer must create one or more Server Model diagrams, a Table Definition Report and a document specifying any changes in the physical model from the logical model after database design generation. These documents are uploaded to the Data Model folder as files.

- DBA will be given access rights to the Data Model container for Physical data model quality assurance.

- The BPM on the project will schedule the **physical** review with the DBA and DA.

- Upon approval of the physical model by DBA, DA informs the contractor to drive the DDL and submit it to DBA through the Application delivery process. The DDL will conform to the standard delivery architecture for DLL structure.

5.4 Model change in the Application delivery process

- After DA signs off the data model in the work area WA_WIP_DEVELOPMENT, it is versioned and the container is checked out for possible change in the application delivery process. If any more changes to the designer model are required for DBA by the developer, to match the physical implementation in the database, they will be done in RDRPROD1 or NRSMR1 by checking out the objects according to the principles stated in section 8. It is the responsibility of the developer to inform DA that the changes have occurred and request DA that they be checked in.

5.5 Repository Data Model container migration

- It is the responsibility of DBA to inform DA that the DDL has been migrated to production and then request Data Model container check-in within the corporate oracle repository. Before the configuration is made and the container is checked in, DA must supply change history notes in the NOTES properties box at the Data Model container level.

- DA then checks in the container and exports the Data Model to the application delivery server in the case directory of the application. DA creates an updated configuration and replaces the existing configuration in WA_PRODUCTION. It will already be viewable by the ROB (Repository Object Browser - RDRPROD1 only) so grants need not be applied.

- DA will then remove insert, update, delete privileges on both the Data Model container and WA_WIP_DEVELOPMENT work area from the developer leaving only select. The privileges need not be removed from the work area if the developer is working on more than one Data Model for the ministry.

6 PROCEDURE for developers in WAREHOUSE DATA MODELS

The process and procedures for warehouse data modelling and deliveries (formally ILMB) has transferred to the **Enterprise Data Services**, **Ministry of Labour, Citizens' Services and Open Government.**

7 CHECKIN - CHECKOUT PRINCIPLES applied to versioned Data Models (mandatory)

7.1 Rules

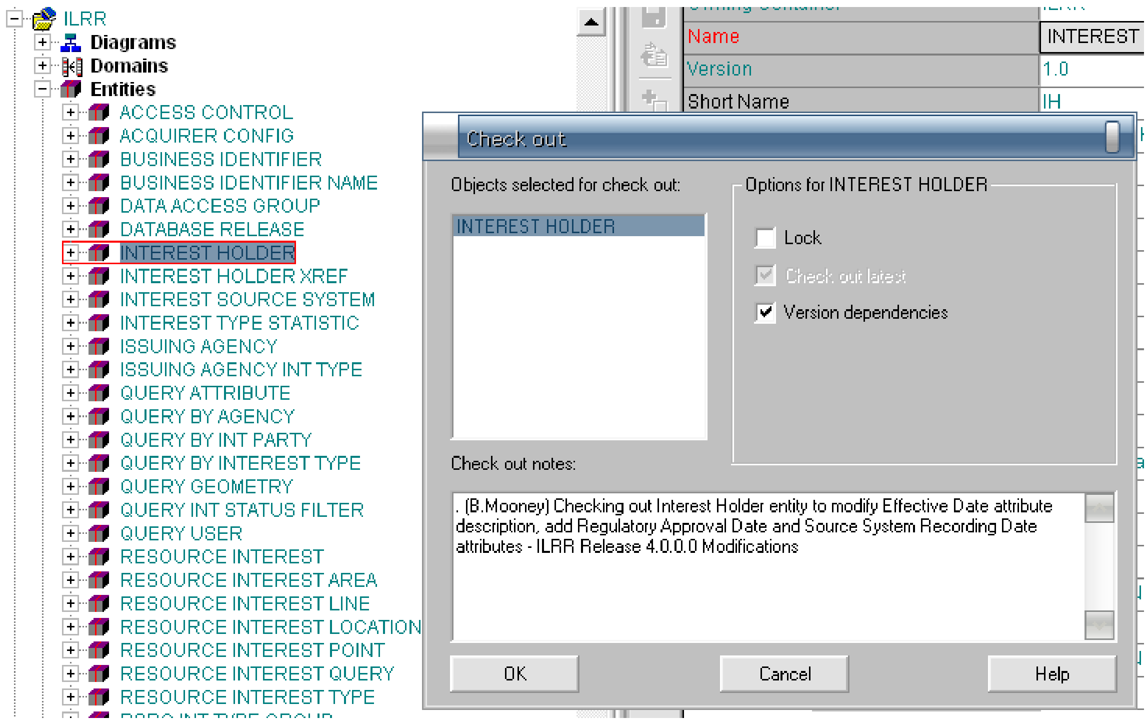
- **Developers do not check in objects,** that is the responsibility of DA. The only exception is when a versioned object is to be removed from a checked out container. See 9.2

- Developers **never** use the command **check out contents**, it is reserved for DA or in the case where all objects are going to be updated as in a total new version of the Data Model is being developed (e.g. 1.0 to 2.0).

7.2 Contractor/Vendor responsibility

- The objects within the container (PAC elements) are the responsibility of the developer for checkout. The checkout notes must have the following information. The reason the object is being checked out, and the developer that is checking it out (e.g. . (G. Gale) updating column descriptions for CORS_PERSONS table). The Date is automatically supplied by Oracle.

- The checkout note should be in detail so that the DA can use them to quickly review any changes using a tool called "compare previous version". These checkout notes will also be used by DA or DBA for check in notes so they should be on an object basis, not a grouping of objects. *Sometimes an object may be checked-out with notes and some other issues arise that require changes or an object may be checked out when using a diagram tool that a developer is not aware of → In this case the note property of the PAC object can be used for additional changes that are not reflected in the original checkout notes or if they were missed.*



ILRR
- Diagrams
- Domains
- Entities
  - ACCESS CONTROL
  - ACQUIRER CONFIG
  - BUSINESS IDENTIFIER
  - BUSINESS IDENTIFIER NAME
  - DATA ACCESS GROUP
  - DATABASE RELEASE
  - INTEREST HOLDER
  - INTEREST HOLDER XREF
  - INTEREST SOURCE SYSTEM
  - INTEREST TYPE STATISTIC
  - ISSUING AGENCY
  - ISSUING AGENCY INT TYPE
  - QUERY ATTRIBUTE
  - QUERY BY AGENCY
  - QUERY BY INT PARTY
  - QUERY BY INTEREST TYPE
  - QUERY GEOMETRY
  - QUERY INT STATUS FILTER
  - QUERY USER
  - RESOURCE INTEREST
  - RESOURCE INTEREST AREA
  - RESOURCE INTEREST LINE
  - RESOURCE INTEREST LOCATION
  - RESOURCE INTEREST POINT
  - RESOURCE INTEREST QUERY
  - RESOURCE INTEREST TYPE

Name	INTEREST
Version	1.0
Short Name	IH

Check out

Objects selected for check out: INTEREST HOLDER

Options for INTEREST HOLDER
- ☐ Lock
- ☑ Check out latest
- ☑ Version dependencies

Check out notes:

. (B.Mooney) Checking out Interest Holder entity to modify Effective Date attribute description, add Regulatory Approval Date and Source System Recording Date attributes - ILRR Release 4.0.0.0 Modifications

- If a new object is going to be added or any physically implemented objects properties are going to change, it is strongly recommended to first check out the OPERATIONAL grouping under the Oracle Database tab.

- To avoid problems when checking out objects in a versioned repository it is important that the correct tool be used for the checking out process.
  1) Entities use the (Repository Object Navigator (RON).

  2) Materialized views, views, Tables use the RON preferably or the Design Editor.

