SQL Formatting For Oracle

**SQL Clauses**

Each of the SQL clauses must be on a separate line. These include the following:

* WITH
* SELECT
* INTO
* FROM
* WHERE
* GROUP BY
* HAVING
* ORDER BY
* UPDATE
* INSERT
* DELETE

For example:

SELECT column1, column2

FROM my\_table

WHERE column3 = 53

GROUP BY column1

ORDER BY column1

**Capitalization**

**Reserved Words**

Reserved words in SQL must be in all capitals.

**Built-in Objects**

The names of objects that are built into Oracle, such as built in tables, views, functions, procedures, packages, etc. must be in upper case. For example, "SYSDATE" or "USER\_TABLES".

**User Defined Objects**

The name of user defined objects, such as tables, view, functions, stored procedures, etc. should be in all lower case. If the name contains multiple words they should be separated by an underscore. For example, "student\_list" is acceptable.

**Selecting Multiple Columns**

If you are selecting multiple columns you may list all the columns on the same as long as none of the columns contains any sort of a function or calculation, and also does not have an alias. Columns with a table name specified (or table alias) may also be listed on the same line. Like all other lines, these lines must be 80 characters or less total.

You may also list each column on a separate line. Any column that has any sort of function or calculation must be listed on a line by itself. Any column that has an alias must be listed on a line by itself. If the column list won't fit on one line, list each column on a separate line with proper indentation (see below for indentation rules). You may place the comma separating the columns either as the first or last character on the line, although you must be consistent. If you put the comma at the beginning of the line it must be followed by a single space.

The following examples are acceptable.

SELECT student\_nbr, first\_name, last\_name,

last\_name, middle\_name, street\_address1,

ISNULL(street\_address2,'-') AS street\_address2,

city, state, postal\_code

FROM students

WHERE grade = 5

SELECT student\_nbr,

first\_name,

last\_name,

MiddleName,

street\_address1,

ISNULL(s.street\_address2,'-') AS street\_address2,

city,

state,

postal\_code

FROM students

WHERE grade = 5

SELECT student\_nbr

, first\_name

, last\_name

, MiddleName

, street\_address1

, ISNULL(s.street\_address2,'-') AS street\_address2

, city

, state

, postal\_code

FROM students

WHERE grade = 5

**Complex Where Clauses**

If your **WHERE** clause contains more than one condition you may place all the condition on the same line as the **WHERE** as long as the lists fits on a single 80 column line. If your **WHERE** clause does not fit on a single line and condition should be placed on a separate line and indented properly see below for indentation rules). The following examples are acceptable.

SELECT student\_nbr, first\_name, last\_name

FROM students

WHERE grade = 5 OR grade = 6 OR grade = 7

SELECT student\_nbr, first\_name, last\_name

FROM students

WHERE grade = 5

AND first\_name = 'John'

AND last\_name = 'Smith'

AND city = 'CHICAGO'

AND state = 'IL'

AND postal\_code = '60606-0342'

**Indentation**

Each line in your statement that starts with a reserved word must use indentation to separate this reserved word from the rest of the line so the the rest of the line for line lines up in a single column. If your select clause is on multiple lines, the column names must also be lined up along this column. You may left justify or right justify the reserved word.

See details below if you query contains sub-queries or your **WHERE** clause has parenthesis and does not fit on one line.

The following examples are both acceptable.

SELECT student\_nbr,

first\_name,

last\_name

FROM students

WHERE grade = 5

AND last\_name = 'Smith'

SELECT student\_nbr,

first\_name,

last\_name

FROM students

WHERE grade = 5

AND last\_name = 'Smith'

**Aliases**

For both column names aliases use **AS** when defining an alias. Aliasing table names in a join is optional.

**Updates**

Update statements should be formatted similar to **SELECT** statements. If more than one column is updated then list each column on a separate line, similar to table names in an implicitly defined join (see above). For example:

UPDATE employees

SET salary = 85345.32,

last\_updated\_date = SYSDATE

WHERE emp\_nbr = 'THX1138'

**Inserts**

*Under construction.*

**Select Into**

**SELECT INTO** statements should be formatted just like regular **SELECT** statements. The **INTO** should be treated as another clause. For example:

SELECT student\_nbr,

first\_name,

last\_name

INTO fifth\_graders

FROM students

WHERE grade = 5

**Insert Into**

**INSERT INTO** statements should be formatted just like regular **SELECT** statements. The **INSERT INTO** should be indented in a column with the**SELECT** statement. For example:

INSERT INTO fifth\_graders

SELECT student\_nbr,

first\_name,

last\_name

FROM students

WHERE grade = 5

**Views**

Everything before the **SELECT** in a **CREATE VIEW** statement should be on a single line. The **SELECT** statement should start on the second line, and be formatted like any other **SELECT** statement. The **CREATE VIEW** should be indented in a column with the **SELECT** statement. For example:

CREATE VIEW emp\_dep\_view AS

SELECT d.last\_name, d.first\_name

FROM employees e

INNER JOIN dependents d

ON e.employee\_nbr = d.employee\_nbr

INNER JOIN dependent\_types t

ON d.dep\_type = t.dep\_type

WHERE e.employee\_nbr = '9832'

**Implicit Joins**

Implicit joins are joins that do not use the **JOIN** key word. Each table should be listed on a separate line, and the indentation and use of commas should follow the rules or a multi-line **SELECT** clause. The following examples are acceptable.

SELECT d.last\_name, d.first\_name

FROM employees e,

dependents d

WHERE e.employee\_nbr = '9832'

SELECT d.last\_name, d.first\_name

FROM employees e

, dependents d

WHERE e.employee\_nbr = '9832'

**Explicit Joins**

Explicit joins are joins that use the **JOIN** key word. The first table should be listed on the same line as the **FROM**. The **JOIN** and the table to be joined to should be listed on a separate line. If an **ON** is used it should be listed on a separate line, and formatted like a **WHERE** clause.

The use of **OUTER** in a **LEFT**, **RIGHT**, or **FULL** join is optional. You must specify **INNER JOIN** for an inner join (do not use just **JOIN**).

The following examples is acceptable.

SELECT d.last\_name, d.first\_name

FROM employees e

INNER JOIN dependents d

ON e.employee\_nbr = d.employee\_nbr

WHERE e.employee\_nbr = '9832'

SELECT d.last\_name, d.first\_name

FROM employees e

INNER JOIN dependents d

ON e.employee\_nbr = d.employee\_nbr

INNER JOIN dependent\_types t

ON d.dep\_type = t.dep\_type

WHERE e.employee\_nbr = '9832'

**Sub-queries**

*Under construction.*

**Where Clauses with Parenthesis**

*Under construction.*

**Create Table**

*Under construction.*

**Foreign Keys**

*Under construction.*

**Implicit Data Type Conversions**

Do not use implicit data type conversions. For example, the following is *not* valid:

SELECT '1' + '5' FROM dual;

When you need to convert from one type to another use an explicit conversion function, for example:

SELECT TO\_NUMBER('1') + TO\_NUMBER('5') FROM dual;

**The One-Line Exception**

If the entire statement can fit on one 80 column line then you do not have to follow any of the rules above regarding breaking statements up onto multiple lines. All other rules (e.g. capitalization) still apply. If you statement goes over 80 characters than all of the above rules apply. The following are acceptable:

SELECT \* FROM employees

SELECT \* FROM a, b WHERE a.m\_id = b.m\_id AND a.grp=2 ORDER BY a.grp, b.x