Corporate Services for the Natural Resource Sector

Information Management Branch

Data Naming Standards for the Natural Resource Sector

Last Updated:	April, 2013
Version:	1.0.1
Document:	NRS_Data_Naming_Standards_1.0.1

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 • Guide S19 – Data Naming Standards for System Development (Ministry of Forests) • Naming and Describing Standards (V 2.4.1) (MoE/MAL/EAO/ILMB) • Programming Standards (Ministry of Small Business, Technology, & Economic Development and Ministry of Energy Mines and Petroleum Resources) – **Obsolete December 31, 2011**	Jeremey Janzen (Forests Author) Randy Hoffner (MoE/MAL Author) Kelly McNulty (Editor)	December 2011
1.0	Consolidated multiple ministry standards where standards were the same. Did not consolidate where standards deviated.	Kelly McNulty	January 2012
1.0.1	Change occurrences of Data Administration to Data Architecture. Updated links to reflect new NRS site.	Bill Holland	April 2013

Contents

1 Introduction

This document was created by the Data Architecture/Data Management section of the Information Management Branch (IMB) as an initial consolidation and refinement of naming and describing standards for the Natural Resource Sector (NRS) ministries. These ministries are the Ministry of Aboriginal Relations and Reconciliation (MARR), Ministry of Agriculture (AGRI), Ministry of Energy and Mines (MEM), Ministry of Environment (MoE), and Ministry of Forests, Lands and Natural Resource Operations (MFLNRO).

As the sector continues to collaborate, the Data Architecture/Data Management section in the Information Management Branch will look for opportunities to move toward a single set of standards if that is the best solution.

1.1 Purpose & Scope

The purpose of this guide is to outline the various data naming and describing standards used within the NRS ministries and to provide one location where the stakeholders can find this information. Standards listed in this guide depict the current state of naming and describing related standards, and this guide has been formatted to display existing standards by ministry for ministries that had a data naming and describing standard before NRS consolidation.

If you have any questions about this standard for the NRS, see one of the Data Architecture staff or Database Administration staff in the Information Management Branch.

1.2 How to Use this Guide

Please be advised that this guide contains a listing of data naming and/or describing standards specific to each ministry that had a standard within the NRS.

In application development projects, where data naming changes occur, the project team will conduct a technical White Board Session to determine which standards will be used.

As the IMB Data Architecture/Data Management section works with our stakeholders to build a consolidated guide for all NRS applications, it is likely that existing applications will use the same standards (if it still exists) and any net new sector applications will use standards determined by the project team.

1.3 Intended Audience

The intended audience for this document are analysts and designers who have some knowledge of the techniques and procedures involved in information engineering methodology in general and in data modelling in particular. Specifically all developers who create data components that support the applications in the Sector. They will also be used by data architects/database administrators to review relational logical-physical models and Class data models:

- Developers
- IMB Staff (eg; Data Architects, Database Administrators)
- Business area staff (eg; Data Standards Managers, Application Administrators)

1.4 Items Audience Must Understand

Audience Member	Component of Guide
Business Portfolio Managers	SDLC process and how the data naming standards fit into the overall project
Data Designers (Data Modellers / Application Developers)	Must understand the designated standards for data naming being used in projects they are involved. Physical Naming standards will vary depending on delivery environment. Data Architecture / Database Administration staff will guide this decision.
Data Architects / Database Administrators	Will use these standards for project logical data model and data naming reviews.
Database Administrators	Will use these standards for project physical data model and data naming reviews.

1.5 Related Standards and Documents

Name	**Description**
Systems Development Life Cycle (SDLC)	http://www.nrs.gov.bc.ca
NRS Data Modelling Standards	http://www.nrs.gov.bc.ca/sdlc/NRS_standards/NRS_Data_Modelling_Standards.pdf
NRS Code Management Standards	http://www.nrs.gov.bc.ca/sdlc/NRS_standards/NRS_Code_Management_Standards.pdf
Guidelines for Best Practices in Data Management	http://www.for.gov.bc.ca/his/datadmin/DataMgmtRolesResp-2010Sept-FINAL-Approved.pdf

2 The Need for Naming Standards

The size of the Natural Resource Sector, its applications, and the high degree of information exchange between all parts of the sector dictates the need for highly organized systems development. The names used to identify facts about business objects, concepts, and events are critical to the general understanding and knowledge of sector staff. Using good naming standards that are easy to follow, systems designers can produce products that do not introduce ambiguity or misunderstanding into the business. Without adherence to naming standards, confusion is created.

The creation of precise, logical names for the corporate data language is essential to the success of any organization.

Data naming Standards were written and are used to develop consistent names for data components and for the development and implementation of systems in the sector.

When naming standards are used, any person working with these data components can make intelligent assumptions on the nature of the data object, and can also be assured that the likelihood of synonyms and duplicate names is reduced. Developers' efforts can be focussed on developing systems, and not on creating another way of naming things.

Naming Standards reduce the probability for misinterpretation when searching data as well as aid in the use and quality of the data. Naming Standards are also important in the quality management of data.

This document presents a convention for naming data and data related components that is precise, logical, and conforms to Sector systems' development methodology. The techniques described are used throughout the industry. The effectiveness of these conventions becomes very clear during a systems development project, when attempting to develop the physical data model design using the initial logical data model.

2.1 Purpose

This standard explains and illustrates the form, standards, and conventions to be used for naming logical business entities and attributes and physical tables and columns during systems development projects.

The intent of this standard is to ensure that effort in development is addressed to the topic system, and that objects are consistently recognised and referred to. In addition, migrations of data from one system to another are possible and the integration of systems is easier.

Particularly, it provides a platform for ensuring data sharing and consistency within the organization(s). It improves communication for staff who manage data, developers and users.

Standards developed for descriptions and definitions ensure that the understanding and use of the data beyond its immediate context is accurate and valid.

There are two drivers for good names. The first is to aid in the recognition of the data component without external reference. The second is to remove ambiguity and avoid errors in management of the data.

For the deliverables produced, there are a variety of users, and these standards ensure there is an understanding beyond the immediate context of the business area and business application.

The goal of any development standard is the delivery of quality software solutions. In the case of the development of application systems, following these few simple rules can be the difference between the success and failure of a project.

Most parts of this guide are written specifically for relational systems environments, no matter what the vendor or product. However, when discussing physical table and column names, there will be vendor specific restrictions on the physical storage limit for names.

This document presents general standards for naming and describing data components such as Objects like Entities and Tables. If you have any questions, see the staff in the Data Architecture group. All logical and physical names must be approved by Data Architecture/Database Administration staff prior to any tables being built.

3 General Naming

3.1 Logical and Physical Data Model Names

For a more complete discussion on the differences between Logical and Physical Data Models, refer to the NRS Data Modelling Standards.

3.2 Logical Data Models

The purpose of the logical data model is to show the data that the application must store in order to satisfy business requirements. It also shows how this data is related. It is created without any specific computer environment in mind. No optimization for performance, data storage, or even application development is done (or is desired). The intent is to produce a purely logical view of the data required by the business area.

Logical model names are normally not concerned with physical storage restriction, so theoretically can be any length with no need for abbreviation. However, since the logical name is used to generate the physical table name and column names, a limit (varies by ministry) is recommended for entity names and data attribute names including spaces.

3.3 Physical Data Models

The purpose of the physical data model is to show how the data elements will be implemented and stored on the database. Physical models may vary from the logical model in that the physical model may introduce objects that do not contribute to meeting the business requirements of the application. These new objects may be created in order to speed response times, reduce storage requirements, ensure that the application fits within the physical limitations of the computing environment, improve maintenance turnaround, or for other reasons.

3.4 General Format of Data Names

Each entity name must be unique throughout a NRS ministry, and each attribute of an entity must be unique only within that entity. The logical model's entity-attribute combination follows through, with some limitations, to the physical model's table-column definition. In both cases, the following format for choosing names should be applied:

- use <noun> alone wherever possible (Eg. CONTRACT);
- use <adjective><noun>, or if needed <adjective><adjective><noun> to clarify meaning (Eg. LEASE CONTRACT or PURCHASE CONTRACT) but see next point;
- minimize the use of adjectives when they are not completely necessary (Eg. use CONTRACTOR, not CONTRACT EMPLOYEE, or use WAREHOUSE, not STORAGE BUILDING);

- use singular names to emphasize that the entity or attribute name (and corresponding definition) is a pattern for every instance of the object that is being defined (Eg. use CLIENT, not CLIENTS) - one way to think about it is that you are naming one single occurrence of the element; and
- separate each different word of the object name with a space (" ") this is used at the logical level and will be replaced with an underscore "_" at the physical level.
- Acronyms are acceptable and even encouraged where they are universally understood and accepted, especially where the acronym is more commonly used than the long form. Examples of accepted acronyms are: CITS, OAP, SIN. However, application acronyms are generally only accepted where the table is specific to only one application (now and in the future) and is more technical in nature, e.g. ECAS Audit Event.

3.5 General Structure of Data Names

Some general comments about the structure of entity and attribute names; names must be:

- meaningful;
- self-documenting (by looking at the name, the reader should have a good idea what the name means without having to read the description);
- derived from the business use or purpose;
- repeatable (different people from different areas of the ministry reading the name at different times must have the same understanding of what the name means -- a corporate-wide use of the name);
- Only use abbreviations or acronyms where no combination of names would produce an acceptable result. Avoid unless universally understood;
- Object (class/entity) and attribute names do not contain '_' (underscores);
- If the (class/entity) object represents a code table it has the suffix CODE or CD; and
- If the (class/entity) object represents a cross reference table it has the suffix XREF.

3.6 Logical Naming Standards and Structure

Logical data models are generated by identifying each entity, along with attributes, or characteristics, that the business records information about. An entity is a person, place, thing, event, or concept about which the ministry is interested and facts are gathered and recorded. The name used to define each entity must be unique within the enterprise. This is accomplished by placing qualifiers or quantifiers in front of the entity name

Example: Entity Names

don't use	LICENCE	use instead	FOREST LICENCE
don't use	PERMIT	use instead	CUTTING PERMIT
		or	RANGE TENURE PERMIT
		or	BURNING PERMIT

The general form of the Class/Entity Name is:
MODIFIERS : ENTITY NAME : MODIFIERS : DOMAIN NAME
Spaces delimit the words. The full name is in upper case.

For example, “Casual Employee Hire Date” the class/entity name would be CASUAL EMPLOYEE HIRE DATE.