E.g. of good check out note

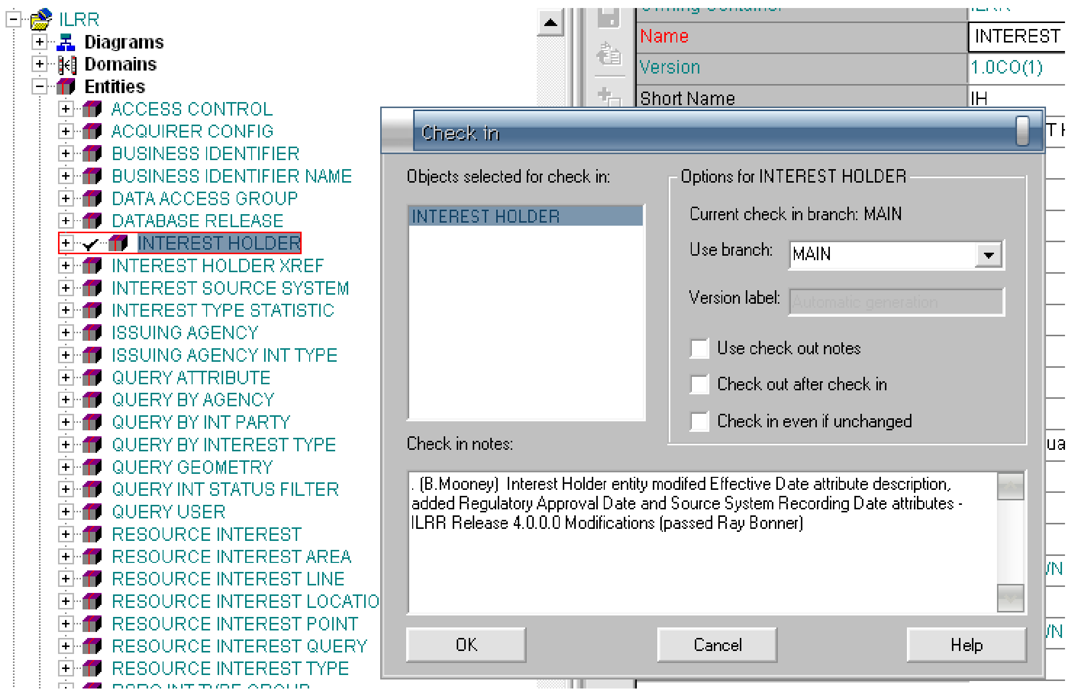
. (G. Gale) Checking out entity BATHYMETRIC_MAPSHEET to change attribute name SHEET NO to BATHYMETRIC MAPSHEET NO and BATHYMETRIC_MAPSHEET entity name to BATHYMETRIC MAPSHEET as per sector convention.

7.3 DA responsibility

- SCM Administrator DA will check the notes property and the checkout note and compose the appropriate check in note at the time of review.

- The Data Model container is the responsibility of SCM Administrator DA and is checked out with appropriate check out notes. The checkout notes should have the following information. The reason the container is being checked out, who it is being checked out for (i.e. the business area and developer (vendor), the DA that is checking it out and the Patch number that will be used in application delivery. The Date is automatically supplied by Oracle and is not necessary. Containers are the responsibility of DA.

- The QA DA is responsible for check-in of logical model components and uses the developer's checkout notes when checking in. This is done on an object basis after DA has reviewed and approved of any changes. The DA will leave the object in a checked out state until any change request has been finished. When checking in the "Use check out notes" box is checked. After the note is modified the check mark disappears.


ILRR
- Diagrams
- Domains
- Entities
  - ACCESS CONTROL
  - ACQUIRER CONFIG
  - BUSINESS IDENTIFIER
  - BUSINESS IDENTIFIER NAME
  - DATA ACCESS GROUP
  - DATABASE RELEASE
  - ✓ INTEREST HOLDER
  - INTEREST HOLDER XREF
  - INTEREST SOURCE SYSTEM
  - INTEREST TYPE STATISTIC
  - ISSUING AGENCY
  - ISSUING AGENCY INT TYPE
  - QUERY ATTRIBUTE
  - QUERY BY AGENCY
  - QUERY BY INT PARTY
  - QUERY BY INTEREST TYPE
  - QUERY GEOMETRY
  - QUERY INT STATUS FILTER
  - QUERY USER
  - RESOURCE INTEREST
  - RESOURCE INTEREST AREA
  - RESOURCE INTEREST LINE
  - RESOURCE INTEREST LOCATIO
  - RESOURCE INTEREST POINT
  - RESOURCE INTEREST QUERY
  - RESOURCE INTEREST TYPE

Name	INTEREST
Version	1.0CO(1)
Short Name	IH

Check in

Objects selected for check in: INTEREST HOLDER

Options for INTEREST HOLDER

Current check in branch: MAIN

Use branch: MAIN

Version label: Automatic generation

☐ Use check out notes

☐ Check out after check in

☐ Check in even if unchanged

Check in notes:

. (B.Mooney) Interest Holder entity modifed Effective Date attribute description, added Regulatory Approval Date and Source System Recording Date attributes - ILRR Release 4.0.0.0 Modifications (passed Ray Bonner)

- DA is responsible for check-in of physical model components and uses the developer's checkout notes when checking in. This is done on an object basis after DBA has reviewed and approved of any changes. The DA will leave the object in a checked out state until any change request has happened and verified by the DBA. DBA has to contact DA to do the final check-in.

- For new objects in a versioned Data Model container or a non versioned Data Model container, DBA and DA will have to make their own check-in notes following this example.
  "(G. Gale) added entity BATHYMETRIC_MAPSHEET– (approved R. Hoffner)"

8 How EXTENDED COPY works for non versioned and versioned objects.

8.1 Setup Process for copying an existing object

- The process for copying existing objects from other Data Model containers may occur during the logical or physical model development or on the developers or business areas own initiative. The developer or business area has intranet access to the Oracle Repository Object Browser (ROB) and can query the CSD Corporate Production Repository for common objects. A list of Corporate and Data Model shared objects can be found in the repository in the containers prefixed with “CORP_” along with a definition of the differences and their original location. They will stay exposed in WA_WIP_DEVELOPMENT. These stub models are in the process of continual update as corporate and shared objects are being identified.

- While in the modelling process, it is the responsibility of the developer to send a request to the DA to expose (grant access) the desired Data Model container and objects requested in a checked out state in WA_WIP_DEVELOPMENT.

8.2 Most common reason for copies

- At the logical and physical level, all attributes or columns of corporate or shared objects are duplicated, as sharing between Data Models in any corporate sector repository is not permitted. This restriction is based on the degree of difficulty associated with importing and exporting of shared Data Models. (Also see document on corporate code sharing - Corporate Code Usage).

- Where reference to entities (other than corporate codes) in other Data Models is required, Data Administration will accept the concept of stubbing entities. By stubbing, only the entities’ name and primary key are copied from the source Data Model. The description of the entity is then modified to include a statement that includes the name of the source data model container and that it is deemed as a stub. Example “The object AIR PHOTO is a stub that was sourced from the data model container APS.”

- When materialized views for a Data Model that spans multiple business areas are required, a copy of the table definitions from the source Data Model containers to the working data model container is made. Materialized views across Data Model containers are not permitted.

