

# SQL Style Guide

## Table of Contents

- [Consistency](#)
- [Reserved Words](#)
- [Variable Names](#)
- [Be Explicit](#)
  - [Aliasing](#)
  - [Joins](#)
  - [Grouping Columns](#)
- [Left Align Root Keywords](#)
- [Code Blocks](#)
- [Join Conditions](#)
- [Parentheses](#)
- [Boolean at the Beginning of Line](#)
- [Comma at the End of the Line](#)
- [Nested Queries](#)
- [About this Document](#)

## Consistency

From [Pep8](#):

---

A style guide is about consistency. Consistency with this style guide is important. Consistency within a project is more important. Consistency within one module or function is the most important.

However, know when to be inconsistent -- sometimes style guide recommendations just aren't applicable. When in doubt, use your best judgment. Look at other examples and decide what looks best. And don't hesitate to ask!

---

# Reserved Words

Always use uppercase for reserved keywords like `SELECT` , `WHERE` , or `AS` .

# Variable Names

1. Use consistent and descriptive identifiers and names.
2. Use lower case names with underscores, such as `first_name` . Do not use camelCase.
3. Functions, such as `cardinality` , `approx_distinct` , or `substr` , [are identifiers](#) and should be treated like variable names.
4. Names must begin with a letter and may not end in an underscore.
5. Only use letters, numbers, and underscores in variable names.

# Be Explicit

When choosing between explicit or implicit syntax, prefer explicit.

## Aliasing

Always include the `AS` keyword when aliasing a variable or table name, it's easier to read when explicit.

### Good

```
SELECT
    date(submission_timestamp) AS day
FROM
    telemetry.main
LIMIT
    10
```

### Bad

```
SELECT
    date(submission_timestamp) day
FROM
    telemetry.main
LIMIT
    10
```

## Joins

Always include the `JOIN` type rather than relying on the default join.

### Good

```
-- BigQuery Standard SQL Syntax
SELECT
  submission_date,
  experiment.key AS experiment_id,
  experiment.value AS experiment_branch,
  count(*) AS count
FROM
  telemetry.clients_daily
CROSS JOIN
  UNNEST(experiments.key_value) AS experiment
WHERE
  submission_date > '2019-07-01'
  AND sample_id = '10'
GROUP BY
  submission_date,
  experiment_id,
  experiment_branch
```

### Bad

```
-- BigQuery Standard SQL Syntax
SELECT
  submission_date,
  experiment.key AS experiment_id,
  experiment.value AS experiment_branch,
  count(*) AS count
FROM
  telemetry.clients_daily,
  UNNEST(experiments.key_value) AS experiment -- Implicit JOIN
WHERE
  submission_date > '2019-07-01'
  AND sample_id = '10'
GROUP BY
  1, 2, 3 -- Implicit grouping column names
```

## Grouping Columns

In the previous example, implicit grouping columns were discouraged, but there are cases where it makes sense.

In some SQL flavors (such as [Presto](#)) grouping elements must refer to the expression before any aliasing is done. If you are grouping by a complex expression it may be desirable to use

implicit grouping columns rather than repeating the expression.

## Good

```
-- BigQuery SQL Syntax
SELECT
  submission_date,
  normalized_channel IN ('nightly', 'aurora', 'beta') AS is_prerelease,
  count(*) AS count
FROM
  telemetry.clients_daily
WHERE
  submission_date > '2019-07-01'
GROUP BY
  submission_date,
  is_prerelease -- Grouping by aliases is supported in BigQuery
```

## Good

```
-- Presto SQL Syntax
SELECT
  submission_date,
  normalized_channel IN ('nightly', 'aurora', 'beta') AS is_prerelease,
  count(*) AS count
FROM
  telemetry.clients_daily
WHERE
  submission_date > '20190701'
GROUP BY
  1, 2 -- Implicit grouping avoids repeating expressions
```

## Bad

```
-- Presto SQL Syntax
SELECT
  submission_date,
  normalized_channel IN ('nightly', 'aurora', 'beta') AS is_prerelease,
  count(*) AS count
FROM
  telemetry.clients_daily
WHERE
  submission_date > '20190701'
GROUP BY
  submission_date,
  normalized_channel IN ('nightly', 'aurora', 'beta')
```

# Left Align Root Keywords

Root keywords should all start on the same character boundary. This is counter to the common "rivers" pattern [described here](#).

