Data Naming Standards for System Development - Guide S19

Purpose of Document

This guide 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 at the Ministry of Forests and Range.

The guide is intended for systems analysts and systems designers who have some knowledge of the techniques and procedures involved in information engineering methodology in general and data modelling in particular (see Guide S7: Data Modelling Guide for Relational Systems). Note that this guide deals only with data naming standards for data attributes and physical data columns. It does not provide standards for program variable names and other programming naming standards.

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.

If you have any questions, see the staff in the Data Administration group. All logical and physical names must be approved by Data Administration staff prior to any tables being built.

Naming Standards Are Evolutionary

The naming standards outlined in this guide evolved over about a year's time during 1989, and continue to evolve and change (we hope for the better!). There exist many table and column names in our current databases that do not conform to the current standard, simply because the standards have become better, and we had to implement something back then even when we didn't have a finalized standard.

These kinds of grandfathered names we will all have to live with as a fact of life until we have the money and resources to fix everything. Again, standards tend to change over time and what is current may not be so a few years from now. This guide is the best place to see what the current standard is.

Document Status

Although this is a "final" document it is updated periodically, so suggestions for improvement and/or clarification are welcome. Contact any member of the Data Administration group if you have any comments.

Acknowledgement

This document contains excerpts from the Naming Standards document produced by Blane Hill, Data Base Application Services, BC Systems Corporation, in 1987. There
have been changes and additions made from the original document to accommodate differences in rule application between BCSC and the Ministry of Forests and Range.

Back to top

**The Need for Naming Standards**

The size of the ministry, its applications, and the high degree of information exchange between all parts of the ministry dictates the need for highly organized systems development. The names used to identify facts about objects, concepts, and events are critical to the general understanding and knowledge of ministry 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. This document presents a convention for naming system components that is precise, logical, and conforms with the Ministry of Forests and Range 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.

**Logical vs. Physical Data Model Names**

For a more complete discussion on the differences between Logical and Physical Data Models, refer to [Guide S7: Data Modelling Guide for Relational Applications](#).

**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 in Oracle Designer is used to generate the physical table name and column names, a limit of 30 chars is recommended for entity names and data attribute names including spaces.

**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.

Physical model names are restricted to physical storage limits (see the section titled
 Physical Model Data Names in this document) 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 entity names, must be unique
 throughout the ministry.

Example: Differences between Logical and Physical model names

<table>
<thead>
<tr>
<th>Logical Model Name</th>
<th>Physical Model Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>FOREST CLIENT</td>
<td>FOREST_CLIENT</td>
</tr>
<tr>
<td>SEED GERMINATION MEASUREMENT</td>
<td>SEED_GERMINATION_MEASUREMENT</td>
</tr>
</tbody>
</table>

General Structure of Data Names

Logical data models are generated by identifying each entity, or object, along with its
attributes, or characteristics, that the business records information about. An entity
(object) 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 (see the section below titled General Format).

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

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

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.

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; and
- 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).

**General Format**

As mentioned earlier, each entity name must be unique throughout the 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 an underscore ("_") - 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
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 |

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 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.

Appendix B 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 Appendix A 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 Guide S21: Use of Standard Code Tables in Relational Applications) 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 Appendix B, 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

**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 Administration group prior to any tables being built.

**PC/LAN Server Restrictions**

Follow the guidelines listed above for PC or LAN Server applications.

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

---

**Appendix A - Conventions**

**Suffix Conventions**

<table>
<thead>
<tr>
<th>Entity/Table or Attribute</th>
<th>Mandatory Abbr</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORG_UNIT</td>
<td>_DIST or _REG or _ORGU</td>
</tr>
</tbody>
</table>

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**

<table>
<thead>
<tr>
<th>Physical DB Implementation Names</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequences</td>
<td>_SEQ</td>
</tr>
<tr>
<td>Indexes</td>
<td>_I</td>
</tr>
<tr>
<td>Primary Keys Constraint Names (not on data columns)</td>
<td>_PK</td>
</tr>
<tr>
<td>Foreign Key Constraint Names (not on data columns)</td>
<td>_FK</td>
</tr>
<tr>
<td>Views</td>
<td>_VW</td>
</tr>
</tbody>
</table>

**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.

**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 these dates.

**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

**Status Codes**

All status codes within the ministry are defined to a particular standard, displayed in 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.).
**Multiple View Use**

If multiple views are required for a table (for example where some columns are restricted from public access -- see Guide S7: Data Modelling Guide for Relational Applications, section Physical Data Model, subsection Managing Views), 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.**

**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.

**Appendix B - 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.

<table>
<thead>
<tr>
<th>Address</th>
<th>Abbreviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADDRESS</td>
<td>ADDR</td>
</tr>
<tr>
<td>ADMINISTRATION</td>
<td>ADMIN</td>
</tr>
<tr>
<td>ALTERNATE</td>
<td>ALT</td>
</tr>
<tr>
<td>AMOUNT</td>
<td>AMT</td>
</tr>
<tr>
<td>AMERICAN</td>
<td>USA</td>
</tr>
<tr>
<td>APPLICATION</td>
<td>APPL</td>
</tr>
<tr>
<td>AUTHORITY</td>
<td>AUTH</td>
</tr>
<tr>
<td>AVERAGE</td>
<td>AVG</td>
</tr>
<tr>
<td>BREASTHEIGHT</td>
<td>BH</td>
</tr>
<tr>
<td>BUSINESS</td>
<td>BUS</td>
</tr>
<tr>
<td>CANADIAN</td>
<td>CDN</td>
</tr>
<tr>
<td>CATEGORY</td>
<td>CAT</td>
</tr>
<tr>
<td>Description</td>
<td>Abbreviation</td>
</tr>
<tr>
<td>----------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>Total</td>
<td>TOT</td>
</tr>
<tr>
<td>Transaction</td>
<td>TXN</td>
</tr>
<tr>
<td>Treatment</td>
<td>TRTMT</td>
</tr>
<tr>
<td>Type</td>
<td>TYP</td>
</tr>
<tr>
<td>Userid</td>
<td>UID</td>
</tr>
<tr>
<td>Value</td>
<td>VAL</td>
</tr>
<tr>
<td>Version</td>
<td>VER</td>
</tr>
<tr>
<td>Visitation</td>
<td>VISIT</td>
</tr>
<tr>
<td>Volume</td>
<td>VOL</td>
</tr>
<tr>
<td>Withdrawal</td>
<td>WD</td>
</tr>
<tr>
<td>Weight</td>
<td>WGT</td>
</tr>
<tr>
<td>Xref</td>
<td>XF</td>
</tr>
<tr>
<td>Year</td>
<td>YR</td>
</tr>
<tr>
<td>Year-to-date</td>
<td>YTD</td>
</tr>
</tbody>
</table>

[Back to top](#)
[Back to Data Administration Section](#)