8.3 Method

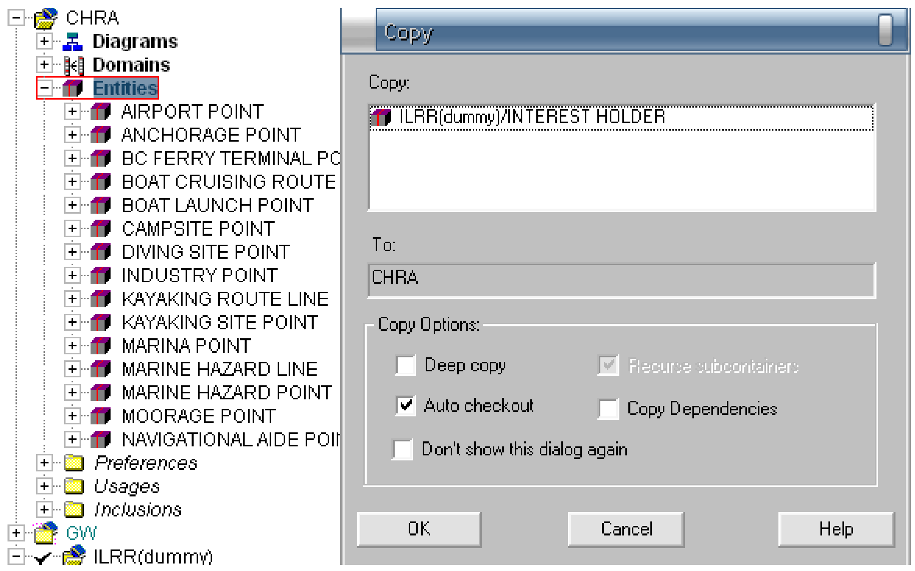
- If the Data Model is versioned, both the Data Model container and the object or group of objects needs to be in a checked out state, to copy them to a new container. That is, the PAC element must be in a checked out state. If a PAC is checked out the SAC elements are checked out by default. For non versioned objects this is not a problem because there state is checked out.

- In order to copy an object from one Data Model container to another, the Data Model folders must be in the same work area. In our case the work area WA_WIP_DEVELOPMENT is where copying takes place.

- When doing a extended copy it is advised not to do a deep copy (unless you know what is happening here), for this will want to maintain referential integrity and DA will need to checkout any and every object besides the one you want to copy that has reference to it. (In most all cases undo the checkboxes - deep copy and copy external references). In most cases you only want to create an entity with the same name and then copy its attributes. Note: you should quality check data formats and lengths for this seems to not always copy across the same. Almost never have the check box checked, copy dependencies.

Example screens for copying an entity where we do not get external references to resolve:

8.3.1 Copy the entity name



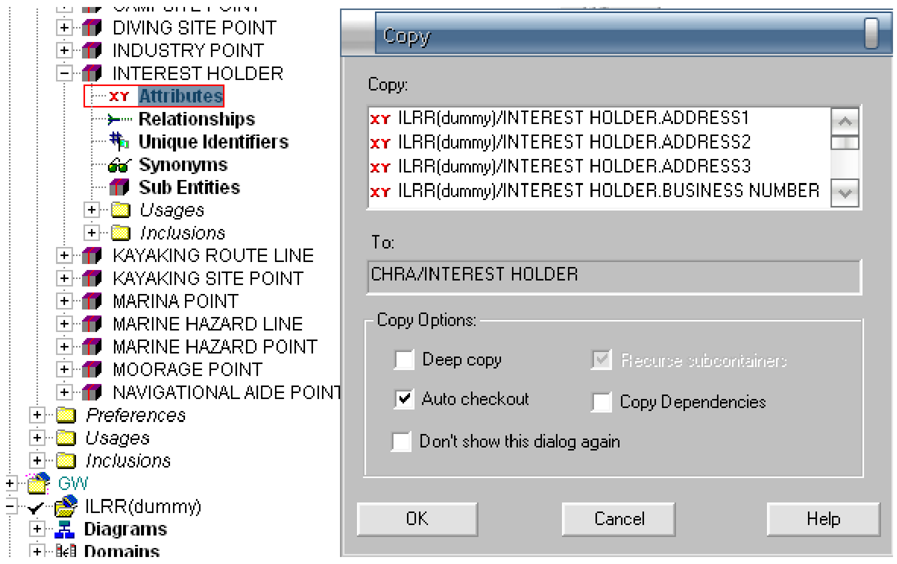
Copy: ILRR(dummy)/INTEREST HOLDER

To: CHRA

Copy Options:

- [ ] Deep copy
- [✓] Recurse subcontainers
- [✓] Auto checkout
- [ ] Copy Dependencies
- [ ] Don't show this dialog again

8.3.2 Copy the entity attributes



Copy:

ILRR(dummy)/INTEREST HOLDER.ADDRESS1
ILRR(dummy)/INTEREST HOLDER.ADDRESS2
ILRR(dummy)/INTEREST HOLDER.ADDRESS3
ILRR(dummy)/INTEREST HOLDER.BUSINESS NUMBER

To:

CHRA/INTEREST HOLDER

Copy Options:

- [ ] Deep copy
- [✓] Recurse subcontainers
- [✓] Auto checkout
- [ ] Copy Dependencies
- [ ] Don't show this dialog again

8.3.3 Manually remake any relationships and unique keys etc.

8.4 Stubbing

Stubbing is a process of copying that replaces sharing or referencing in the Ministries corporate repository so external references do not occur. Stubbing is the process of copying a entity, table, view, materialized view, plsql definition, or object type from one repository application container to another. If one of these objects is needed in a design it is copied as a stub to show reference only. When a PAC is stubbed then only the mandatory SAC elements are required in the data model and the definition (description) of the stubbed PAC has to include a statement that points to the owning container for the referenced PAC object.

9 DELETE, FORCE DELETE, PURGE and FORCE PURGE.

-- PAC vs. SAC elements

NOTE: All developers get the DELETE privilege when working in WA_WIP_DEVELOPMENT. The PURGE privilege is only give on request.

Oracle Repository enables you to manage the removal of repository objects in a carefully controlled manner. Many repository objects include references to other objects that exist in the same repository. When you delete a repository object, you need to specify what happens to any objects that reference the deleted object. If an object is under version control, you will need to specify whether you want to delete an object version (that is, remove that version from a particular container) or purge one or more versions of an object (that is, remove them from the repository altogether). Entities cannot be totally removed from an Data Model Container without removing any reference to them. But they may be removed from the checked out container.

Note: if you create a new version of a object and check it in. you cannot remove it from the checked out container without checking the container in.

See the accompanying document on the web named: "About deleting and purging repository objects.pdf" for an understanding on SAC_PAC elements and the differences between DELETE, FORCE DELETE, PURGE and FORCE PURGE

All external references must be resolved when removing an object from the repository whether it is a versioned or non-versioned container. It is up to the **developer** to resolve these external references in the Data Model that they are working with. It is best to use force delete for SAC elements. FORCE DELETE and FORCE PURGE can be viewed as a synonym for cascade delete in PL/SQL. But even though you FORCE DELETE an object it does not mean that all references are removed. There are differences between a reference (or association) and objects like (PAC) or (SAC) elements.

9.1 Deleting non versioned objects

To cleanly remove non-versioned objects from the repository (RON)

- all references to that object should be deleted (check under usages)
- the elements (SAC) that make up the object should be force deleted
- Delete the object.

One may use force delete, if knowledgeable, but the commands delete and delete association are cleaner.

9.2 Deleting versioned objects

- To cleanly remove a versioned object from its owning container in the RON, the object must be checked out and all references should be deleted to that object, from any other objects (which mean they must be checked out also)
- the SAC elements are then deleted for said object,
- the object in question that is being removed, that was checked out in this process, must then be checked in. **(this is only time a object is to be checked in by a developer)**

- At this point, you can DELETE the object from the version of the container, which is checked out.
- DA should be notified to check the lost & found and refresh the work area.

WORK AREA and container privileges are not given to the developer. If the developer has correctly removed the references to the object in question it will be removed from the repository container. If not it will end up in the lost & found.

For some understanding on the process for removing objects from the oracle repository see the document "Removing Application containers from the Repository.doc. This document provides some insight into how complicated it is to remove a container from the repository once it is versioned.

10 EXTERNAL REFERENCES

- It is the responsibility of the Developer to query the Data Model container for external references.