3.6.1 Logical Data Model Names

The logical data model is used to define the application's data requirements, usually done at a project level. This model is not concerned with what kind of database system or computer system the application may run on; i.e. it is independent of the physical structure.

With the data modelling software available today, there is usually either no limit to the length of entity and attribute names, or the limit is generous -- for example, 40 characters. Therefore, logical data model designers must take advantage of this freedom and fully spell out all names, space permitting. The use of acronyms (Eg. SIN instead of Social Insurance Number) is the only exception.

In the logical model, all words are completely spelled out; no abbreviations unless absolutely necessary for emphasis or clarity.

Logical Data Model Naming Example

Entity	FOREST CLIENT
Attribute	CLIENT NUMBER
Attribute	CLIENT NAME
Attribute	LEGAL FIRST NAME
Attribute	BC DRIVERS LICENSE NUMBER

3.7 Attribute Naming

An attribute is a uniquely named characteristic of an (class/entity) so that for any single occurrence of the (class/entity), the named attribute associates one value with that occurrence. The attribute name must be unique within the (class/entity). Attribute names consist of a domain name (a common term that gives meaning to the (class/entity) object, such as date, user-id, name, or number) preceded by one or more modifiers. Modifiers are terms that define the attribute further or cause the attribute to be uniquely identified within the (class/entity) object data domain. They are included when judged to be essential to the identity of the attribute or to differentiate between attributes of the same class within the same (class/entity).

Unique key attributes that are surrogate keys must end in the suffix ID.

Unique key attributes that are surrogate often have the same name as the database table.

Unique key attributes for code tables have the same name as the database table.

If the attribute is an indicator it must end in IND.

Example: Attribute Names

```
Entity:      FOREST CLIENT

Attributes:  CLIENT NUMBER

             LEGAL FIRST NAME
             LAST NAME

             ADDRESS

             PHONE NUMBER
```

Note that the PHONE NUMBER attribute is different from the NUMBER attribute; NUMBER is a class name, and PHONE is a modifier.

3.8 Relationship Naming

- Relationship names must be meaningful.
- Both sides of a relationship must be named.
- "catchall" phrases (EG 'has' 'is' 'related to', 'associated with') are avoided in favour of more descriptive relationship names.
- Relationship names do not contain '_' (underscores).
- Relationships are all lower case.

Note: *It is often helpful to consider the relationship name in the context of a sentence as follows: EACH entity1 MUST BE/MAY BE relationship ONE AND ONLYONE/ONE OR MORE entity2*

Example: EACH student MUST BE enrolled in ONE OR MORE classes, or EACH class MAY BE comprised of ONE OR MORE students.

3.9 Physical Naming Standards and Structure

Physical model names have different length restrictions in different types of databases so some abbreviation and ingenuity may be required when translating logical model names to physical model names. However, since Oracle allows for 30 character names at the physical level, this has removed the need for abbreviating in many cases. Some restrictions from the logical model rules still apply, e.g. table names, like class/entity names, must be unique throughout the ministry.

Example: Differences between Logical and Physical model names

Logical Model Name	**Physical Model Name**
FOREST CLIENT	FOREST_CLIENT
SEED GERMINATION MEASUREMENT	SEED_GERMINATION_MEASUREMENT

In the physical model, table and/or column names should be spelled out as far as possible up to the current physical length limit for the platform being developed on, and abbreviations should only be used where necessary due to length limits. The general structure for data naming outlined previously must still be followed to produce a table or column name.

READ THIS - AT THIS POINT THERE ARE TWO DIFFERENT SETS OF STANDARDS, MAINLY PHYSICALLY DRIVEN – A WHITEBOARD SESSION AND A PROJECT ASSESSMENT WILL DETERMINE WHICH STANDARDS TO FOLLOW – PLEASE CONTACT YOUR IMB BUSINESS PORTFOLIO MANAGER FOR ASSISTANCE. Chapter 4 is Ministry of Forests Standards and Chapters 5, 6, 7, 8 are from Environment/Agriculture/ILMB Standards. Standards to follow for MAAR and MEM will also be determined at the whiteboard session.

4 Forests Physical Standards

4.1 Using This Standard

This standard is used to name and describe data components. A data component is any object used to capture or store information about the data and systems in a Ministry. Typically these include Classes/Entities and Relationships, Tables and Columns, Views/Materialized views, Application Acronyms, Designer Folders and more.

The developer should be familiar with the General Naming Standards later in this section. When creating and naming a data component during development, the general rules should be applied to give a short and meaningful name within the constraints of the software that will hold the component. If the component is of one of the types named in the specific standards, those standards are to be applied to the name. Once the component is named, there may be additional properties to specify.

4.1.1 Physical Data Model Names

In the physical model, table and/or column names should be spelled out as far as possible up to the current physical length limit for the platform being developed on, and abbreviations only used where necessary due to length limits. The general structure for data naming outlined previously must still be followed to produce a table or column name.

Appendix A contains abbreviations for common words or phrases the ministry has standardized; where such an abbreviation exists, use it instead of the following rules. It should be pointed out that a considerable amount of creativity is often required to come up with an acceptable abbreviation from a corporate database perspective.

General table/column naming standards (see section 4.2 Conventions for suffix and other conventions):

- where reasonable, foreign key column names must be the same as the original table's primary key names; and
- where possible, all column names that point to a code lookup (see NRS Code Management Standards) should have the same name as the code name pointed to - in making this a standard, we judged that the benefit of using the same name (i.e. knowing they are the same just by looking at the column/code names) was of more value than allowing the flexibility of calling the column name something else.
- physical names are also singular. (Oracle Designer defaults for plurals **and must be modified to the singular form** to avoid generation of the table name in plural form.)
- an underscore "_" is used to separate words or components of the name.

To abbreviate a logical model's entity/attribute name so that it will fit into a table/column name:

- if the table or attribute name will fit as is within the character length limit, do not abbreviate, spell out the entire name(s);
- if the word (modifier or class name) appears in Section 4.3, use the abbreviation corresponding to it;
- remove one of all double consonants;
- remove all non-leading vowels; remove 'y' only when it is a vowel and is not the first or last character of the name (Eg. leave the 'y' in for 'MINISTRY' ==> 'MNSTRY'); and
- leave trailing 'E's where they provide a 'soft c' or 'soft g' sound (Eg. leave in the trailing 'e' for 'PIECE' ==> 'PCE'; remove the trailing 'e' for 'MIRACLE' ==> 'MRCL').

Physical Data Model Naming Example

Table	FOREST_CLIENT
Column	CLIENT_NUMBER
Column	CLIENT_NAME
Column	LEGAL_FIRST_NAME
Column	BC_DRIVERS_LICENSE_NO

4.1.2 Oracle Restrictions

Because of the large character length allowances in Oracle, it will be quite common that table and column names for an application's data do not need abbreviating at all. However, if some abbreviation is required, follow the guidelines listed above. The ministry's character length limitations are:

Oracle Entity Name 40 (but 30 is recommended)

Oracle Attribute Name 40 (but 30 is recommended)

Oracle Table Name:	30 characters
Oracle Column Name:	30 characters

Note that since the physical names are generated from the logical names, it is recommended that the logical names also be 30 characters maximum.

Also, for generation, turn off the "set column prefix" option, which would otherwise automatically prefix each attribute name with the entity's acronym.

Note that all table and column names must be approved by the Data Architecture/Database Administration groups prior to any tables being built.

4.2 Conventions

4.2.1 Suffix Conventions

Entity/Table or Attribute	Mandatory Abbr
ORG_UNIT	_DIST or _REG
Application Codes (attributes marked as codes)	_CODE
Y/N Fields	_IND
Audit Tables	_AUD
Link Tables	_LINK
Transaction Tables	_TXN
Validation Tables	_TBL
Cross Reference Tables	_XREF
Physical DB Implementation Names	
Sequences	_SEQ
Indexes	_I
Primary Keys Constraint Names (not on data columns)	_PK
Foreign Key Constraint Names (not on data columns)	_FK
Views	_VW

4.2.2 Foreign Key Conventions

All foreign keys (columns that exist in one table to provide a primary key link back to another table) must use the same name as the original primary key in the original table. Some exceptions are allowed, e.g. prefixing the name with a modifier when two or more relationships exist between two entities. For example, when there are two relationships to Org Unit, one for Region and one for District, a prefix is required resulting in Region_Org_Unit_No or District_Org_Unit_No. Another example is when more information is needed to describe the attribute, e.g. Age_class_code could be prefixed to result in Prev_age_class_code.

4.2.3 Date Conventions

Dates are commonly treated as structures within programs. Note that CYMD is the international standard. Although the abbreviations 'yy-mm-dd' and 'hh-mm-ss' are acceptable everywhere on forms, screens, etc, they should not be used to define the name of a variable (Eg. column name). Instead, the full descriptive phrase of what the date refers to should be used. E.g. APPROVAL_DATE

_DATE is used (instead of DTTM) as a suffix, even though DATE includes the date and time.

_TIMESTAMP is used in naming an attribute which records the specific date and time when an event occurred, e.g. the record was created or updated. However the timestamp column is formatted as DATE.

As a general rule, EFFECTIVE_DATE and EXPIRY_DATE are commonly used terms for describing when data values are effective or expired.

4.2.4 Other Naming Conventions

Some column names within tables mean the same thing although the actual columns themselves are storing information relating to different entities. To help with understanding, the following column names have been defined as standard and should be used in any table where required.

ENTRY_TIMESTAMP

ENTRY_USERID

UPDATE_TIMESTAMP

UPDATE_USERID

4.2.5 Status Codes

All status codes are defined to a particular standard, displayed in the Integrated Data Dictionary (IDD) under the code name STATUS_CODE. Any systems requiring use of status codes must access them through their own defined names, implemented in an application code table. This is to ensure that across systems, all screen entries for status codes remain consistent, so that users are not continually having to change the codes they use for the same status result. The standard status codes are three-character mnemonic (eg. ACT = Active; COM = Complete; EXP = Expired; etc.).

4.2.6 Multiple View Use

If multiple views are required for a table, for example where some columns are restricted from public access (see NRS Data Modelling Standards, section 4.2.4.7 Managing Views - Data Access in an Integrated Environment), then the following naming standards apply:

restricted views (i.e. views that display fewer than the full set of columns contained in the original view) - suffixed with "_VW" (E.g. FOREST_CLIENT_VW); and

join views (i.e. views based on a join or union of two or more tables) - suffixed with "_VJ" (E.g. FOREST_CLIENT_VJ).

if the view is application specific, the view should be prefixed with the application acronym. (E.g. "ECAS_FOREST_CLIENT_VW"

Note that views are always to be created on tables, not on other views.

