CS 328 SQL and PL/SQL Coding Standards so far

last modified: 2025-02-11

Here is the current set of CS 328 SQL, SQL*Plus, and PL/SQL Style Standards -- your CS 328 SQL, SQL*Plus, and PL/SQL code is expected to conform to these standards:

General

- Put a blank line **before** and **after** each comment, for better readability.
  - However: if the comment is the FIRST thing within a file or within a PL/SQL block, it does not have to be preceded by a blank line unless you wish.
- When a statement or command is long (more than 80 characters), continue it on the next line(s) as needed, **indenting** the continuations by **at least 3** spaces.
  - Likewise, if a statement *clause* is longer than one line, **indent** the continuation on the next line by **at least 3** spaces (so it is clear which clause it "belongs" to).
- Do **NOT** have more than one statement per line.
- Put a **blank line before** and **after** each **SELECT** statement,

  before and **after** each PL/SQL **subroutine**, and

  usually **before** and **after** each **multi-line** SQL statement,

  for better readability.
  - Also, logically group SQL*Plus statements within a script.
  - However: If the SQL statement is the FIRST statement within a PL/SQL block, it does not have to be preceded by a blank line unless you wish. For example, I would accept this:

```
begin
    select ttl_auth_lname
    into   found_auth_name
    from   title
    where  ttl_auth_lname = p_author_name;

    if found_auth_name is not null then
        ...
```

- When in doubt, follow the style of posted class examples, AND ask me.

SQL SELECT-specific

- Write the beginning of a **SELECT** statement's **FROM** clause on its OWN line.
- Write the beginning of a **SELECT** statement's **WHERE** clause on its OWN line.
- When a **SELECT** statement has N tables/relations in its **FROM** clause (OR for joins written using the

ASCII JOIN syntax), be sure you have (N-1) join conditions (unless one REALLY, TRULY wants a true Cartesian product, a rare occurrence!).

- Use mnemonic table aliases (**d** and **e** for tables dept and empl, for example, not **x** and **y** or **a** and **b** - there should be some **obvious relationship** between the alias and the table name).

- Use an **ORDER BY** clause ONLY for an outermost **SELECT** (not within any sub-select), and it shall be indented to make clear that it "belongs" to the outermost **SELECT**.

- Use a **GROUP BY** clause ONLY when one has a good reason (usually a computation that you wish done to those groups).

SQL nested SELECTS/sub-selects

- Surround each nested **select** statement with a set of parentheses **( )**.

  - (this is syntactically required in some contexts, such as after the **IN** operator. But it is apparently syntactically permitted to omit such parentheses when the nested **select** is, for example, one of the operands for **union**, **intersect**, or **minus**.)

  - (so, it is a class style standard that such nested **select** statements always be surrounded by parentheses, even when such parentheses are not required)

- **Indent** nested **select** statements by **at least 3** spaces...

- ...**EXCEPT** when the nested **select** is one of the operands for **union**, **intersect**, or **minus**. That is,

  - ...nested **select**s (sub-selects) should be indented within their outer **select**, STILL with their **from** and **where** clauses each on their own line -- for example, **both** of the following meet class style standards:

```
select  empl_last_name, salary
from    empl
where   dept_num IN
        (select dept_num
         from   dept
         where  dept_loc = 'Dallas');

select  empl_last_name, salary
from    empl
where   dept_num IN (select dept_num
                     from   dept
                     where  dept_loc = 'Dallas');
```

  - But, this is not required for nested **select**s (sub-selects) that are operands to **union**, **intersect**, or **minus**:

```
(select empl_last_name, salary "Total Compensation"
 from   empl
 where  commission is null)
union
```

```
(select empl_last_name, salary + commission
 from   empl
 where  commission is not null)
order by "Total Compensation";
```

- When using **EXISTS** or **NOT EXISTS**:

  - its sub-select argument is EXPECTED to include an appropriate **correlation condition**.

  - its sub-select argument is EXPECTED to project a literal (since these predicates only "care" if any rows exist in that sub-select's results, NOT those rows' contents, and why bother doing much work projecting those contents, then?)

PL/SQL-specific

- Precede each PL/SQL subroutine with an **opening comment block** that includes at least:

  - the subroutine's **name** for a trigger, and a **signature** for a function or procedure

  - a **purpose statement** which explicitly describes what the subroutine **expects** (for a trigger, describe **when it is fired**), and what it **does** and/or **returns**

- Within a **declare** section, **indent** local variable declarations by **at least 3** spaces, and declare at most one variable per line.

- Put any type of **BEGIN .. END**, **IF .. END IF**, **LOOP .. END LOOP**, **EXCEPTION .. END**, **WHEN**, etc., each on their **own** line,

  - and **indent** statements within them by **at least 3** spaces.

- Do NOT have more than one statement per line.

- Write a **SELECT** statement's **INTO** clause on its **OWN** line.

- Do **NOT** give a **parameter or local variable** a name that is exactly the same as a column name in any of the tables involved in that subroutine.

- Do **NOT** use **GOTO** or **EXIT WHEN** statements in CS 328.

...and I reserve the option to add to this list over the course of the semester.