**DRAFT**

Database Coding Standard

##### 

Last Updated: **Sptember 05, 2013**

Version: 0.1

**TABLE OF CONTENTS**

1. Preface 3

1.1. Authors 3

1.2. Contributors 3

1.3. History 3

1.4. Document Approvals 3

1.5. Restricted Rights 3

1.6. Trademarks Notice 3

1.7. Copyright Notice 3

2. Introduction 4

2.1. Terminology 4

2.2. Objective 4

2.3. Scope 4

3. Naming Conventions 4

3.1. Packages 4

3.2. Procedures 4

3.3. Functions 5

3.4. Parameters 5

3.5. Variables 6

3.6. PL/SQL Types 6

3.7. Cursors 6

3.8. Rowtypes 7

3.9. Tables 7

3.10. Views 7

3.11. Materialized Views 8

3.12. Columns 8

3.13. Sequences 8

3.14. Triggers 9

3.15. Indexes 9

3.16. Constraints 9

3.17. Schemas 9

4. System Standards 10

5. Coding Standards 10

6. Best Practices 11

6.1. Deferred Requirements 12

# Preface

## Authors

## Contributors/Reference

Commercial Workbench Database Team

## History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version Number** | **Date Changed** | **Author of Change** | **Purpose of Change** |
|  |  |  |  |

## Document Approvals

|  |  |  |
| --- | --- | --- |
| **Stakeholders** | **Date** | **Signature** |
|  |  |  |

## Restricted Rights

This document contains proprietary and confidential information that is the property of Comcast Corporation. It is protected by copyright and distributed under licenses restricting its use, copying, and distribution. No part of this document may be reproduced in any form without prior written authorization from Comcast Corporation.

## Trademarks Notice

All trademarks are acknowledged.

## Copyright Notice

**© Comcast Corporation 2009**

No part of this publication may be reproduced, transmitted, transcribed stored in any retrieval system or translated into any human or computer language without the express prior permission of Comcast Corporation.

# Introduction

## Terminology

The following is a list of terms used in this document in describing the design.

|  |  |
| --- | --- |
| **Term** | **Definition** |
|  |  |
|  |  |

## Objective

Standardise development in order to create systems that are easily readable, maintainable and have a minimal amount of errors.

## Scope

This document includes naming conventions, development standards and best practices for oracle database for commercial workbench application . This document provides developers with standards for use during development and code review

# Naming Conventions

This section descriobes naming standards for all Oracle and PL/SQL objects.It gives details for the general style,prefixes and suffixes for naming each object

## Packages

Prefixes:

* pkg

Name:

* First letter lowercase
* No Underscores
* For every new word, prefix, suffix or abbreviation the first letter will be uppercase
* Ex: pkgInventoryFeed

## Procedures

Prefixes:

* prc

General Style:

* First letter lowercase
* No Underscores
* For every new word, prefix, suffix or abbreviation the first letter will be uppercase
* Ex: prGenerateId

Formatting:

* Declaration of a proc should have the proc name on one line then the first parameter should be on the next line and indented by one tab. Also, the parameter names should line up.
  + Ex: Procedure procName

<tab>(param1…,

…);

* Call to a proc should have the proc name and the first param on the same line, with every other param on its own line. Also, the parameter names should line up.
  + Ex: procName(param1,

param2);

## Functions

Prefixes:

* fn

General Style:

* First letter lowercase
* No Underscores
* For every new word, prefix, suffix or abbreviation the first letter will be uppercase
* The name shall state the data being returned
* Ex: fnGetData

Formatting:

* Declaration of a function should have the function name on one line then the first parameter should be on the next line and indented by one tab. Also, the parameter names should line up.
  + Ex: Function functionName

<tab>(param1…,

…)

<tab>Return Number;

* Call to a function should have the function name and the first param on the same line, with every other param on its own line. Also, the parameter names should line up.
  + Ex: functionName(param1,

param2);

## Parameters

Prefixes:

* pi – for In
* po – for Out
* pio – for In Out

General Style:

* First letter lowercase
* No Underscores
* For every new word, prefix, suffix or abbreviation the first letter will be uppercase
* Ex: piParamName

Formatting:

* Explicitly designate each parameter as IN, OUT or IN OUT.
  + Each parameter should be on its own line
* The flow of the parameter (IN/OUT/IN OUT) shall always be lined up with all parameters in the object declaration
* The datatype of the parameter shall always be lined up with all parameters in the object declaration
* Declare the parameter as
  + <parameter name> tab {IN/OUT/IN OUT} tab <datatype>

## Variables

Prefixes:

* k – for any Constant no matter the datatype
* v – for a Variable of Varchar2, Char, Number, Integer or any variation
* b – for a Boolean
* e – for an Exception
* gk – Global Constant
* gv – Global Variable
* gb – Global Boolean
* ge – Global Exception
* gu – Global Utility

General Style:

* First letter lowercase
* No Underscores
* For every new word, prefix, suffix or abbreviation the first letter will be uppercase
* Ex: vVariableName
* Ex: gkConstantName

## PL/SQL Types

Use these rules for all Type objects and all Type instances.