4.2.7 Sequences/Indexes

Beyond using the appropriate suffix, sequences and indexes should be spelled out in full, and only abbreviated as necessary. E.g. use FOREST_COVER_SPECIES_SEQ, instead of FCS_SEQ which might cause collisions with other sequence names.

4.2.8 4.3 Standard Approved Abbreviations

See Appendix A for a list of standard abbreviations.

5 Agriculture/Environment/ILMB Physical Standards

5.1 Using This Standard

This standard is used to name and describe data components. A data component is any object used to capture or store information about the data and systems in a Ministry. Typically these include Classes/Entities and Relationships, Tables and Columns, Views/Materialized views, Application Acronyms, Designer Folders and more.

The developer should be familiar with the General Naming Standards later in this section. When creating and naming a data component during development, the general rules should be applied to give a short and meaningful name within the constraints of the software that will hold the component. If the component is of one of the types named in the specific standards, those standards are to be applied to the name. Once the component is named, there may be additional properties to specify.

5.1.1 Document Arrangement

The abbreviation section provides a general method for creating an abbreviation for all or part of a data component to be named. General Naming and General Descriptions provide standards and methods that apply globally and are independent of software environments.

Entities/Classes and Attributes are included because they may be used within one or many software tools or may be used outside the tools.

5.1.2 Policy

The Naming and Describing Standards will be used for naming all data components during development of all systems, data models and databases.

5.1.3 Responsibility

Data Architecture in the Architecture section is responsible for maintaining these standards. Periodic updates will be made. Comments will be compiled for the update process. Systems developers and project managers are responsible for ensuring the conventions are followed, and that improvements are suggested to Data Architecture.

5.1.4 Variations from Naming Standards

Where specific conditions complicate the use of the naming standards, consult the Data Architecture section for approval prior to deviation from the standards.

5.2 General Naming

This section presents the general principles used for naming any of the data components. Subsequent sections show where these principles are augmented and constrained to meet the more particular naming requirements of the software or management of the components.

General Naming provides the rules to be followed for the creation of names of a broad range of data components. It includes standard abbreviations and the process for creating abbreviations that are not in the standard list. It shows the process for creating acronyms, and an approved list.

For UML modelling all logical naming is done using UpperCamelCase for classes (e.g. ProtectedLandAreaSP) which map to relational entities in all upper case (e.g. PROTECTED LAND AREA SP). Attributes in UML are lowerCamelCase (e.g. protectedLandAreaID) which map to relational attributes in all upper case (e.g. PROTECTED LAND AREA ID)

It also includes the process to be used to create an application acronym, which is first identified in the Whiteboard sessions.

5.2.1 Abbreviations

This section provides the methods to determine abbreviations for use in naming data components. The approved abbreviation list should be used before creating a new abbreviation.

5.2.1.1 Approved Abbreviations List

See Appendix B for a list of standard abbreviations.

5.2.2 Standards for Creation of Abbreviations

No abbreviations may duplicate those appearing on the approved list. An abbreviation of 4 characters cannot be a word. Abbreviations are determined for words/terms that do not appear on the approved list as follows:

1. Determine length of abbreviation.
2. Put first and last letter of word in first and last position of abbreviation.
3. Remove vowels.
4. Remove one of any double consonants.
5. Fill remaining spaces of abbreviation with consonants of the word in the order in which they appear until the required number of letters is obtained.
   EXAMPLE: To abbreviate 'ABBREVIATION' to 5 characters.

   - Put `A' in location 1 and `N' in location 5.
   - Remove 'E', 'I', 'A', 'I', and 'O'.
   - Remove one 'B'.
   - This leaves 'BRVT' for 3 spaces. Select appropriate characters and fill spaces 2 to 4. Result is ABRVN

5.3 Domains and Suffixing

5.3.1 Domains

A Domain is a collection of descriptors for an entity, usually with a standard representation (data-type). Examples are NAME, DATE, AMOUNT, KEY etc. Some domains are directly supported by a datatype in programming languages; others must be specified for a particular environment. A list of the current standard classes is included in this document.

Some data elements are common to many applications. These are enumerated in the Standard Corporate Data Elements section. The standard elements should be checked and used when developing elements.

Domains are standard representations for collections of objects. Often data elements for many different applications will be built from the same classes. For example, BIRTH_DATE, HIRE_DATE, and CONTRACT_DATE all have the same domain, but may apply to many different entities. One or more standard attributes will be developed for the following classes (some are apparent from the Standard Corporate Data Elements). The standard class name or its abbreviation should be used in the development of element names.

Standard Domain Name	**Abbreviation**
ADDRESS	ADRS
ADDRESSLINE	ADRSLN
AMOUNT (A numeric value of money)	A,AMT,AMNT
CODE	CD
COUNT	CNT
CROSS REFERNECE	XREF
DATE	DT
DAY	DY
YEAR	YR
MONTH	MNTH
DESCRIPTION	DSCRPTN
FLAG	FLG
IDENTIFIER	ID
INDICATOR	IND
KEY	KY
LOCATION	LCN
NAME	NM
NUMBER	NMBR
SERVICE	SRVC
SEX	SX
STATUS	STS
TEXT	TXT

Standard Domain Name	Abbreviation
UNIT	UNT

5.3.2 Suffixing

There are specific suffixing name requirements for physical database objects. These naming requirements are further detailed in section ##. Because physical objects are derived from logical objects the same suffixing applies at the logical level too. The following is a summary:

1. Primary key names are suffixed with _PK (no under bar in logical model)
2. Foreign key names are suffixed with _FK (no under bar in logical model)
3. Unique (primary) key names are suffixed with _UK (no under bar in logical model)
4. Check key names are suffixed with _CHK (no under bar in logical model)
5. Sequences keys are suffixed with _SEQ
6. Indexes are suffixed with _I
7. Standard views are suffixed with _VW
8. Spatial views are suffixed with _SVW
9. Materialized views are suffixed with _MVW and spatial materialized views are suffixed with _SMVW.
10. Triggers are suffixed with _TRG
11. Functions are suffixed with _F, procedures are suffixed with _P, cursors are suffixed with _CSR and packages are suffixed with _PKG
12. Oracle Advanced Queue tables are suffixed with _QTB and the corresponding queue is suffixed with _QUE.
13. Transitional views are suffixed with _TVW.

5.4 Application Acronym Derivation

The dataset or application acronym is used to locate information about the data set, and to name data components within the corporate jurisdiction. It is first identified and confirmed in the Whiteboard sessions.

The title of the dataset or application, and the acronym derived from it, are the first pieces of information derived from the initial whiteboard sessions and must be passed before any data design occurs.

The acronym is 3 to 5 characters in length and should be a representation of the title given to the data set or application within the metadata collection and the data model repository. It is used to identify the dataset or application in the Meta Repository, the container name in the data model repository, all accounts on delivery servers, database schemas, and the prefix for all physical objects created on any of the data bases. It is mandatory that the dataset acronym be cleared of any collision possibility by DBA, before it is accepted.

An acronym can be associated with either an application or a dataset or used for both. It is up to the business area whether they want separate acronyms for a dataset or application. But in most cases the acronym is the same for both.

The acronym is usually derived from the title of the dataset or application and the acronym is built as the first letter of each word in the title of the application or dataset.

A good example of application acronym naming is FPW which maps to the title "Flood Protection Works". The proposed acronym must pass by a DBA search of current acronyms in use for both operational and warehouse use.

An Acronym:

- Mirrors the acronym in metadata requirements collection
- Acts as a prefix for physical objects on implementation.
- Is used in application delivery for script naming.
- Is used for application documentation naming and directory storage.
- A general means of communication through emails, etc.
- Should be one and only one in the corporate jurisdiction.

The Application Acronym will be automatically prefixed to **all** 'physical' database objects such as tables, views, packages, sequences, primary key names, foreign key names and roles. Functions and procedures that are not encapsulated in packages must also be prefixed with this acronym.

The intent of prefixing the Application Name on all objects is to reduce the possibility of namespace collisions. For example, the Data Registry application uses DR as its short name. Therefore, the DATA UNIT entity/class becomes the DR_DATA_UNITS table. Approval to use a new or change and existing application acronym must be obtained from the Database Administrator section to ensure that there are no duplicate names.

5.5 Use of Business Acronyms in Descriptions

When an acronym is used in a textual description, they must be shown in full and qualified. The only time an acronym doesn't need to be qualified is if it is universally understood like the acronym SIN for Social Insurance Number. An acronym is qualified by spelling the name in full followed by the name in brackets.

EXAMPLE: Global Positioning System (GPS).

A hierarchal acronym is defined as drill down in nature. That is, if we use a qualified acronym at the dataset description level in the data model, it can be used by itself without qualification throughout the data model in any object.

If an acronym is first introduced at a (class/entity) object level in the description, it must be qualified, but afterwards in can be used without qualifications in any of the attributes associated with that (class/entity) object.

If an acronym is used and is qualified in one (class/entity) object description and then the same acronym is used in another's (class/entity) object description it must be qualified again. If an acronym is used in an attribute's description it must be fully qualified, unless the acronym occurs in the attribute's (class/entity) object level.

Within Oracle Designer acronyms qualified at the (class/entity) object level do not need to be qualified at the attributes level. The qualification is considered hierarchical. However they must be qualified when used in another (class/entity) object.

5.6 Deriving Names

Logical model names are not normally concerned with physical storage restriction, so theoretically can be any length with no need for abbreviation. However, since the logical name is used to generate the physical table name and column names, a limit of 24 to 26 characters is recommended for (class/entity) object names and data attribute names including spaces. (class/entity) object names must allow for prefixing with the application acronym.

5.7 Logical Naming Standards and Structure

In the logical model, all words ((class/entity) object and attributes) are completely spelled out; no abbreviations unless absolutely necessary for emphasis or clarity. Abbreviations are only used that are universally accepted. Example: SIN (Social Insurance Number)

5.8 Class/Entity and Attribute Names

The general form of the Class/Entity Name is:
MODIFIERS : ENTITY NAME : MODIFIERS : DOMAIN NAME
Spaces delimit the words. The full name is in upper case.

For example, "Casual Employee Hire Date" the class/entity name would be CASUAL EMPLOYEE HIRE DATE.