- A VALID external reference is a reference to a shared object in another container that exists and is being used by the child container that was queried.

- An INVALID external reference is a pointer to something that is not there. All external references in the Data Model folder that have pointers to the folder that was queried itself are invalid and must be resolved. INVALID external references mostly occur when objects are improperly removed, copied or renamed with regards to a Data Model container, or when diagrams have not been updated after changes have occurred.

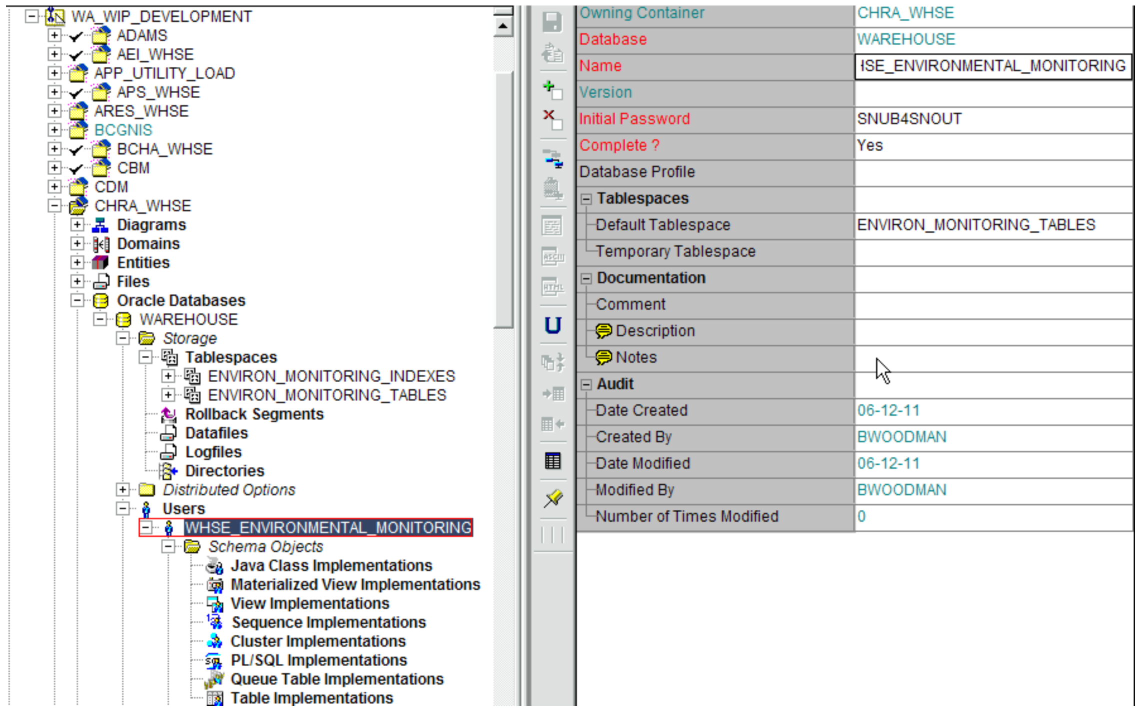
- At this time CSD does not promote sharing of objects. This might change, if at some time there is a truly shared working development repository. The only external references we have in the Corporate Repository are from legacy Data Models that have shares.

11 TRANSFORMING Entities to Tables

The logical model is transformed to the physical model after these steps:

- The entities have been passed by the QA DA. Entities for this procedure include that the application prefix be found in the entity short name property and the entity plural name property.
- The Storage definitions are defined to standard. Note: they are not required for a 10G database.
- The Oracle Database tab has been created with the at least the schema “user” defined to standard.
- The Table spaces are defined to standard.

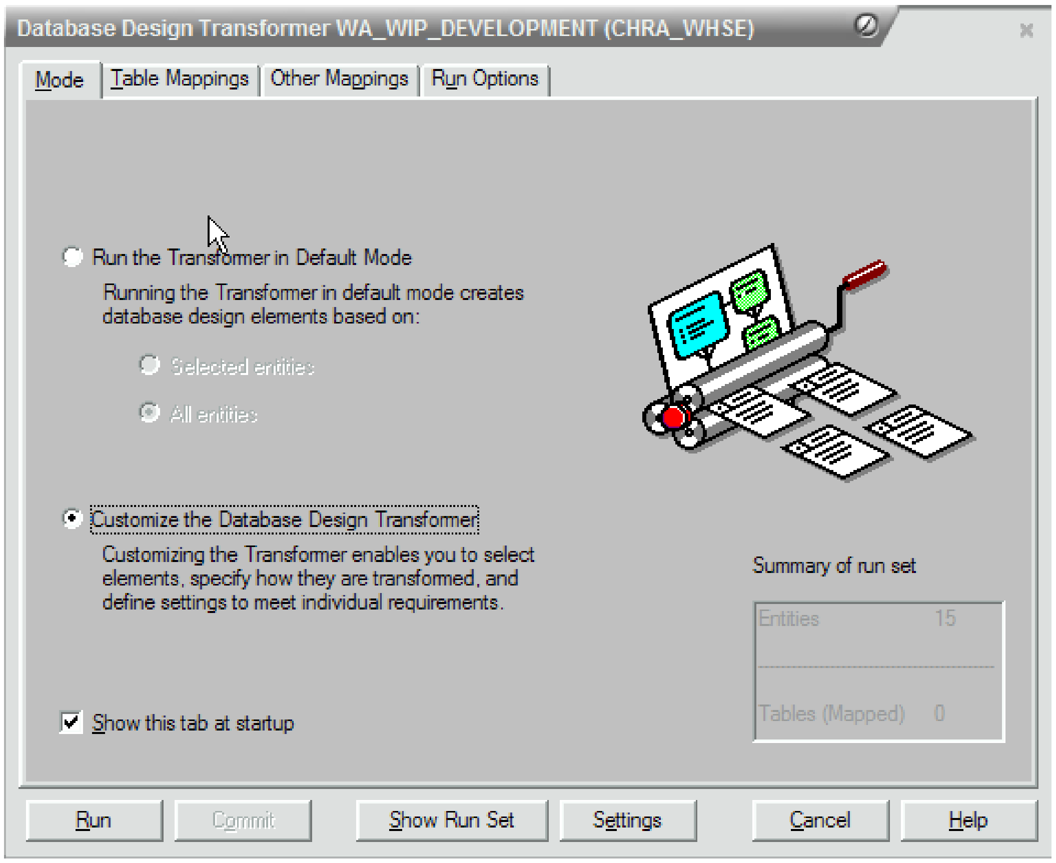
These last two are above defined in the following place in designer:



```
WA_WIP_DEVELOPMENT
  ADAMS
  AEI_WHSE
  APP_UTILITY_LOAD
  APS_WHSE
  ARES_WHSE
  BCGNIS
  BCHA_WHSE
  CBM
  CDM
  CHRA_WHSE
    Diagrams
    Domains
    Entities
    Files
    Oracle Databases
      WAREHOUSE
        Storage
          Tablespaces
            ENVIRON_MONITORING_INDEXES
            ENVIRON_MONITORING_TABLES
          Rollback Segments
          Datafiles
          Logfiles
          Directories
        Distributed Options
        Users
          WHSE_ENVIRONMENTAL_MONITORING
            Schema Objects
              Java Class Implementations
              Materialized View Implementations
              View Implementations
              Sequence Implementations
              Cluster Implementations
              PL/SQL Implementations
              Queue Table Implementations
              Table Implementations
```

Owning Container	CHRA_WHSE
Database	WAREHOUSE
Name	ISE_ENVIRONMENTAL_MONITORING
Version	
Initial Password	SNUB4SNOUT
Complete ?	Yes
Database Profile	
Tablespaces	
Default Tablespace	ENVIRON_MONITORING_TABLES
Temporary Tablespace	
Documentation	
Comment	
Description	
Notes	
Audit	
Date Created	06-12-11
Created By	BWOODMAN
Date Modified	06-12-11
Modified By	BWOODMAN
Number of Times Modified	0