Prefixes:

* t – for any type, i.e. table, varray.
* gt – global Type

General Style:

* First letter lowercase
* No Underscores
* For every new word, prefix, suffix or abbreviation the first letter will be uppercase
* This naming convention will be used for the object declaration and the instance declaration.
* Ex: tTypeName

## Cursors

Prefixes:

* c – for all declared cursors
* rc – for all declared ref cursors
* gc – global cursor
* grc – global ref cursor

General Style:

* First letter lowercase
* No Underscores
* For every new word, prefix, suffix or abbreviation the first letter will be uppercase
* Within the cursor query, each column shall be explicitly stated. No SELECT \*.
* Ex: cCursorName

## Rowtypes

Prefixes:

* r – for all rowtypes
* gr – Global rowtype

General Style:

* First letter lowercase
* No Underscores
* For every new word, prefix, suffix or abbreviation the first letter will be uppercase
  + The description after the prefix should be the same as the description of the object for which it is holding data (i.e. table, cursor). If cursor name is cCursor, then the rowtype name should be rCursor.
* Ex: rCursorName

## Tables

Prefixes:

* tbl
* stg – for stage tables
* tmp – for true temporary tables
* bak – for backup tables

General Style:

* No Underscores
* All letters Uppercase
* All table names shall be singular
* All tables will have a column called ID, which will be its Primary Key.
* All tables will have an abbreviation, which is based on each syllable of the table name, i.e. TBLTABLENAMEONE = TNO.
* Ex: TBLTABLENAME

## Views

Prefixes:

* vw

General Style:

* No Underscores
* All letters Uppercase
* All view names shall be singular
* Ex: VWVIEWNAME

## Materialized Views

Prefixes:

* mv

General Style:

* No Underscores
* All letters Uppercase
* All materialized view names shall be singular
* MVVIEWNAME

## Columns

General Style:

* First letter lowercase
* No Underscores
* For every new word, prefix, suffix or abbreviation the first letter will be uppercase
* Do NOT use the table name as part of a column name
* The length of the column name will be 20 characters or less because we must allow for that column name to be used within other object names, i.e. constraints, sequences, indexes.
* Use the hierarchical approach to create all Foreign Key column names.
  + Ex: TBLTABLENAMEONE.id is a foreign key named TBLTABLENAMETWO.tnoID
* Use abbreviations for date columns
  + Ds – Date Stamp
  + Ts – Time Stamp
  + Dts – Date and Time Stamp
  + Ex: createDts

## Sequences

Prefixes:

* seq – for all sequences

General Style:

* First letter lowercase
* No Underscores
* For every new word, prefix, suffix or abbreviation the first letter will be uppercase
* Use the hierarchical approach to create the rest of the name
* Ex: seqTnColumnOne for TBLTABLENAME.columnOne).

## Triggers

Prefixes:

* trg

General Style:

* First letter lowercase
* No Underscores
* For every new word, prefix, suffix or abbreviation the first letter will be uppercase
* The rest of the trigger name should be the table name on which the trigger is owned
* Ex: trgTblTableNameBiud

Suffixes:

* b – before
* a - after
* i – insert
* u – update
* d – delete

## Indexes

Prefixes:

* idx – oracle standard

General Style:

* First letter lowercase
* No Underscores
* For every new word, prefix, suffix or abbreviation the first letter will be uppercase
* Use the hierarchical approach along with a combination of the columns involved to create the rest of the name
* Ex: idxTnColOneColTwo for TBLTABLENAME.columnOne and TBLTABLENAME.columnTwo).