Some general comments about the structure of (class/entity) object and attribute names:

Names must be:

- Meaningful (use of one to four real words. When the class/entity name is only one word, it must not be an Oracle reserved word or a word that does not stand by itself, meaning, fully explanatory on its own. This meaning is at the discretion of the Data Architect.) For example, the word ACTION does not stand by itself, as there can be many kinds of ACTION. The use of one-word names should therefore be limited. Another example, in the corporate context there is likely to be more than one type of RANGE or PERMIT, so it must be qualified.

- Self-documenting (by looking at the name, the reader should have a good idea what the name means without having to read the description). Use names that have meaning to the users. If you find yourself having to explain what your name means, then it is a good indication that the name is not meaningful.

- Derived from the business use or purpose but understandable to a non business user (or by different people from different areas of the public, reading the name at different times). These users should have the same understanding of what the name means as the business area of the application.

- Abbreviations or acronyms should be used only where no combination of names would produce an acceptable result. They are to be avoided unless universally understood.

- (Class/Entity) Object and attribute names do not contain '_' (underscores).

- Class names are modelled in UpperCamelCase, so for example when using Unified Modeling Language (UML) the class would be named InventoryForestProject, compared to the Entity Relationship (ER) structure the name would be INVENTORY FOREST PROJECT.

- Attribute names in UML are lowerCamelCase where the first word is all lower case by convention. So an attribute called FOREST INVENTORY COLLECTION PARAMS in the ER structure, would be named forestInventoryCollectionParams.

- If the (class/entity) object represents a code table it has the suffix CODE or CD.

- If the (class/entity) object represents a cross reference table it has the suffix XREF.

5.9 Physical Naming Standards and Structure

In the physical model, table and/or column names should be spelled out as far as possible up to the current physical limit for the platform being developed on, **and abbreviations used only where no combination of names would produce an acceptable result.** The general structure outlined previously must still be followed to produce a table or column name. See Abbreviations section5.2.1 for common abbreviations and method of determination.

Once the object is named at the logical level and approved, the abbreviation process should be used to develop shorter names where necessary for physical implementation. If using the Oracle Designer tool, this may be done in a logical review when prefixing in the plural property of the entity.

Physical names are restricted to physical storage limits so some abbreviation and ingenuity may be required when translating logical model names to physical model names. However, since Oracle allows for 30 character names at the physical level, this has removed the need for abbreviating in many cases. Some restrictions from the logical model rules still apply, e.g. table names, like class/entity object names, must be unique throughout the ministry. Because of the requirement for the application prefix or acronym to be assigned to the object, the limitation is 24 to 26 characters. The acronym can be up to 5 characters, generally not greater than 4 characters and usually 3 characters in length.

5.10 Domains

The Domain Name (within Oracle Designer) should not be duplicated if previously defined as corporate. Instead it should be copied from corporate domains and re-used.

6 Agriculture/Environment/ILMB General Describing Guidelines

This chapter is from the Ministries of Agriculture, Environment and ILMB Standards

(Class/entity) object type and attribute definitions should clearly describe what business information they record, using:

- precision - they resolve ambiguities and qualify imprecise terms;
- completeness - all terms are defined;
- clarity - plain English sentences;
- brevity - brief and to the point;
- compatibility - with other definitions;
- understandable - to a non-business point of view.

Descriptions for all (class/entity) objects and attributes are used to populate table and column comments when building the physical database objects in Oracle Designer. The comments may be a summary of or the full description, it is acceptable to be the first sentence of the description.

Note: Descriptions shall be full sentences or a combination of sentences, as a general standard the object name is the subject of the first sentence.

The entity or attribute name is mostly used as the subject of the sentence in both Entity Relationship (ER) modelling and Object Oriented Modelling (OOM) modelling. The (class/entity) object or attribute name is capitalized and no underbars are included in the name.

Example: for an attribute EMPLOYEE ID the description would be: "An EMPLOYEE ID is a unique identifier for each employee, alive or dead, who works or has worked, for the company."

If any other attribute or (class/entity) object is referenced in a description it is also capitalized in this manner with no underbars but spaces in the name. UML describing follows ER standards, for object names imbedded in descriptions.

6.1 Describing Standards and Structure

The purpose of describing in a data model is to show (class/entity) object and attribute definitions that an application must store in order to satisfy business requirements and provide an explanation of how names are used, in a manner that a layperson can understand. It may also show how the data is related.

- Logical descriptions are used to enhance the understanding of (class/entity) object and attribute names.

- Descriptions for all (class/entity) objects and attributes must be used to populate table and column comments when building the physical database objects. The comments may be a summary of or the full description. It is acceptable to be the first sentence of the description, which is intended to be the summary.
- Generally the description must explain what the (class/entity) object is to non-application personnel, or non business users.
- Descriptions for abstract (class/entity) objects should contain concrete examples, but code attributes must have examples.

Structured Descriptions are categorized into three categories, Definition, Example and Miscellaneous, (these category titles are not included in the description, they show the paragraph order): for example:
DEFINITION
A SURVEYOR ROLE TYPE is a type of role that a party may play in the context of a permit, PMP, licence or certificate.
EXAMPLE
Examples of SURVEYOR ROLE TYPE are 'Located Within' (i.e. Region or regions that the PMA, Licence or Certificate are physically located or affect in the case of items that cross regional boundaries.),'Administering Region' (i.e. the region actually responsible for a permit. PMP, Licence or certificate), 'Consultative' and 'Contact'.
MISCELLANEOUS
Note that Administering Region may include HQ, as HQ staff may perform the hands-on administration of a permit, PMP, licence or certificate.

This entity is used to hold the valid list of Approval Roles that are available for use in CRISP. This list is only to be modified by the Application Manager, and then, only when adding new functionality to the system.

Recently, we've added 'Approved Training Agency', which will now share the duties of issuing Certificates with 'Administering Region'.

6.2 Descriptions for (Class/Entities) Objects

- This section guides the production of clear, concise and understandable descriptions. See the Data Development Guide for standards related to the meaning of the entities.
- *First sentence summarizes class or entity.* The first sentence of the definition gives a noun, or group of nouns, with modifying adjectives or phrases which summarize the meaning of the (class/entity) object, a verb and predicate. It must explain what the (class/entity) object is to non-application personnel, or non business users. Descriptions for abstract (class/entity) objects contain concrete examples. It can be categorized into Definition, Example and Miscellaneous.
- *Example must be included.* The first example should be one which is firmly in the collection, which will solidly reinforce the readers understanding. Other examples should reveal the bounds

of the (class/entity) object. An example which the reader might not consider as an instance will help to establish the boundary. This requires anticipating instances where there may be confusion for the reader. Also of value is an example of an instance which is not a valid member of the group. This shows the boundary from the other side. Often wording which explains why examples are, or are not, included is necessary. The definition should include clear explanation of why these examples are, or are not, entities.

E.G. 1: "As soon as a person becomes a candidate for a position in the company they become an employee whose status is unhired. This allows personal information to be entered only once. Applicants who are not considered for a position are not employees. "
E.G. 2: "A COMPANY EMPLOYEE is a person, alive or dead, who works or has worked, for the company."

- *Business intent of (class/entity) object included.* Intent: Where possible, the intent of the (class/entity) object should be shown. This allows the reader to understand the rationale for the (class/entity) object and fit their own understanding of the business into the one represented by this (class/entity) object.

  E.G.: The intent of the COMPANY EMPLOYEE is to capture any information about a person the company deals with in an employee employer relationship. This includes personal information, tax and statute information and skills and capabilities.
- *Verbiage in (Class/Entity) object Descriptions*: The description is worded without reference to data modeling terminology. That is refraining from using the words column, attribute, table, entity and class in the description. The words Class, Entity and Table are replaced with object. The descriptions for (class/entity) objects are not to be worded to contain verbs that reference physical objects or mention of the physical database object. That is (class/entity) objects do not "contain" or "hold" information, tables do that.

- *Acronym Qualification in description (see section 5.5)*
  For each (class/entity) object, if it has an acronym in its definition, the acronym must be fully qualified, in the suggested format: Universal Transverse Mercator (UTM). If the fully qualified acronym exists in the application system documentation group it can be freely used throughout without qualification.

- *Differentiated from similar (class/entity) objects*. Differentiation: Where there may be confusion between two similar (class/entity) objects, either because their names are similar or because their descriptions or relationships are similar, there should be explicit text which explains the difference.

  E.G.: The COMPANY VOLUNTEER object may be confused with the COMPANY EMPLOYEE object. The WORKER object is used to collect information about clients who are supported in the work they do.

6.3 Descriptions for Attributes

- *First sentence summarizes attribute description.* Nouns: The first sentence of the definition gives a noun, or group of nouns, with modifying adjectives or phrases which summarize the meaning of the attribute, a verb and predicate.

  E.G.: An EMPLOYEE STATUS is the status of a person, who is alive or dead, who works or has worked, for the company. EMPLOYEE STATUS captures field values such as sick, on maternity leave, on vacation.

- *Example should be included for clarity, but must be included for code attribute descriptions* The first example should be one which is firmly in the collection, which will solidly reinforce the readers understanding. Other examples should reveal the bounds of the attribute. An example which the reader might not consider as an instance will help to establish the boundary. This requires anticipating instances where there may be confusion for the reader. Also of value is an example of an instance which is not a valid member of the group. This shows the boundary from the other side. Often wording which explains why examples are, or are not, included is necessary.

- *Differentiated from similar attributes.* Differentiation: Where there may be confusion between two similar attributes in an (class/entity) object, either because their names are similar or because their descriptions are similar, there should be explicit text which explains the difference between the attributes.

- *Verbiage in Attribute Descriptions*: Attribute descriptions: The description should be worded without reference to data modeling terminology. Refrain from using the words column, attribute, table, and entity in the description. It is the usual practise to use descriptions for table comments and that being the case it is better to refer to (class/entity) objects and tables as objects, if necessary. Columns and attributes should be described as characteristics of the object.

- *Acronym Qualification in description (see section 5.5)*: For each attribute, if it has an acronym in its definition, the acronym must be fully qualified, in the suggested format: Universal Transverse Mercator (UTM). If the fully qualified acronym exists in the application system documentation group it can be freely used throughout without qualification.

6.4 Descriptions for Unique Identifiers

A unique surrogate identifier that is sequenced, and becomes the primary key for the table with the suffix _ID will have a generic description, as a standard, for all entities and tables. The description/comment for a unique surrogate identifier will be in the following form:

“The <Unique-attribute-name> is a unique surrogate identifier for the object <class/entity-name>.”