Once all the above conditions are met the physical model can be transformed from here. To transform the logical model to a physical model highlight the application container to be transformed, select the command “Database Design Transformer” located on the pull down menu Utilities→Designer side menu. The Transformer starts:



Database Design Transformer WA_WIP_DEVELOPMENT (CHRA_WHSE)

Mode | Table Mappings | Other Mappings | Run Options

( ) Run the Transformer in Default Mode

Running the Transformer in default mode creates database design elements based on:

( ) Selected entities

(•) All entities

(•) Customize the Database Design Transformer

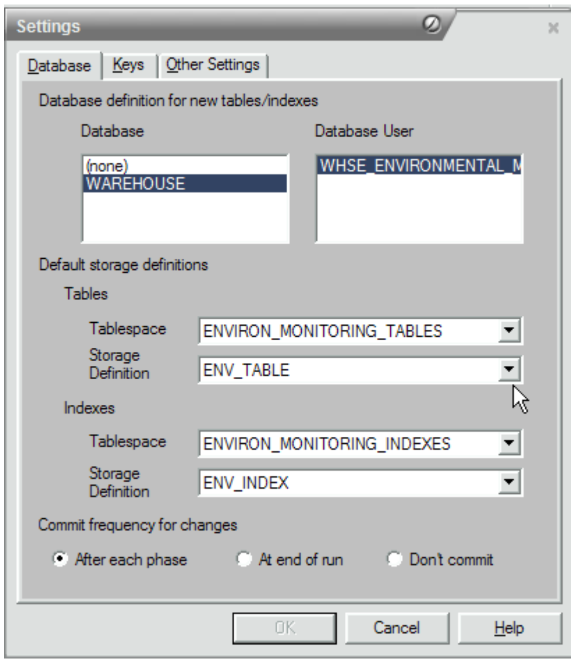
Customizing the Transformer enables you to select elements, specify how they are transformed, and define settings to meet individual requirements.

[✓] Show this tab at startup

Summary of run set

Entities	15
Tables (Mapped)	0

The screen above can stay default as is. The “Table Mappings”, Other Mappings”, and “Run Options” are at the discretion of the developer. The “Settings” button has some standard setting for the Ministry. They are shown in the following screen shots. First the “Database” tab is configured like:


Settings

Database | Keys | Other Settings

Database definition for new tables/indexes

Database: (none), WAREHOUSE (selected)

Database User: WHSE_ENVIRONMENTAL_M

Default storage definitions

Tables

Tablespace: ENVIRON_MONITORING_TABLES

Storage Definition: ENV_TABLE

Indexes

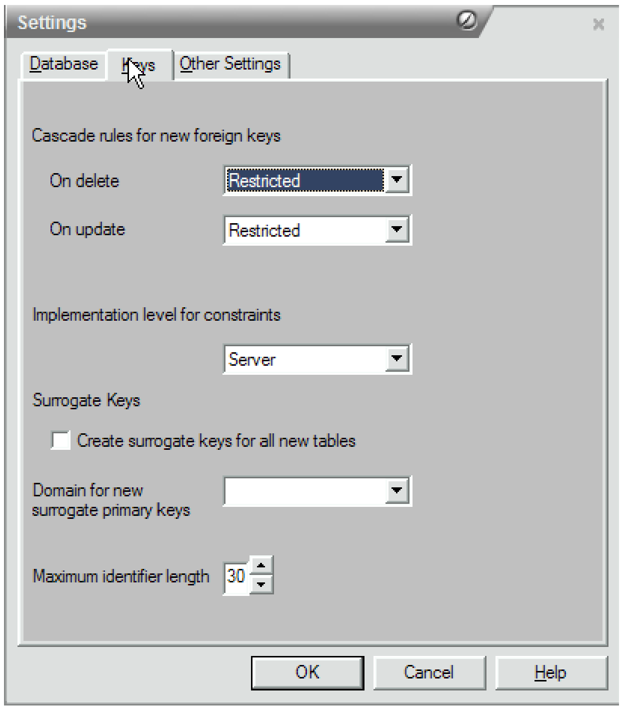
Tablespace: ENVIRON_MONITORING_INDEXES

Storage Definition: ENV_INDEX

Commit frequency for changes

(●) After each phase ( ) At end of run ( ) Don't commit

The table spaces and storage definitions for tables and indexes are added form the drop downs since they were defined earlier. The Database and Database User that was defined is highlighted not "(none)". Second the "Keys" tab is configured like:


Settings

Database | Keys | Other Settings

Cascade rules for new foreign keys

On delete: Restricted

On update: Restricted

Implementation level for constraints: Server

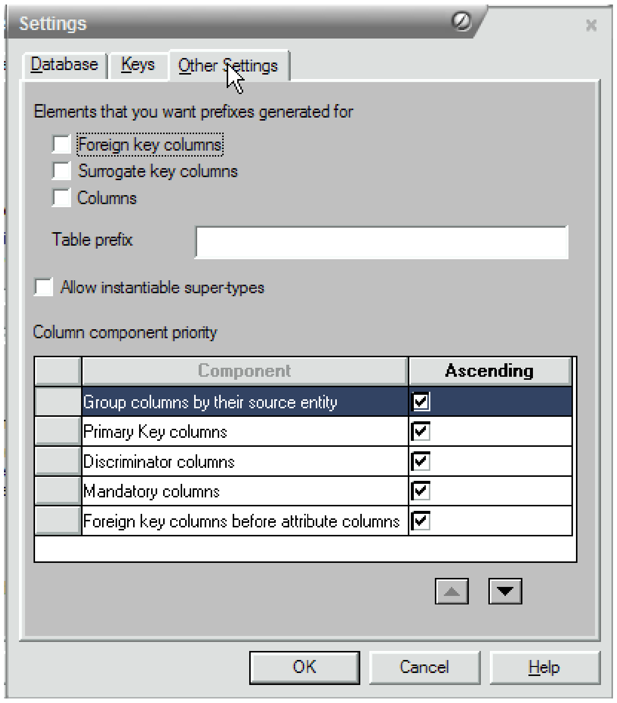
Surrogate Keys

[ ] Create surrogate keys for all new tables

Domain for new surrogate primary keys:

Maximum identifier length: 30

Third the “Other Settings” tab is configured like:



Elements that you want prefixes generated for

- [ ] Foreign key columns
- [ ] Surrogate key columns
- [ ] Columns

Table prefix

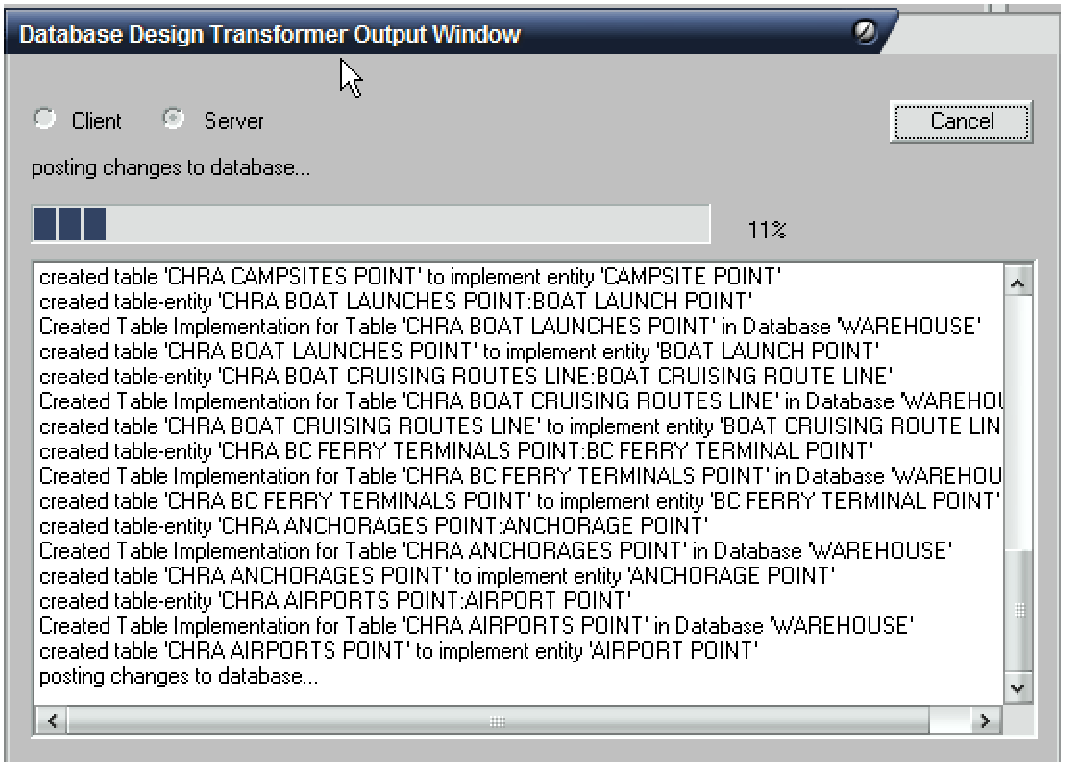
- [ ] Allow instantiable super-types

Column component priority

	Component	Ascending
	Group columns by their source entity	✓
	Primary Key columns	✓
	Discriminator columns	✓
	Mandatory columns	✓
	Foreign key columns before attribute columns	✓

Note: There is no prefixing since it was done at the logical level in the entity short and plural property.

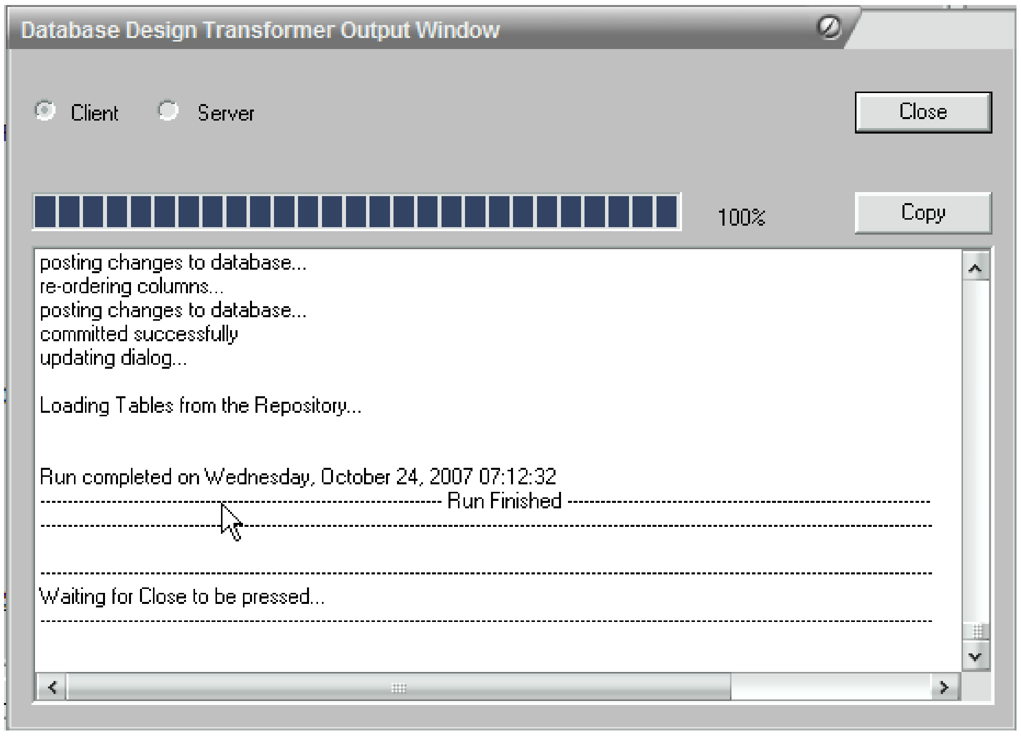
Selecting “OK” on the settings tab, requests that these be saved to the database as default. Accept then select the “Run” button on the Database Design Transformer screen. It will Process:



```
posting changes to database...
```

```
created table 'CHRA CAMPSITES POINT' to implement entity 'CAMPSITE POINT'
created table-entity 'CHRA BOAT LAUNCHES POINT:BOAT LAUNCH POINT'
Created Table Implementation for Table 'CHRA BOAT LAUNCHES POINT' in Database 'WAREHOUSE'
created table 'CHRA BOAT LAUNCHES POINT' to implement entity 'BOAT LAUNCH POINT'
created table-entity 'CHRA BOAT CRUISING ROUTES LINE:BOAT CRUISING ROUTE LINE'
Created Table Implementation for Table 'CHRA BOAT CRUISING ROUTES LINE' in Database 'WAREHOU
created table 'CHRA BOAT CRUISING ROUTES LINE' to implement entity 'BOAT CRUISING ROUTE LIN
created table-entity 'CHRA BC FERRY TERMINALS POINT:BC FERRY TERMINAL POINT'
Created Table Implementation for Table 'CHRA BC FERRY TERMINALS POINT' in Database 'WAREHOU
created table 'CHRA BC FERRY TERMINALS POINT' to implement entity 'BC FERRY TERMINAL POINT'
created table-entity 'CHRA ANCHORAGES POINT:ANCHORAGE POINT'
Created Table Implementation for Table 'CHRA ANCHORAGES POINT' in Database 'WAREHOUSE'
created table 'CHRA ANCHORAGES POINT' to implement entity 'ANCHORAGE POINT'
created table-entity 'CHRA AIRPORTS POINT:AIRPORT POINT'
Created Table Implementation for Table 'CHRA AIRPORTS POINT' in Database 'WAREHOUSE'
created table 'CHRA AIRPORTS POINT' to implement entity 'AIRPORT POINT'
posting changes to database...
```

And then finish. Select “Close”:



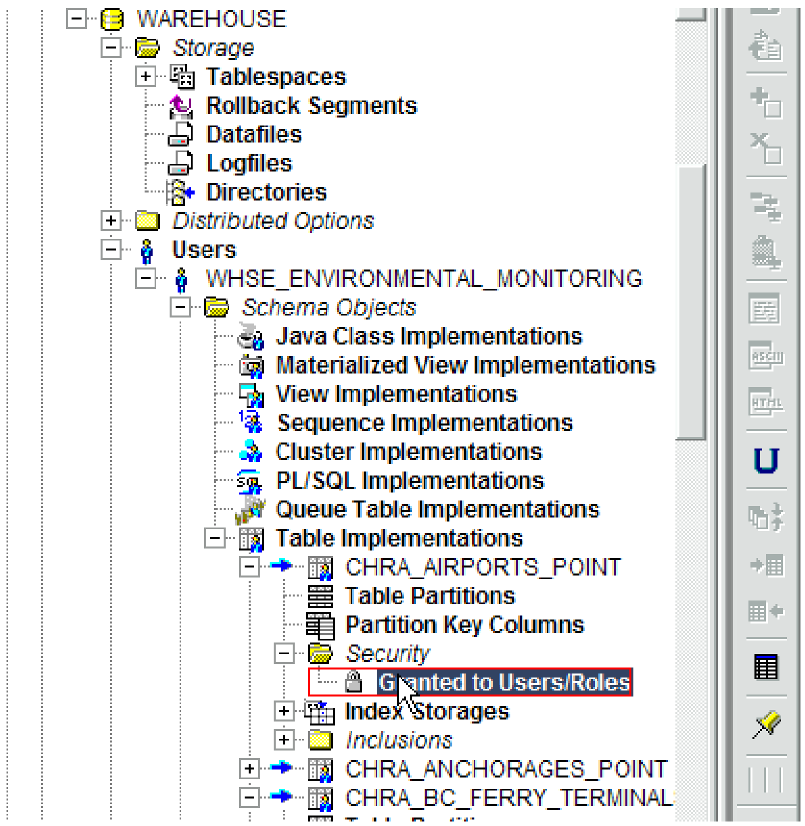
```
posting changes to database...
re-ordering columns...
posting changes to database...
committed successfully
updating dialog...

Loading Tables from the Repository...


Run completed on Wednesday, October 24, 2007 07:12:32
---------------------------------------------- Run Finished ----------------------------------------------
----------------------------------------------------------------------------------------------------------

----------------------------------------------------------------------------------------------------------
Waiting for Close to be pressed...
----------------------------------------------------------------------------------------------------------
```