## Constraints

Prefixes:

* fk – for Foreign Keys
* pk – for Primary Keys
* unq – for Unique
* nn – for Not Null
* chk – for Check

General Style:

* First letter lowercase
* No Underscores
* For every new word, prefix, suffix or abbreviation the first letter will be uppercase
* Use the hierarchical approach to create the rest of the name
* Ex: nnTnColumnOne for TBLTABLENAME.columnOne

## Schemas

General Style:

* First letter lowercase
* No Underscores
* For every new word, prefix, suffix or abbreviation the first letter will be uppercase
* Create the schema name based on an abbreviation of the application
* Ex: ordSys for Order System

# System Standards

1. Do not use any reserved words as complete names (i.e. Type, Name).
2. Add a comment for every table and for every column in every table.
3. Define all datatypes for every column in every table based on the data being stored. Do NOT just use Varchar2. Understand the data being stored and create the datatype accordingly.
4. For Boolean values in a table use Char(1) along w/a check constraint, which only allows ‘Y’ and ‘N’.

# Coding Standards

1. There must be a block of comments for each Package Body and Package spec. Also there must be a block of comments before every Procedure and Function within each package. This block must include the following:

* Purpose
* Person who made the update
* Date the change occurred
* Comments on the change.

1. Always using the schema name in front of all database objects (i.e tables, packages) with the dot notation.
   1. Ex: schemaName.TBLTABLENAMEONE
2. For all Select statements, these standards apply
   1. Every column in the select clause shall be on its own line
   2. Every table in the From clause shall be on its own line
   3. Every condition in the Where clause shall be on its own line
   4. Every column in the Order By clause shall be on its own line
   5. Every column in the Group By clause shall be on its own line
   6. Every condition in the Having clause shall be on its own line
   7. When using the ANSI standard the Join statement shall be on its own line, the On statement shall be on its own line and any extra condition shall be on its own line
   8. Ex 1: SELECT MAX(colOne) colOne,

MAX(colTwo) colTwo

FROM TBLTABLEONE to

JOIN TBLTABLETWO tt

ON to.ID = tt.ttID

WHERE colOne = 1

OR colTwo = 2

GROUP BY colOne,

colTwo

HAVING count(colOne) > 1;

* 1. Ex 2: SELECT MAX(colOne) colOne,

MAX(colTwo) colTwo

FROM TBLTABLEONE to,

TBLTABLETWO tt

WHERE to.ID = tt.ttID

AND colOne = vValOne

OR colTwo = vValTwo

GROUP BY colOne,

colTwo

HAVING COUNT(colOne) > kOne;

1. For all Insert statements, these standards apply
   1. Every column defined for the Insert statement shall be on its own line
   2. Every value being stored to the table shall be on its own line
   3. Ex: INSERT INTO TBLTABLEONE

(colOne,

colTwo)

VALUES (vValOne,

vValTwo);

1. For all Update statements, these standards apply
   1. The Update and Set clauses shall start on their own lines
   2. Every column being updated shall be on its own line
   3. Every condition in the Where clause should be on its own line
   4. Ex: UPDATE TBLTABLEONE

SET colOne = vValOne,

colTwo = vValTwo

WHERE colThree = vValThree;

1. For all Delete statements, these standards apply
   1. Every condition in the Where clause should be on its own line
   2. Ex: DELETE FROM TBLTABLENAME

WHERE colOne = vValOne;

1. Use a column definition by using %TYPE for all possible objects (i.e. variables, types, parameters). This allows for easier readability. Also, it makes for simpler maintenance when the datatype or column length changes for that column.

# Best Practices

* + 1. Never exit from a FOR loop (numeric or cursor) with an EXIT or RETURN statement. A FOR loop is a promise: my code will iterate from the starting to the ending value and will then stop execution.
    2. Never exit from a WHILE loop with an EXIT or RETURN statement. Rely solely on the WHILE loop condition to terminate the loop.
    3. Ensure that a function has a single successful RETURN statement as the last line of the executable section. Normally, each exception handler in a function would also return a value.
    4. Don't let functions have OUT or IN OUT parameters. The function should only return value(s) through the RETURN clause.
    5. Make sure that the name of a function describes the value being returned (noun structure, as in "totalCompensation"). The name of a procedure should describe the actions taken (verb-noun structure, as in "calculateTotals").
    6. Do not use exceptions to perform branching logic. When you define your own exceptions, these should describe error situations only.
    7. Do not "SELECT COUNT(\*)" from a table unless you really need to know the total number of "hits." If you only need to know whether there is more than one match, simply fetch twice with an explicit cursor.
    8. Do not use the exact same names of tables or columns for variable names. This can cause compile errors. It can also result in unpredictable behavior inside SQL statements in your PL/SQL code.
    9. Avoid searching using not equals operators (<> and NOT) as they result in table and index scans