This was determined to be a standard:

- So that foreign keys would have a description that points to the correct object.
- So that (class/entity) object and attribute names were used in the description, instead of that for tables and columns with the appropriate wording.

E.G.: The EMPLOYEE ID is a unique surrogate identifier for the object COMPANY EMPLOYEE.

If the unique identifier has business context and is not a surrogate, the description should follow the standard description for attributes as stated above. NOTE: It is encouraged that the developer uses a surrogate unique identifier. Rather than having a business key as the unique identifier for the (class/entity) object, model the business key as a unique key and have a surrogate unique identifier for the main key for the (class/entity) object.

6.5 Descriptions for All Views

- All views (Materialized and Spatial included) must follow the conventions for describing entities and attributes 5.1, 5.2 and 5.3 above. The exception is the transitional view, which needs no descriptions on the attributes.
- The views need to be described and commented just like entities and their attributes.
- If the view in the current data model sources an object from another data model, the sourced table and column must indicate it in the column description.

E.G.: This is the FEATURE NAME for the English name of the feature appended with the FEATURE ATTRIBUTE, being a characteristic or subtype of the feature that serves to differentiate it from features of other subtypes. This column is derived from the table CORP FEATURES by appending three columns. The PLSQL Syntax is: rtrim(b.feature_name)||decode(b.feature_attribute,NULL,NULL,' - '||rtrim(b.feature_attribute))"

6.6 Structured Notes

Issues, decisions and notes should be recorded in the Notes property of the relevant object (e.g. entity, attribute, function, table, column, and module). The suggested format is to prefix the text with indicators of

1. What phase of development (Analysis, Design, or Build) (A, D, B)

2. What type of note (Question, Point, Answer, or Decision) (?, !, A, D)

3. Date that the issue was raised, or resolved (YYYY-MM-DD)

4. Initials of the analyst who raised this issue

An example is:

A? 1998-01-18 GW There may be an opportunity to share this functionality with Record contact information about a new permittee/PMP'er/Certificate holder.

The notation is:
<Phase><Note Type> <Date> <Initials> <Note Text>
<Phase> is one of A, D, B
<Note Type> is one of ?, !, A, D
<Date> is in the format YYYY-MM-DD

Structured analysis or design comments should be placed here, for example:

OUTSTANDING ISSUES
===================
A! 1998-01-21 GW GENERAL LOCATION, and all information specifically

related to time/place details have been deleted (R.

Adams, 1998-01-19) from EXAMINATION entity and

the entire entity has been deleted.

IMPLEMENTATION NOTES
=====================

MISCELLANEOUS
==============

CHANGE HISTORY
===============

7 Agriculture/Environment/ILMB Oracle Naming

This chapter is from the Ministries of Agriculture, Environment and ILMB

This section presents the standards for providing names and text (definitions, description, or other) in the production of documentation for data components.

Data components held and managed in the Oracle Database are called Oracle Objects. These include Tables, Synonyms, Tablespaces, etc. Naming standards for these are driven by the constraints of the Oracle database and the need to recognize and manage these components.

Note: The Database Design Transformer copies this information from the entity or attribute to the corresponding physical objects in Oracle Designer. This section covers most of what is not done logically.

7.1.1 Object Naming Conventions

The conventions for naming database objects (including tables, views, columns, indexes, sequences, roles, packages and functions, etc.) follow those basic naming conventions imposed by Oracle:

- object names should be maximum of thirty (30) characters long with these exceptions: - names of databases are limited to 8 characters - names of database links can be as long as 128 characters
- should not contain quotation marks
- are not case-sensitive
- can only contain alphanumeric characters from the database character set and the characters _, $, and #. The use of $ and # is strongly discouraged. Names of database links can also contain periods (.) and at-signs (@)
- should contain underscores (_) for visual clarity
- must begin with a letter
- must not duplicate an ORACLE reserved word
- should not contain the word DUAL (e.g. DUAL is the name of a dummy table frequently accessed by Oracle7 tools such as SQL*Plus and Forms)
- must not duplicate the name of another database object
- should use nouns, rather than verbs
- should be as descriptive and as short as possible
- should use standard abbreviations when required
- should not be ambiguous

In addition, it is a standard that the Application Name (acronym such as CRS) be prefixed to all 'physical' database objects such as tables, views, packages, sequences and roles, etc.

7.2 Oracle Object Naming Conventions

These objects are first named within Oracle Designer, usually at the logical level with generation to the physical level.

NOTE:
The physical alias property of the table (derived from short name property of the entity) is highly recommended to contain the application acronym to easily generate prefixes when using Designer. This is especially useful because it becomes the default name used in creating other objects in Oracle designer. Since all physical objects are prefixed this is a time saver.

7.2.1 Table Names

- The table name must be prefixed with the Application Acronym and an underscore, e.g.: GOAT_THEMES
- Cross-reference table names should be suffixed with _XREF
- Code tables must be suffixed with CODES, TYPE_CODES or _TYPE_CDS, _CDS
- Spatial tables must be suffixed with _POLY, _POINT, _LINE, or ANNO depending on the type of layer feature or SP or SPG. For spatial layer standards see SDE Layer Standards Section 7.6.
- Table names are highly recommended to be kept as generated from the associated Entity Plural Name in the logical. That is the table name should be passed in the logical review by looking at the plural property of the entity when using Oracle Designer.
  NOTE: mandatory suffixes are not pluralized, except for CD and CODE.
- If a table is defined manually, the table name should be plural.
- An Oracle Advanced Queue table and its associated queue have the same name except for the suffix. Queue tables are suffixed with _QTB and the corresponding queue is suffixed with _QUE.

7.2.1.1 Table Trigger Names
See section 7.2.10 Database Triggers.

7.2.1.2 Column Names
- Column names, including foreign key columns, must NOT be prefixed.
- The default column name generated using the Database Design Transformer is the name of the corresponding attribute.
- If the primary key column is not a unique business identifier but a system generated (surrogate) primary key column, the column name must be suffixed by _ID and be the same name as the **"table"** without the application prefix with no plurality.
- That is to reiterate, the CLASS (suffix) for surrogate key column is _ID. The surrogate key usually is a unique sequence in this instance and the primary key of the table or unique primary key at the logical level. If the primary key is not a unique business identifier but a system generated surrogate primary key column, the column name must be suffixed with _ID and be the same name as the **"table"** without the application prefix and no plurality.

- If a Super-type (Single Table) implementation of Sub-types is chosen, specify the name of the discriminator column as: _TYPE;
- Where reasonable, foreign key column names must be the same as the original table's primary key names.
- The primary key column for code tables has the same name as the database table without the application prefix. It is not pluralized like the table name. That is where possible, all column names that point to a code lookup must have the same name as the code name pointed to - in making this a standard, we judged that the benefit of using the same name (i.e. knowing they are the same just by looking at the column/code names) was of more value than allowing the flexibility of calling the column name something else.
- The CLASS (suffix) for indicators or flag columns is _IND. The actual indicators must be fully described in the description, if not in Allowable defined values field or in a domain.

7.2.2 Primary Key Name

The default name, as generated by the Data Design Transformer, should be kept as-is. This ensures that:

- It contains the table alias, suffixed with _PK. The table alias should contain the acronym when using Oracle Designer, it came from the short name property of the entity.
- It is prefixed with the Application Acronym and an underscore, e.g.: FNI_TREATIES_PK

7.2.3 Foreign Key Name

- It contains the table alias, suffixed with _FK. The table alias should contain the acronym when using designer, it came from the short name property of the entity.
- It is prefixed with the Application Acronym and an underscore,
  E.G.: FNI_TREATIES_FNI_TREAT_RES_FK

7.2.4 Check Constraint Name

- It must contain the table alias suffixed with _CHK. The table alias should contain the acronym when using Oracle Designer, it came from the short name property of the entity.
- If multiple check constraints are defined for a table, then they should be suffixed with _CHK
- Must be prefixed with the Application Short Name and an underscore,
  E.G.: FNI_REGIONS_CHK

7.2.5 Index Names

- Must be prefixed with the application acronym and underscore and suffixed with _I.
- Table aliases should be used (e.g. emp_dept_fk_i is a foreign key index from the employees table to the departments table)
- Indexes supporting foreign key constraints should have the same prefix (e.g. index t_mbr_c_thm_fk_i supports the foreign key constraint t_mbr_c_thm_fk)

7.2.6 View Names

View names must be prefixed with the Application Acronym and an underscore (e.g. GOAT_ GOAT_REGIONAL_THEMES_1998 where GOAT is the application acronym). The view name should be descriptive of the data that the view is presenting.

The following suffixes are used to classify types of views:

- _VW for a standard view;
- _SVW for a spatial view;
- _MVW for a materialized view;
- _SMVW for an Oracle spatial materialized view;
- _TVW for a transitional view

NOTE: *The purpose of a transitional view is to change the form/structure of a dataset from its source to its target.* A transitional view does not conform to the Oracle Designer standard in that the description/comment on the view only specifies its system use. Transitional view columns do not require comments or descriptions. Otherwise all other standards for views are required.

A transitional view is not to be used for queries or have roles applied with grants. Its target object must exist and be fully described.

Transitional views are currently used in FME replication, WUTIL replication and the capture of the current state of ESRI geodatabase multi-versioned objects.)

7.2.7 Sequence Names

- Must be prefixed with the table name (which itself is prefixed with the Application Short Name) and an underscore, e.g.: GOAT_THEMES_SEQ
  (where the table name is THEMES and the ACRONYM is GOAT)

- Sequences are named in the following manner:
  The standard is <table_name including acronym> _SEQ1 --- <table_name including acronym > _SEQ2 if multiple columns in a table. Table name contains the prefix or acronym.

- If the sequence is too long, standard abbreviation techniques apply.

  Note: If the Sequence name is too long the standard abbreviation rule found below in section 10 apply, or the alias of the table is allowed as long as it contains the prefix.

- Must be suffixed with _SEQ.

- If multiple sequences are required for a single table, the sequence name should be suffixed with _SEQ* where the '*' increments sequentially.

7.2.8 Unique Key Constraints

The default name, as generated by the Data Design Transformer, should be kept as-is. This ensures that it:

- It contains the table alias (which must contain the acronym as the prefix)
- It is suffixed with _UK
- It is prefixed with the Application Short Name and an underscore, e.g.: RCM_BAP_UK1
- Additional unique keys are numbered consecutively (UK1, UK2, UK3...)

