6.3 Use the Project Code as a prefixed for the Table Name

Every table should be prefixed with the Trigram code for the project. If the initial project code is defined as MMS, table should be named MMS_<FUNCTIONAL_MEANINGFULL_NAME>. Eg. MMS_COMPONENT

6.4 Table names

Table names should be singular, this to easy the development, so you don't have to think about what the plural is of the name. Example: use MMS_COMPONENT instead of MMS_COMPONENTS

6.5 Use Primary Keys for each table

Every table should have its own primary key. The primary key of the table is the column "id", a number column for which the value is set by a sequence. A trigger on the table sets the value of the primary key.

6.6 Use of audit columns for each table

Every table should use CREATED_BY (char), CREATED_DATE (date), MODIFIED_BY (char) and MODIFIED_DATE (date) columns.

This allows auditing on table level so that we know who is adding/changing the table data. Columns should be filled by a trigger. See the example further in the document on how to populate the fields using a trigger.

6.7 Use a Unique Key where possible

If a non PK column is unique, a Unique Key should be placed on the column(s).

6.8 Index all foreign keys (unless you use Oracle Exadata machines)

If for a table a column is reffering to a column in another table it should be indexed. This allows better performance when selecting using the foreign key.

The only exception is when your hardware is an Oracle Exadata machine, then indexes should not be used as Exadata has such big memory that the entire table gets cached. In this case indexing will even slow down execution of the queries.

6.9 Table Aliasses - Naming Conventions

If an alias is used for a table, following conventions should be met:

3 letters

Tablename consisting of one word with only one syllable:

1st letter of the word
a logical letter in the word, preferably a consonant
last letter of the word (singular)

Example: mms_field => fld (table names should be singular)

Tablename consisting of one word with more than one syllable:

1st letter of the word
1st letter of the 2nd syllable of the word
last letter of the word (singular)

Example: mms_component => cpt

Tablename consisting of more than one word:

1st letter of the 1st word
1st letter of the 2nd word
last letter of the last word (singular)

Example: mms_main_assembly => may

If an alias already exists in a schema, a substitute (logical) letter can be chosen to create a unique combination.

Intersection tables:

<project code>_<table alias1>_<table alias2>

alias of intersection table according to the alias naming conventions

Example: the intersection table for tables mms_component_type (cte) and mss_component_attribute (cae) is:

mms_cte_cae with alias cce

For regular tables as well as for intersection tables, the alias is used in sequence, constraints etc.

6.10 Primary Keys - Naming Conventions

Primary Keys should be named as follows:

<project code>_<table alias>_pk

Example: mms_cpt_pk for mms_component (cpt)

6.11 Unique Keys - Naming Conventions

Unique Keys should be named as follows:

<project code>_<table alias>_uk <number>

Example: mms_cpt_uk1 for mms_component (cpt)

6.12 Foreign Keys - Naming Conventions

Foreign Keys should be named as follows:

<project code>_<table alias primary>_<table alias secondary>_fk <number>

The number refers to the secondary table. If a second foreign key is created for the same primary table to the same secondary table, then the number is 2.

Example: mms_cpt_cte_fk1 for foreign key of mms_component (cpt) referring to mms_component_type (cte)

If it concerns a foreign key to a table in another schema, the shema should also be incorporated

Example: mms_fld_hoe_tdr_fk1 for foreign key of mms_field (fld) referring to hoe_tender (tdr)

6.13 Indexes - Naming Conventions

Indexes should be named as follows:

<project code>_ <table alias>_ idx<number>

Example: mms_cpt_idx1 for an index on a foreign key of mms_component (cpt)

Indexes for primary and unique keys have the same name as the constraints.