The Oracle Database tab WAREHOUSE for the user WHSE_ENVIRONMENTAL_MONITORING will be populated with the implementations of the tables. All that need be done is add the roles to the user to define the Security on the objects (in this case tables). As shown below:

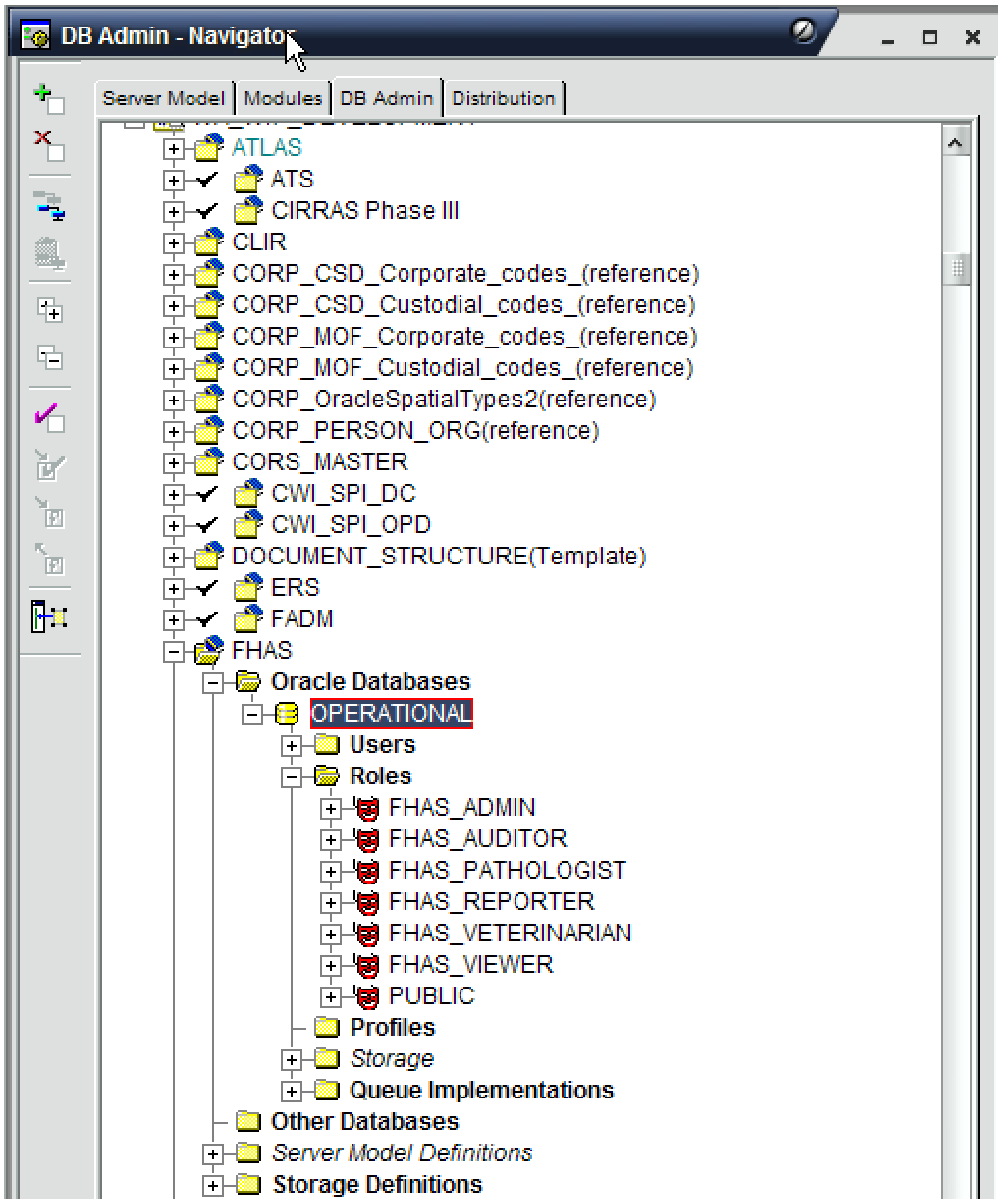


12 DATABASE DESIGN GENERATION

The procedure for generating the DDL follows after the Physical Data model has been completed and approved.

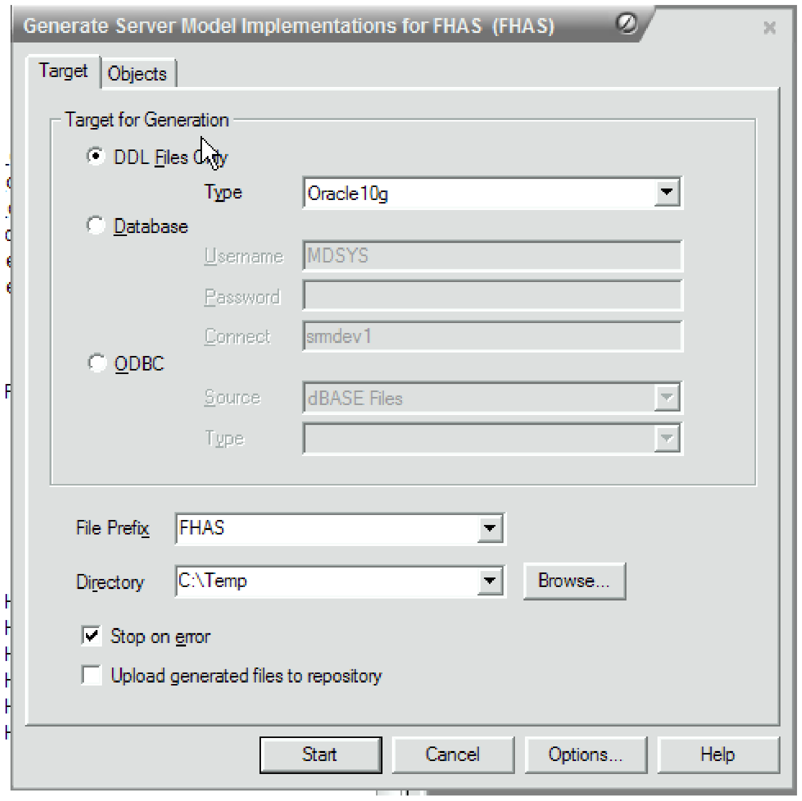
> Before generation, make sure the roles are defined and the “granted to User/Roles” security input for the objects being implemented (tables, views, materialized views, plsql, and sequences).