7.2.9 Database Triggers

- Database Triggers are named in the following manner:

  The standard is <acronym> <table_alias> _<B/A> <R/S> _<I/U/D> _TRG

  Note: *B/A is Before/After, R/S is Row/Statement, IUD is Insert / Update / Delete.*

  For example: > GOAT_THM_ BR_IUD_TRG is a trigger on the GOAT_THEMES table, triggered Before Row upon the operations Insert, Update and Delete.

- Database Triggers for updateable views are named in the following manner:

  The standard is <acronym> <table_alias> _<IO> _<I/U/D> _TRG

  Note: *IO states the view is instantiated , IUD is Insert / Update / Delete.*

  For example: > GOAT_THM_ IO_IUD_TRG is a trigger on the GOAT_THEMES_VW view, triggered upon the operations Insert, Update and Delete.

The name must reflect the trigger properties.

7.2.10 Proxy or Queue Names

- If a queue account is needed for an application the standard naming is QUEUE_<acronym>.
- If a proxy account is needed for an application the standard naming is PROXY_<acronym>

7.2.11 Role Naming Standards

This standard should be consistently applied across platforms (e.g. Unix, Windows, Database), applications (e.g. Oracle forms, Web applications, SQL*Plus), and users (e.g. internal users accessing Ministry applications, general public accessing the master warehouse).

This naming standard is based upon the Role Based Access Control (RBAC) security model and the IDIR naming standard that is used by Shared Services BC (SSBC).

7.2.11.1 Role Naming Standard Description

When naming roles, use the following standards:

- Do not include spaces in the role name.
- Use only the characters A-Z, underscore (_), and 0-9.
- Use only upper case.
- Use a maximum of 30 characters, including underscores.
- Use the following naming convention:
- MMMM_TTTT_NNNNNNNNNNNNNNNNNNNN

MMM	Use a specific 3 character role prefix - SRM (SystemRoleMatrix) for operational and warehouse databases within the CSD infrastructure databases.
TTTT	2 to 4 character role type
NNN...	Up to 24 character descriptive role name so full role name does not exceed 30 characters, including underscores. A location code (or owner) can be placed here as a prefix, if appropriate (see NET role type below). As well, the Generic Descriptive Words described below must be used to the exclusion of other words that describe a similar function.

7.2.11.2 Role Types

The purpose of the role type is to categorize the different roles to make the model easier to administer. When there are many roles it is difficult to determine which role is the most appropriate for a particular purpose. With the role type, the number of roles that need to be investigated (i.e. reading the description and usage policy) is reduced. It will be easy to find a role associated with a particular application, as the role type will be the assigned application acronym for that application. The Role Types are extensible in the sense that if a need arises for a grouping not fulfilled by the types below, you can create a new role type. The role type can be one of the following:

APP	Ministry-assigned application acronym (e.g. CMS, COOR, ERS, FTP) to be used for any access associated with the application (e.g. network, Oracle). Where there is a need for a role that crosses technology (e.g. Oracle, Unix, network) but is not part of a designated application, use APP itself as the designation.
WHSE	Warehouse (both Oracle warehouse and any associated file system data, e.g. spatial data). If the role is to be associated with an application (e.g. GOAT), then the APP role type above should be used instead.
NET	Unix or LAN access for privileges to the network for a particular organizational area (e.g. SRM_NET_EXEC_BASE for the privileges everyone in the executive receives). This role type is intended for access to network resources only. If the access is used for, or tightly associated with, warehousing or an application, the role type for those purposes should be used instead. This can easily be determined if the users who will be a part of the role are the same as those who are part of the application or warehousing role(s). A location code denoting the owner or area responsible for the role should be placed at the beginning of the descriptive name (e.g. WLAP_NET_FSJ_TEMPLATE_SHARE is owned by Fort St. John).
ORA	Oracle only role (e.g. DBA type roles can go here). If the purpose of the role is to provide access for an application or warehouse, use the APP or WHSE role type shown above instead.
SEC	Security administration roles that hold permissions to grant or revoke permissions for a certain area.

Table 1: Role Types

7.2.11.3 Generic Descriptive Words

The descriptive naming portion of the role name (the NNNN portion) is used to describe the purpose of the role. Below is a list of descriptive words that should be used to describe certain aspects of a role.

These words provide for easy administration of roles by consistently using the same words for the same purpose. Once an administrator knows these generic words, they will be easily able to know the function of a role. The two things that are described by these words are the placement of the role in the role network and the target security level of the audience. For example, BASE and ADMIN describe the least and most privileged roles in a network and PUB and MIN describe the security level of the intended audience of the role. Where appropriate, these names can be combined in a role name to better describe the role. For example, the role SRM_WHSE_EMS_BASE_PUB describes the least privileged role for the EMS warehouse. This role provides those privileges that are appropriate for the public.

ADMIN	Highest privileged role in a network for an application or functional area. There should be only one ADMIN role for each application or functional area. If there is a need for two different roles that have similar but different administration needs, the MGR name should be used instead.
MGR	Highest privileged role in a network for a portion of an application or functional area; this name will only be used as a suffix of a description of the portion to be managed (e.g. SRM_COOR_CASE_MGR). MGR roles are junior in privileges to the ADMIN role
USER	Least privileged role in a network needed to use the application or functional area; may require the base role; due to the multiple uses of the term user, adding additional description to the role name is mandatory (e.g. SRM_EMS_EXECUTIVE_USER)
BASE	Least privileged role for an application or functional area; replaces roles such as READ_ONLY and VIEWER. A base role must never have any junior roles. Where appropriate, a suffix of the security level for the target recipient of the role should be included (e.g. SRM_COOR_BASE_MIN).
PUB	A role that contains privileges acceptable for anyone in the public to receive.
GOV	A role that contains privileges acceptable for anyone in the provincial government to receive.
MIN	A role that contains privileges acceptable for anyone in the Ministry identified in the role name to receive.
xxx	Lower levels of security (business area or individual business function) should be described using a descriptive name (e.g. SRM_CRS_ACCOUNTS_CLERK).

Table 2: Generic Descriptive Words

7.2.11.4 *Data Warehouse Roles*

Each warehouse resource data schema must have at least 3 roles and can have up to two more special roles defined.

Mandatory roles (created by the DBA's in conjunction with the schema creation):

PUBLIC	unrestricted access to public
GOVERNMENT	access to internal employees
ADMIN	full control of dataset

Optional roles:

BUSINESS AREA	specific to business area requirements
SPECIAL ACCESS	custom access requirements

Note: roles used specifically for queuing need a "_QUE" suffix.

7.2.11.5 Role Naming Standard Examples

SRM_EMS_BASE_PUB	SRM_EMS_USER
SRM_COOR_CASE_MGR	SRM_COOR_ADMIN
SRM_ORA_VIC_PPR_DBA	SRM_NET_VIC_EXEC_BASE
SRM_NET_VIC_EXEC_ADM	SRM_NET_FIN_BASE
SRM_SEC_ADMIN	SRM_SEC_ADMIN
SRM_APP_LIBC	SRM_NET_KAMLOOPS_ENF_BASE
SRM_WHSE_AIR_BASE_PUB	SRM_WHSE_AIR_USER_GOV
SRM_WHSE_AIR_ ADMIN	SRM_WHSE_CRIMS_READ_SA

7.2.11.6 Profile Naming Standards for WebADE

WebADE profiles names follow the above role naming standards (Sections 6.1.12.1 and 6.1.12.2). The difference is that they do not carry the SRM_ prefix, but the context is the same where a Profile maps to a database role. When a profile does not have a database role but maps to a function or something else, it follows the standard role naming conventions above without the SRM_ prefix. So for example:

Database role	*WebADE Profile*	*WebADE Proxy account*
SRM_EMS_USER	EMS_USER	PROXY_EMS_USER
SRM_ATIS_ASSET_MANAGER	ATIS_ASSET_MANAGER	PROXY_ATIS_ASSET_MANAGER
(none)	EMS_MAPLOCATOR	PROXY_EMS_MAPLOCATOR
(none)	QRGL_GEODE_APPROVER	(none)

7.2.11.7 PLSQL Names

PLSQL packages, procedures, functions and cursors are named in the following manner:

- PLSQL is named according to good naming practices as mentioned in Section 4 "General Naming".
- They are all prefixed with the application acronym
- They are suffixed according to the suffixing rules in Section 4.2.2, point 11.
- If a procedure, a function or a cursor is a part of a package, then suffixing and prefixing does not apply.

7.3 Standard Corporate Data Elements

Standard data element for Auditing

For existing applications:

- WHO CREATED not null VARCHAR2 (30) --- (32 if using GUID)
- WHEN CREATED not null DATE default sysdate
- WHO UPDATED VARCHAR2 (30) --- (32 if using GUID)
- WHEN UPDATED DATE optional default sysdate

For new applications:

- WHEN CREATED not null TIMESTAMP default systimestamp
- WHEN UPDATED not null TIMESTAMP default systimestamp (same as WHEN_CREATED for new record)

The standard data elements for temporality

For existing applications:

- EXPIRY DATE   DATE (optional) default sysdate

For new applications:

- EXPIRY DATE not null TIMESTAMP (make Oracle default artificial high value)
- ESTABLISHED DATE not null TIMESTAMP default systimestamp

If you are not using the attribute WHEN CREATED, ESTABLISHED_DATE is recommended. Temporality and date comparison is discussed in more length in the document Ministry Specific Data Design Paradigms.

The standard data elements for spatial layers are:

- OBJECTID ---- (NUMBER (38) OPTIONAL)
- GEOMETRY ---- (NUMBER (38) OPTIONAL)
- FEATURE CODE ---- (VARCHAR (10) OPTIONAL)
- FEATURE CLASS SKEY ---- (VARCHAR (10) OPTIONAL)
- FEATURE AREA ---- (NUMBER (10) OPTIONAL)
- FEATURE PERIMETER ---- (NUMBER (10) OPTIONAL)

The standard data element for Optimistic Locking is:

- REVISION COUNT ---- (NUMBER (10) not NULL default 0)

Optimistic Locking is discussed in more length in the document Ministry Specific Data Design Paradigms.

7.4 Oracle Restricted Words

7.4.1 Oracle Reserved Words

The following list contains these reserved words. Words followed by an asterisk (*) are also ANSI reserved words. Data Architecture strongly advises that these words not be used in naming business objects.