6.14 Sequences - Naming Conventions

Sequences should be named as follows:

<project code>_ <table alias>_seq

Example: mms_cpt_seq for mms_component (cpt)

6.15 Triggers - Naming Conventions

Triggers should be named as follows:
<project code>_<table alias>_ <trigger type>trg

build with audit onboard

Possible trigger types:

b or a => before or after
s or r => statement or row
i, u and/or d => insert, update and/or delete

Example: mms_cpt_britrg for the trigger on mms_components (cpt) that fires before insert on each row

Code:

```
create or replace trigger <table name>_briutrg
before insert or update on <table name>
for each row
begin
  if inserting
  then
    if :new.id is null
    then
      select <table name>_seq.nextval into :new.id from dual;
    end if;
    :new.created_by := nvl(v('APP_USER'), user);
    :new.created_date := sysdate;
  end if;
  if updating
  then
    :new.modified_by := nvl(v('APP_USER'), user);
    :new.modified_date := sysdate;
  end if;
end;
```

6.16 Views - Naming Conventions

<project code>_ <logical name>_vw

singular

Example: mms_milestone_vw

6.17 Packages - Naming Conventions

project code _ logical name_pkg

6.18 Functions - Naming Conventions

project code _ logical name_fnc

6.19 Procedures - Naming Conventions

project code _ logical name_prc

6.20 Packages, Functions, Procedures – coding guidelines

Comments and breaks between procedures and functions, but no empty lines

Constants: no prefix

Global variables: g_

Local variables: l_

Exceptions: e_

Indentation: 2 spaces

Everything in lower case.

See existing PL/SQL code for more format conventions.

6.21 Quick naming conventions overview

Type	Building Blocks	Example
Project	< logical code>	mms
Table	<project code>_ < logical name>	mms_component
Table alias (comment)	< logical alias>	cpt
Primary Key	<project code>_<table alias>_pk	mms_cpt_pk
Unique Key	<project code>_<table alias>_uk <number>	mms_cpt_uk1
Foreign Key	<project code>_<table alias primary>_<table alias secondary>_fk <number>	mms_cpt_cte_fk1
Index	<project code> _<table alias>_idx<number>	mms_cpt_idx1
Sequence	<project code> _<table alias>_ seq	mms_cpt_seq
Trigger	<project code> _<table alias>_<trigger type>trg	mms_cpt_briutrg
View	<project code> _ <logical name> _ vw	mms_component_vw
Package	<project code> _ <logical name> _ pkg	mms_util_pkg
Procedure	<project code> _ <logical name> _ prc	mms_login_prc

6.22 Use lowercase all the time

Use lowercase on all levels, even for reserved words like Example: CURSOR, SELECT, … This standard should be followed within APEX, packages, functions, procedures & triggers.

6.23 Usage of PL/SQL Doc

JavaDoc is a standard used by java developers so that comments within their code can be easily viewed by external people to see what is being done in each bit of the code.

This can also be done for PL/SQL code and is called PL/SQL Doc. Just add good information about what is done in each piece of your code. Then with a wrapper, HTML pages can easily be generated so that users can quickly search through your code and the necessary documentation is being generated.

```
Example:
create or replace package customer_pkg
is
/**
* Project:         Test Project (<a href="http://hostname">Test</a>)<br/>
* Description:     Customer Data Management<br/>
* DB impact:       YES<br/>
* Commit inside:   NO<br/>
* Rollback inside: NO<br/>
* @headcom
*/

/**
* Record of customer data.
*
* @param id                customer ID
* @param name              customer name
* @param regno             registration number or SSN
* @param language  preferred language
*/
…etc
```

As you can see, even HTML tags can be set in your comment sections, and they are usable afterwards since they will be picked up by the wrapper script that will extract the comments from the code.

A known wrapper is pldoc (http://pldoc.sourceforge.net/maven-site/), but other wrappers exists on the market.

We also recommend using https://github.com/OraOpenSource/plsql-md-doc which will generate a markdown file with all the documentation based on the comments you added.

6.24 Usage of Packages to store business logic

We recommend to store all business logic and reusable components in database packages.

6.25 Code formatting

Typically we use Oracle SQL Developer to code our PL/SQL packages. SQL Developer allows you to format your code. We use lower case everywhere and indent with 2 spaces.

6.26 Auditing

If detailed auditing is necessary it's worth while to look into Flashback Data Archive.