To start generation open up the Design Editor: In the navigator select the “DB Admin” tab. From expand the application to generate. On the pull down menu “Generate” there are two commands “Generate Database from Server model” and “Generate Database Administrative Objects”. “Generate Database from Server model” will generate all the database objects with grants, etc. and “Generate Database Administrative Objects” will generate the users and table spaces. (Note: the later scripts will end up being run by Sector DBA.)



```
ATLAS
✓ ATS
✓ CIRRAS Phase III
CLIR
CORP_CSD_Corporate_codes_(reference)
CORP_CSD_Custodial_codes_(reference)
CORP_MOF_Corporate_codes_(reference)
CORP_MOF_Custodial_codes_(reference)
CORP_OracleSpatialTypes2(reference)
CORP_PERSON_ORG(reference)
CORS_MASTER
✓ CWI_SPI_DC
✓ CWI_SPI_OPD
DOCUMENT_STRUCTURE(Template)
✓ ERS
✓ FADM
FHAS
  Oracle Databases
    OPERATIONAL
      Users
      Roles
        FHAS_ADMIN
        FHAS_AUDITOR
        FHAS_PATHOLOGIST
        FHAS_REPORTER
        FHAS_VETERINARIAN
        FHAS_VIEWER
        PUBLIC
      Profiles
      Storage
      Queue Implementations
  Other Databases
  Server Model Definitions
  Storage Definitions
```

For a “Generate Database from Server model”, the Implementations screen comes up. On the Target tab key in the “File prefix”, the rest are standard:



Generate Server Model Implementations for FHAS (FHAS)

Target | Objects

Target for Generation

- (●) DDL Files Only
  - Type: Oracle10g
- ( ) Database
  - Username: MDSYS
  - Password:
  - Connect: srmdev1
- ( ) ODBC
  - Source: dBASE Files
  - Type:

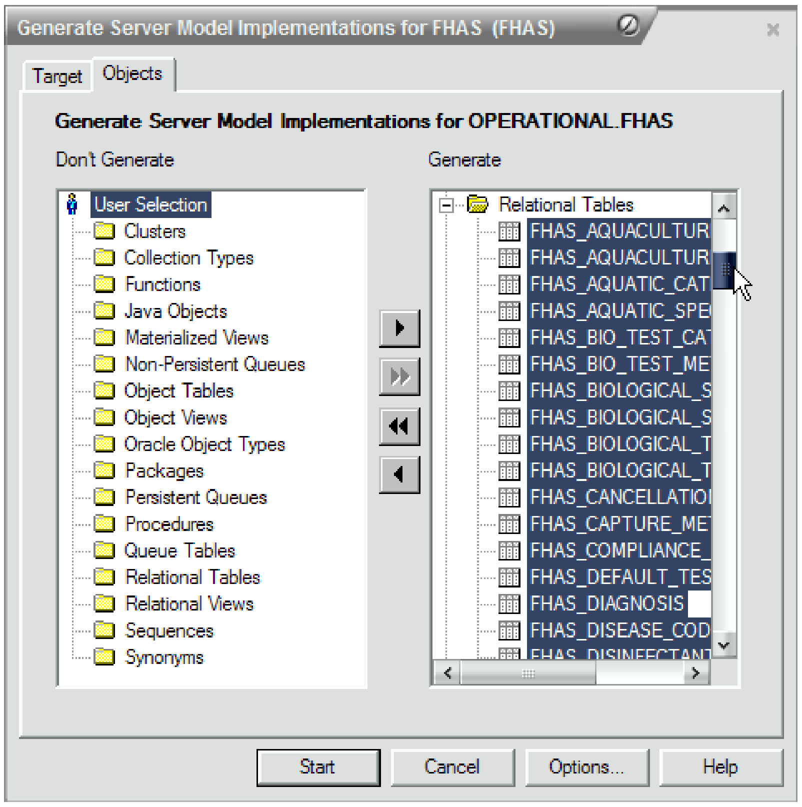
File Prefix: FHAS

Directory: C:\Temp

[✓] Stop on error

[ ] Upload generated files to repository

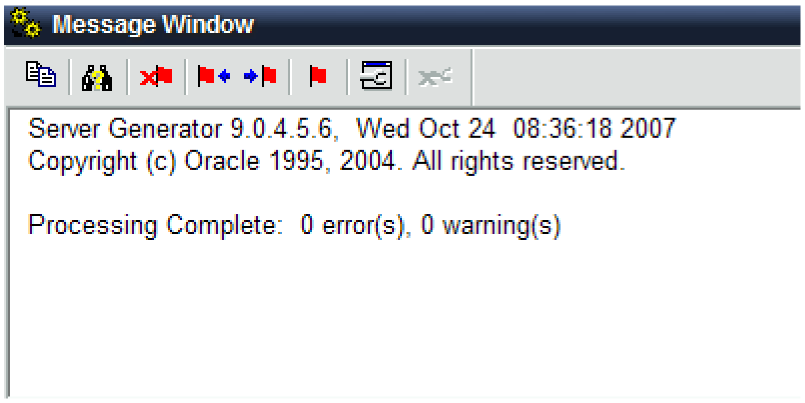
On the Objects tab key the objects are selected:



Generate Server Model Implementations for OPERATIONAL.FHAS

Don't Generate	Generate
User Selection	Relational Tables
Clusters	FHAS_AQUACULTUR
Collection Types	FHAS_AQUACULTUR
Functions	FHAS_AQUATIC_CAT
Java Objects	FHAS_AQUATIC_SPE
Materialized Views	FHAS_BIO_TEST_CA
Non-Persistent Queues	FHAS_BIO_TEST_ME
Object Tables	FHAS_BIOLOGICAL_S
Object Views	FHAS_BIOLOGICAL_S
Oracle Object Types	FHAS_BIOLOGICAL_T
Packages	FHAS_BIOLOGICAL_T
Persistent Queues	FHAS_CANCELLATIO
Procedures	FHAS_CAPTURE_ME
Queue Tables	FHAS_COMPLIANCE_
Relational Tables	FHAS_DEFAULT_TES
Relational Views	FHAS_DIAGNOSIS
Sequences	FHAS_DISEASE_COD
Synonyms	FHAS_DISINFECTANT

Press “Start” and a Message window will come up till the process is complete:



```
Server Generator 9.0.4.5.6,  Wed Oct 24  08:36:18 2007
Copyright (c) Oracle 1995, 2004. All rights reserved.

Processing Complete:  0 error(s), 0 warning(s)
```

The DDL generated is ready to submit for the application delivery scripting.

13 PRINCIPLES when REVERSE ENGINEERING a database

For document procedures, see Designer Reverse Engineering

Following Sector standards, reverse engineering of a operational Oracle database schemas into an Oracle Designer physical data model, requires but is not limited to the following:

- Liaise with Sector Data Administrators and Database Administrators as required
- Reverse engineer the physical data model from the Ministry database schema, (or export thereof) into a designer repository. The latter must be version compatible with the Ministry Corporate Designer Repository at time of delivery.
- Ensure all implemented database objects are included in the model
- Bring the physical model up to current standard where a generated and implemented change would not cause application failure.
  (For example, object name changes would be exempt from standards since such a change would cause an application failure, whereas descriptions and schema diagrams are in scope since they are passive to the application).
- Interact with business knowledge expert(s) for the collection and/or review of data descriptions
- Deliver an Oracle Designer Data Model export file (.dmp) of the model, including PDF's of the relevant reports and diagrams as specified in the Sector standards. (Requirement unless working in one of the Sector's model repositories)
- Basic logging of observed anomalies, limitations, or deficiencies as appropriate. To be delivered as a Microsoft Word document.

14 CONCLUSION

For any developers working in a Natural Resource Sector Corporate Oracle repository, in summary

- No checking in of objects, unless removing those objects from a Data Model container.
- No container privileges granted
- Must use check out notes, in the format described above
- Delete with caution – copy with caution
- Data model container external references must be resolved and removed by vendor.

Repository access is granted on notification by BPM.