ACCESS	ADD	ALL	ALTER	AND*
ANY*	AS*	ASC*	AUDIT	BETWEEN*
BY*	CHAR*	CHECK*	CLUSTER	COLUMN
COMMENT	COMPRESS	CONNECT	CREATE*	CURRENT*

DATE	DECIMAL	DEFAULT*	DELETE*	DESC*
DISTINCT*	DROPROW	ELSE	EXCLUSIVE	EXISTS*
FILE	FLOAT*	FOR*	FROM*	GRANT*
GROUP*	HAVING*	IDENTIFIED	IMMEDIATE	IN*
INCREMENT	INDEX	INITIAL	INSERT*	INTEGER*
INTERSECT	INTO*	IS*	LEVEL	LIKE*
LOCK	LONG	MAXEXTENTS	MINUS	MODE
MODIFY	NOAUDIT	NOCOMPRESS	NOT*	NOWAIT
NULL*	NUMBER	OF*	OFFLINE	ON*
ONLINE	OPTION*	OR*	ORDER*	PCTFREE
PRIOR	PRIVILEGES*	PUBLIC*	RAW	RENAME
RESOURCE	REVOKE	ROWID	ROWLABEL	ROWNUM
ROWS	SELECT*	SESSION	SET*	SHARE
SIZE	SMALLINT*	START	SUCCESSFUL	SYNONYM
SYSDATE	TABLE*	THEN	TO*	TRIGGER
UID	UNION*	UNIQUE*	UPDATE*	USER*
VALIDATE	VALUES*	VARCHAR	VARCHAR2	VIEW*
WHENEVER	WHERE*	WITH*		

Depending on the Oracle product you plan to use to access a database object, names might be further restricted by other product-specific reserved words. For a list of a product's reserved words, see the manual for the specific product, such as PL/SQL User's Guide and Reference.

7.4.2 Oracle Keywords

Data Architecture strongly advises that these words not be used in naming business objects.

ADMIN	AFTER	ALLOCATE	ANALYZE	ARCHIVE
ARCHIVELOG	AUTHORIZATION*	AVG*	BACKUP	BECOME
BEFORE	BEGIN*	BLOCK	BODY	CACHE
CANCEL	CASCADE	CHANGE	CHARACTER*	CHECKPOINT

CLOSE*	COBOL*	COMMIT*	COMPILE	CONSTRAINT
CONSTRAINTS	CONTENTS	CONTINUE*	CONTROLFILE	COUNT*
CURSOR*	CYCLE	DATABASE	DATAFILE	DBA
DEC*	DECLARE*	DISABLE	DISMOUNT	DOUBLE*
DUMP	EACH	ENABLE	END*	ESCAPE*
EVENTS	EXCEPT	EXCEPTIONS	EXEC*	EXECUTE
EXPLAIN	EXTENT	EXTERNALLY	FETCH*	FLUSH
FORCE	FOREIGN*	FORTRAN*	FOUND*	FREELIST
FREELISTS	FUNCTION	GO*	GOTO*	GROUPS
INCLUDING	INDICATOR*	INITRANS	INSTANCE	INT*
KEY*	LANGUAGE*	LAYER	LINK	LISTS
LOGFILE	MANAGE	MANUAL	MAX*	MAXDATAFILES
MAXINISTANCES	MAXLOGFILES	MAXTRANS	MAXVALUE	MAXLOGMEMBERS
MIN*	MINEXTENTS	MINVALUE	MODMODULE*	MOUNT
NEW	NEXT	NOCACHE	NOCYCLE	NOARCHIVELOG
NOMAXVALUE	NOMINVALUE	NONE	NOORDER	NORMAL
NORESETLOGS	NOSORT	NUMERIC*	OFF	OLD
ONLY	OPEN*	OPTIMAL	OWN	PACKAGE
PARALLEL	PASCAL*	PCTINCREASE	PLAN	PLI*
PRECISION*	PRIMARY*	PRIVATE	PROCEDURE*	PROFILE
QUOTA	READ	REAL*	RECOVER	REFERENCES*
REFERENCING	RESETLOGS	RESTRICTED	REUSE	ROLE
ROLES	ROLLBACK*	SAVEPOINT	SCHEMA*	SCN
SECTION*	SEGMENT	SEQUENCE	SHARED	SNAPSHOT
SOME*	SORT	SQLCODE*	SQLERROR*	STATEMENT_ID
STATISTICS	STOP	STORAGE	SUM*	SWITCH
SYSTEM	TABLES	TEMPORARY	THREAD	TIME
TRACING	TRANSACTION	TRIGGERS	TRUNCATE	UNDER

UNLIMITED	UNTIL	USE	USING	WORK*
WRITE	TABLESWHENPAC	NAME		

7.5 Code Table Naming

See NRS Code Management StandardsSpatial Logical Naming Standards

Spatial layers follow general naming conventions for (class/entity) objects, whether SDO geometry is used or not, unless the SDO geometric layer is to be used independent of ESRI SDE. The layer must be of one and only one feature type and have the class or suffix of SP in the logical name (unless the suffix POINT, POLY, LINE or ANNO is used). When generated to physical the table will have the application acronym as the prefix and the suffix _SP or the geometric suffixes listed above. The name decided upon between these mandatory extensions has to indicate the geometry type with no ambiguity (POINT, POLY, LINE or ANNO can be used if the name of the layer is not clear as to its geometry). A spatial layer must have in its description, an indication of whether the layer is a multipart feature or if it is not a multipart feature. Example (class/entity) object description: “The spatial layer STRUCTURAL BRIDGE SP is a multipart line feature, that shows water and land crossings." or " The spatial layer STRUCTURAL BRIDGE SP is not a multipart line feature, that shows water and land crossings ."

If a spatial generalize layer is needed for one of the above spatial layers, the suffix _SPG is used. A spatial generalized layer is spatial object that is weeded of points so that it will display faster. It is primarily only used for IMF (Internet Mapping Framework) applications, is only used for display, and has access restricted to IMF.

7.6.1 ESRI SDE Spatial Attribute Naming & Describing

The four mandatory attributes in SDE spatial layers or SDE spatial views will be called and described as follows:

1 A unique identifier follows these naming and describing conventions. The unique identifier or surrogate key is used to make sure all records are unique within the on the spatial tile. By convention it carries the name of the table without the prefix and the class suffix ID. If the name of the table has a suffix of SP, the unique key should not have it in its name. The unique identifier requires a sequence if it is being populates on load with a trigger. The sequence is the same name as the table and has the suffix SEQ; if the table name is too long the short name may be used with the application prefix. If a unique business identifier in a coverage is available it should be used instead of the surrogate ID and its sequence. If a unique business identifier is not available and a sequence is used, the description of the class/entity/table must contain in it a warning that the identifier cannot be used to join to other objects if the replication method truncates the table and the rows are re-sequenced on reload. The description for the unique identifier follows the rules in

Section 6.4

2 OBJECTID of type (NUMBER (38) OPTIONAL). The description for OBJECTID as standard is: " OBJECTID is a required element for feature classes and object classes in a geodatabase. This field is added to a SDE layer that was not previously created as part of a geodatabase but is now being registered with a geodatabase."

NOTE: The attribute OBJECTID is not driven to the physical model.

3 GEOMETRY of type (NUMBER (38) OPTIONAL). The description for GEOMETRY as standard is: " GEOMETRY provides an indication of ArcSDE spatiality or coordinates defining the features."

4 FEATURE CODE of type (VARCHAR (10) OPTIONAL). The description for FEATURE CODE is "The FEATURE CODE is most importantly a means of linking a feature to its name and definition. For example, the code GB15300120 on a digital geographic feature links it to the name " Lake - Dry" with the definition "A lake bed from which all water has drained or evaporated." The feature code does NOT mark when it was digitized, what dataset it belongs to, how accurate it is, what it should look like when plotted, or who is responsible for updating it. It only says what it represents in the real world. It also doesn't even matter how the lake is represented. If it is a very small lake, it may be stored as a point feature. If it is large enough to have a shape at the scale of data capture, it may be stored as an outline, or a closed polygon. The same feature code still links it to the same definition."
or
FEATURE CLASS SKEY (VARCHAR (10) OPTIONAL). The description for FEATURE CLASS SKEY is "A FEATURE CLASS SKEY is the unique key assigned to a Feature Class by the Ministry of Forests where a Feature Class is used to define a class of geographic items having the same basic set of characteristics. This is a spatial (class/entity) object or logical business grouping of spatial information much like a textual entity is a logical business grouping of attribute information. All features have a topology type of Polygon, Line or Point. Some examples of feature class are Bridge, Road, Activity Treatment Unit, and Lake. A "feature" differs from a "feature class" in that the feature is an instance of feature class. For example “Lake” or "Forest License Cut block" are feature classes. The features class "Lake” (and associated data) describes the standards to which all lakes are captured. Nitinaht Lake is a feature within the Lake feature class and would be captured to the standards within that class (e.g. Lake is a polygon feature)."

7.6.2 Oracle SDO Spatial Attribute Naming & Describing

There are two ways to define mandatory attributes for a SDO spatial layer, each method depends on the deployment and use of the SDO layer. The two methods will be defined here as Standalone and SDE over SDO.

SDE over SDO:

All the standard mandatory attributes in 7.6.1 apply to a spatial layer that uses the SDO geometry data type, if ESRI SDE is going to be used to display and/or manipulate the layer.

If Oracle spatial is being used to do PLSQL spatial queries and the database is the Oracle Locator version of Oracle Spatial, then it is recommended to add two data elements called by standard FEATURE_AREA and FEATURE_PERIMETER and populate them, on load with triggers.

The description for FEATURE AREA by standard is "FEATURE AREA contains the calculated area of the spatial feature, in Hectares.   It stores this area for efficiency, because Oracle Locator does not support area calculation on the fly."

The description for FEATURE PERIMETER by standard is "FEATURE PERIMETER contains the calculated perimeter of the spatial feature, in Meters.   It stores this perimeter for efficiency, because Oracle Locator does not support perimeter calculation on the fly."

Standalone:

If the Architecture of an application dictates that only Oracle spatial is used, and there is no interaction with SDE, then a spatial layer may have multiple geometries. The Naming convention for these geometric data elements is "<geometry type>_GEOMETRY". So an example name would be POLYGON_GEOMETRY. The description for these geometries follows the form: " <geometry type> GEOMETRY provides an indication of SDO spatiality or coordinates defining the features." The layer itself need not have a indication of spatiality in the name but must be suffixed with SP.