**Good:**

```
SELECT
  client_id,
  submission_date
FROM
  main_summary
WHERE
  sample_id = '42'
  AND submission_date > '20180101'
LIMIT
  10
```

**Bad:**

```
SELECT client_id,
       submission_date
FROM   main_summary
WHERE  sample_id = '42'
       AND submission_date > '20180101'
```

## Code Blocks

Root keywords should be on their own line. For example:

**Good:**

```
SELECT
  client_id,
  submission_date
FROM
  main_summary
WHERE
  submission_date > '20180101'
  AND sample_id = '42'
LIMIT
  10
```

It's acceptable to include an argument on the same line as the root keyword, if there is exactly one argument.

## Acceptable:

```
SELECT
  client_id,
  submission_date
FROM main_summary
WHERE
  submission_date > '20180101'
  AND sample_id = '42'
LIMIT 10
```

Do not include multiple arguments on one line.

## Bad:

```
SELECT client_id, submission_date
FROM main_summary
WHERE
  submission_date > '20180101'
  AND sample_id = '42'
LIMIT 10
```

## Bad

```
SELECT
  client_id,
  submission_date
FROM main_summary
WHERE submission_date > '20180101'
  AND sample_id = '42'
LIMIT 10
```

## Join Conditions

The `ON` and `USING` keywords should start on a new line indented one level more than the join keyword and be followed by the join conditions starting on the same line. For example:

## Good:

```
...
FROM
  telemetry_stable.main_v4
LEFT JOIN
  static.normalized_os_name
  ON main_v4.environment.system.os.name = normalized_os_name.os_name
```

## Bad:

```
...
FROM
    telemetry_stable.main_v4
LEFT JOIN
    static.normalized_os_name ON main_v4.environment.system.os.name =
normalized_os_name.os_name
```

## Bad:

```
...
FROM
    telemetry_stable.main_v4
LEFT JOIN
    static.normalized_os_name
ON
    main_v4.environment.system.os.name = normalized_os_name.os_name
```

# Parentheses

If parentheses span multiple lines:

1. The opening parenthesis should terminate the line.
2. The closing parenthesis should be lined up under the first character of the line that starts the multi-line construct.
3. The contents of the parentheses should be indented one level.

For example:

## Good

```
WITH sample AS (
    SELECT
        client_id,
    FROM
        main_summary
    WHERE
        sample_id = '42'
)
```

**Bad** (Terminating parenthesis on shared line)

```
WITH sample AS (  
  SELECT  
    client_id,  
  FROM  
    main_summary  
  WHERE  
    sample_id = '42')
```

**Bad** (No indent)

```
WITH sample AS (  
  SELECT  
    client_id,  
  FROM  
    main_summary  
  WHERE  
    sample_id = '42'  
)
```

## Boolean at the Beginning of Line

**AND** and **OR** should always be at the beginning of the line. For example:

**Good**

```
...  
WHERE  
  submission_date > 20180101  
  AND sample_id = '42'
```

**Bad**

```
...  
WHERE  
  submission_date > 20180101 AND  
  sample_id = '42'
```

## Comma at the End of the Line

Commas between **SELECT** elements should always be at the end of the line. For example:

**Good**



```
SELECT
    client_id,
    os_name,
    submission_date
FROM
    main_summary
WHERE
    submission_date > '20180101'
LIMIT
    10
```

## Bad

```
SELECT
    client_id
    , os_name
    , submission_date
FROM
    main_summary
WHERE
    submission_date > '20180101'
LIMIT
    10
```

## Nested Queries

Do not use nested queries. Instead, use common table expressions to improve readability.

### Good:

```
WITH sample AS (
    SELECT
        client_id,
        submission_date
    FROM
        main_summary
    WHERE
        sample_id = '42'
)

SELECT *
FROM sample
LIMIT 10
```

### Bad:

```
SELECT *  
FROM (  
  SELECT  
    client_id,  
    submission_date  
  FROM  
    main_summary  
  WHERE  
    sample_id = '42'  
)  
LIMIT 10
```

## About this Document

This document was heavily influenced by <https://www.sqlstyle.guide/>

Changes to the style guide should be reviewed by at least one member of both the Data Engineering team and the Data Science team.

---

Found a bug? [Edit this page on GitHub.](#)