For standalone SDO applications all attributes mentioned in Section 7.6.1 are mandatory for SDO spatial layers, with the exception of the attribute OBJECTID.

8 Agriculture/Environment/ILMB Oracle Designer Naming

This chapter is from the Ministries of Agriculture, Environment and ILMB

For any development, done in the Oracle Designer repository, the following standards apply. This section applies specifically to naming of properties within Designer.

NOTE: For the naming of any Oracle Objects within Designer, see section above.

8.1 Application Properties

Logical data models are generated by identifying each (class/entity) object, along with its attributes, or characteristics, that the business records information about. An (class/entity) object is a person, place, thing, event, or concept about which the ministry is interested and facts are gathered and recorded. (Class/entity) objects are then transformed to physical tables. This is the basis of an application in Oracle Designer, which is known as an Application container. Below are some specific standards for naming and describing this container. The Application Properties will follow the general naming and describing standards in Section 5 & 6 above.

8.1.1 Name Property

Application Name: This is the application acronym. See Section "5.4 Application Acronym Derivation" for very specific instruction on creating an Application name.

It is entered in Designer as the Name Property. The application acronym becomes the repository container name.

If the container is a warehouse only model, it gets the suffix _WHSE.

The underbars in an Application container name signifies only specific model information related to the name of the object. This is only design information pertaining to the type of model and is not used anywhere except for the repository model name for organization. For example CORP_PERSON_ORG(reference) is a Oracle repository model that enterprise name is just CORP. All the information after the first under bar is extra information. Anything in brackets at the end of an Application container name is just extra information)

8.1.2 Title Property

Application Title: This is the full name of the application. Usually the Application Acronym is derived from this. Typically the first letters of the Title become the application acronym. See Section "5.4 Application Acronym Derivation" for very specific instruction on creating an Application title.

It is entered in Designer as the Title Property.

8.1.3 Documentation Group Properties

Summary, Objectives, and Description: This is important information about the Oracle Repository Application Container itself. These fields follow the general Rules of describing. See Section 5 “General Describing” for instruction on describing these fields.

Notes: This field is used by Data Architects for change history of the data model and to provide any extra information that may be useful about the data model. As a standard all Notes in this field should start with the date and data architect’ name. The suggested format is August 11, 2007 (R. Hoffner) - < note body>.

8.2 Other ER specific Entity Properties

Entity Short Name:

- An identifier is up to 10 characters in length. Ideally it is the first letter of the words that make up the name of the Entity.
- The short name property of the entity must contain the application prefix or acronym, so that all objects or references resulting from the transformation are prefixed.
- The entity short name property must not contain an Oracle Reserved Word
- It is all capitals and no underbars
- When sub typing in Oracle Designer the name applied in the short name property of entity for the subtype, becomes one of the type names for the column when transformed. This is to be one or two English words.

Entity Plural Name:

- Used to determine the physical table name
- The unique identifier attribute name (if surrogate) is derived from this field minus plurality and the application prefix. This also applies to the unique attribute for code tables.
- It is a name up to 30 characters in length, derived from the entity name.
- The entity name itself, must be 24-25 characters or less to allow for prefixing and plurality in the plural name property of designer.
- The plural name property of the entity must contain the application prefix or acronym, so that objects resulting from the transformation are prefixed.

  Note: This is the only way to apply multiple prefixes in a transform in Oracle Designer, when required.

- The entity plural name (physical table name) must be plural in context.
  (PROJECT becomes PROJECTS)
- The entity plural name must not contain an Oracle Reserved Word
- It is all capitals and no underbars

- When sub typing in Oracle designer the name applied in the plural name property of entity for the subtype, is usually not used when transformed, since it is not driven to a table but a type column. The name does not include the application prefix and is the same name as the subtype entity by standard.

Class/Entity Unique Identifiers Name:

- It must be comprised of attribute(s) and/or relationship(s) that are defined for the entity. This is to ensure business uniqueness.
- Contains the application acronym as the prefix
- Compound primary keys should be non-volatile and are advised against for applications.
- The preferred method is to create a unique single primary key and bring the rest of the relationships down as a unique key.

Appendix A - Forests Standard Approved Abbreviations

The following abbreviations are to be used if you need to abbreviate the name of an entity/table or attribute/column. As a general rule, spell out the words in full until the 30 character limit is met, and abbreviate only if needed.

ADDRESS	ADDR
ADMINISTRATION	ADMIN
ALTERNATE	ALT
AMOUNT	AMT
AMERICAN	USA
APPLICATION	APPL
AUTHORITY	AUTH
AVERAGE	AVG
BREASTHEIGHT	BH
BUSINESS	BUS
CANADIAN	CDN
CATEGORY	CAT
CLASSIFICATION	CLASS
CLIENT	CLI
COLLECTION	CLCTN
COLUMN	COL
COMMENT	CMT
COMMISSION	COMM
COMMITTEE	CTTE
COMPANY	CO
CONDITION	CONDTN
CONTROL	CTL
CONVERSION	CNV
COORDINATE	COORD
CORPORATION	CORP
CORRECTION	CRCTN
COUNT	CNT
CREDIT	CR
DATE (Gregorian Date)	DT
DAY	DY
DESCRIPTION	DESC
DESTINATION	DEST
DEPARTMENT	DEPT
DETAIL	DTL
DEVELOPMENT	DEV
DIAMETER	DIAM
DIAMETER AT BREAST HEIGHT	DBH
DIAMETER INSIDE BARK	DIB
DIAMETER OUTSIDE BARK	DOB

DISTURBANCE	DISTRB
DOUBLE BARK THICKNESS	DBT
DISTRICT (Forest District)	DIST
DIVISION	DIV
DOCUMENT	DOC
EFFECTIVE	EFF
ELEMENT	ELMNT
ERROR	ERR
ESTIMATE	EST
EXECUTIVE	EXEC
EXPIRY	EXP
FACTOR	FCTR
FEDERAL	FED
FORESTS AND RANGE(Ministry Of)	FOR
GROUP	GRP
HARVEST	HARV
HECTARES	HA
HEIGHT	HGHT
HOUR	HR
IDENTIFICATION	ID
INDEX	INDX
INDICATOR	IND
INITIAL	INIT
INVENTORY	INV
JURISDICTION	JURIS
LATITUDE	LAT
LENGTH	LEN
LETTER	LTR
LICENCE	LIC
LOAD	LD
LOCATION	LOCN
LONGITUDE	LONG
MANAGEMENT	MGT
MAXIMUM	MAX
METERS CUBED	M3
METERS SQUARED	M2
METHOD	MTHD
MINIMUM	MIN
MINUTE	MN
MONTH	MO
NAME	NM
NUMBER	NO
ORGANIZATION	ORG
PAYMENT	PAY
PERCENT	PCT
PERMIT	PRMT
PIECE	PCE
POSITION	POS

PREVIOUS	PREV
PRIMARY	PRI
PRODUCT	PROD
PROJECT	PROJ
QUANTITY	QTY
RECEIVED	RECV
REFERRED	REF
REGION (Forest Region)	REG
REGISTRATION	REGN
RESPONSE CENTRE	RC
REQUEST	RQST
REQUIRED	REQ
REQUIREMENT	RQMT
RETURN	RET
REVENUE	REV
SCHEDULE	SCHED
SCREEN	SCR
SEARCH	SRCH
SECONDARY	SEC
SECTION	SECT
SEQUENCE	SEQ
SERVICE	SRVC
SILVICULTURE	SILV
SOURCE	SRCE
SPECIES	SPP
STATEMENT	STMT
STATUS	STS
STATUTORY	STAT
STATISTICS	STATS
TENURE	TENR
TEXT	TXT
TIMBER	TMBR
TIMBER SUPPLY AREA	TSA
TIMBER SUPPLY BLOCK	TSB
TIME	TM
TIMESTAMP	TS
TITLE	TTL
TOTAL	TOT
TRANSACTION	TXN
TREATMENT	TRTMT
TYPE	TYP
USERID	UID
VALUE	VAL
VERSION	VER
VISITATION	VISIT
VOLUME	VOL
WITHDRAWAL	WD
WEIGHT	WGT

XREF	XF
YEAR	YR
YEAR-TO-DATE	YTD

Appendix B - Agriculture/Environment/ILMB Approved Abbreviations List

ADMINISTRATION	ADMN
ALTERNATE	ALT
APPLICATION	APPL
AUTHORITY	AUTH
BRANCH	BR
BUSINESS	BUS
CANADIAN	CDN
CLASSIFICATION	CLASSN
CLIENT	CLNT
COMMENT	CMNT
COMMITTEE	CTTE
COMPANY	CMPNY
CONTROL	CTL
CORPORATION	CORP
CREDIT	CR
DESTINATION	DEST
DEPARTMENT	DEPT
DISTRICT	DIST
DIVISION	DIVN
EFFECTIVE	EFF
ERROR	ERR
ESTIMATE	EST
EXECUTIVE	EXEC
FEDERAL	FED
HECTARES	HA
IDENTIFICATION	ID
INDEX	INDX
INITIALS	INTLS
INDICATOR	IND
LENGTH	LNTH
LICENCE	LIC
LOAD	LD
LOCATION	LOC
MANAGEMENT	MGMT
MARK	MRK
METHOD	MTHD
MINISTRY	MNSTRY

NAME	NM
ORGANIZATION	ORG
PAYMENT	PYMNT
PERCENT	PCT
PERMIT	PRMT
PIECE	PCE
POSITION	POSN
PRIMARY	PRI
PRODUCT	PRO
PROJECT	PROJ
RECEIVED	RECD
REGION	REG
REGISTRATION	REGN
REQUIRED	REQD
RETURN	RTN
REVENUE	RVNU
SCHEDULE	SCHEDL
SEARCH	SRCH
SECONDARY	SEC
SECTION	SCTN
SEQUENCE	SEQ
SERVICE	SRVC
SQUARE FEET	SQFT
SQUARE INCHES	SQIN
SQUARE METERS	SQM
STATEMENT	STMT
STATUS	STS
STATUTORY	STAT
STATISTICS	STATS
TEXT	TXT
TRANSACTION	TXN
TYPE	TYP
USERID	UID
VALUE	VAL
YEAR	YR
YEAR TO DATE	YTD
British Columbia System Corporation	BCSC
Social Insurance Number	SIN