SQL

25th Sep 2020 | Updated: 6th Sep 2023 | 7 minutes read

SQL JOIN Cheat Sheet

LearnSQL.com Team

Cheat Sheet JOIN

Tabl
Con

Ready to master SQL JOINs? Get your downloadable cheat sheet now! Your shortcut to becoming an SQL JOINs expert is one click away.

Dive deep into the world of SQL `JOIN` s with our detailed **SQL JOIN Cheat Sheet**, a must-have resource for data enthusiasts at every level. Whether you are just starting out or looking to sharpen your skills, this guide is tailored to provide you with the quick reference needed to use SQL `JOINs` efficiently.

In this guide, we show the syntax of each `JOIN` type, coupled with practical examples. From the basics of `INNER JOIN` to the complexities of `FULL OUTER JOIN`

and the seldom-used `NATURAL JOIN`, we have got it all covered for you. Beyond the standard `JOIN` operations, we discuss advanced constructions, such as joining the same table twice (self-join) and using non-equality conditions (non-equi self join). Moreover, we show queries involving multiple `JOIN`s and multiple conditions, empowering you to tackle real-world data scenarios with finesse.

Available in both PDF and PNG formats for your convenience, this cheat sheet is designed to be a readily accessible resource, ideal for printing out and keeping nearby for quick reference.

Download Options:

- Download A4 PDF – The standard size for most printers, offering a balance between space and readability.
- Download Letter PDF – The North American standard paper size, ideal for printing on letter paper readily available in the US and Canada.
- Download A3 PDF – A larger format, perfect for those who prefer a spacious layout with larger text and diagrams.
- Download Ledger PDF – A tabloid size offering ample space for detailed diagrams and examples.
- Download Mobile PDF – A mobile-friendly PDF to browse easily on your mobile device.

For those who prefer working with digital documents, Soda PDF offers excellent tools for viewing and editing SQL cheat sheets and other resources.

> LearnSQL.com lets you learn SQL by writing SQL code on your own. You build your SQL skills gradually. Each new concept is reinforced by an interactive exercise. By actually writing SQL code, you build your confidence.

If you prefer a more visual approach or need a digital-friendly version for various applications, we also offer the cheat sheet as a high-resolution PNG image. To download, right-click (for desktop users) or long tap (for mobile users) on the image.

SQL JOINs Cheat Sheet

LearnSQL.com

JOINING TABLES

`JOIN` combines data from two tables.

TOY

toy_id	toy_name	cat_id
1	ball	3
2	spring	NULL
3	mouse	1
4	mouse	4
5	ball	1

CAT

cat_id	cat_name
1	Kitty
2	Hugo
3	Sam
4	Misty

`JOIN` typically combines rows with equal values for the specified columns. **Usually**, one table contains a **primary key**, which is a column or columns that uniquely identify rows in the table (the `cat_id` column in the `cat` table).
The other table has a column or columns that **refer to the primary key columns** in the first table (the `cat_id` column in the `toy` table). Such columns are **foreign keys**. The `JOIN` condition is the equality between the primary key columns in one table and columns referring to them in the other table.

JOIN

`JOIN` returns all rows that match the `ON` condition. `JOIN` is also called `INNER JOIN`.

```
SELECT *
FROM toy
JOIN cat
  ON toy.cat_id = cat.cat_id;
```

toy_id	toy_name	cat_id	cat_id	cat_name
5	ball	1	1	Kitty
3	mouse	1	1	Kitty
1	ball	3	3	Sam
4	mouse	4	4	Misty

There is also another, older syntax, but it **isn't recommended**.
List joined tables in the `FROM` clause, and place the conditions in the `WHERE` clause.

```
SELECT *
FROM toy, cat
WHERE toy.cat_id = cat.cat_id;
```

JOIN CONDITIONS

The `JOIN` condition doesn't have to be an equality – it can be any condition you want. `JOIN` doesn't interpret the `JOIN` condition, it only checks if the rows satisfy the given condition.

To refer to a column in the `JOIN` query, you have to use the full column name: first the table name, then a dot (`.`) and the column name:

```
  ON cat.cat_id = toy.cat_id
```

You can omit the table name and use just the column name if the name of the column is unique within all columns in the joined tables.

NATURAL JOIN

If the tables have columns with **the same name**, you can use `NATURAL JOIN` instead of `JOIN`.

```
SELECT *
FROM toy
NATURAL JOIN cat;
```

cat_id	toy_id	toy_name	cat_name
1	5	ball	Kitty
1	3	mouse	Kitty
3	1	ball	Sam
4	4	mouse	Misty

The common column appears only once in the result table.
Note: `NATURAL JOIN` is rarely used in real life.

LEFT JOIN

`LEFT JOIN` returns all rows from the **left table** with matching rows from the right table. Rows without a match are filled with `NULL`s. `LEFT JOIN` is also called `LEFT OUTER JOIN`.

```
SELECT *
FROM toy
LEFT JOIN cat
  ON toy.cat_id = cat.cat_id;
```

toy_id	toy_name	cat_id	cat_id	cat_name
5	ball	1	1	Kitty
3	mouse	1	1	Kitty
1	ball	3	3	Sam
4	mouse	4	4	Misty
2	spring	NULL	NULL	NULL

whole left table

RIGHT JOIN

`RIGHT JOIN` returns all rows from the **right table** with matching rows from the left table. Rows without a match are filled with `NULL`s. `RIGHT JOIN` is also called `RIGHT OUTER JOIN`.

```
SELECT *
FROM toy
RIGHT JOIN cat
  ON toy.cat_id = cat.cat_id;
```

toy_id	toy_name	cat_id	cat_id	cat_name
5	ball	1	1	Kitty
3	mouse	1	1	Kitty
NULL	NULL	NULL	2	Hugo
1	ball	3	3	Sam
4	mouse	4	4	Misty

whole right table

FULL JOIN

`FULL JOIN` returns all rows from the **left table** and all rows from the **right table**. It fills the non-matching rows with `NULL`s. `FULL JOIN` is also called `FULL OUTER JOIN`.

```
SELECT *
FROM toy
FULL JOIN cat
  ON toy.cat_id = cat.cat_id;
```

toy_id	toy_name	cat_id	cat_id	cat_name
5	ball	1	1	Kitty
3	mouse	1	1	Kitty
NULL	NULL	NULL	2	Hugo
1	ball	3	3	Sam
4	mouse	4	4	Misty
2	spring	NULL	NULL	NULL

whole left table | whole right table

CROSS JOIN

`CROSS JOIN` returns **all possible combinations** of rows from the left and right tables.

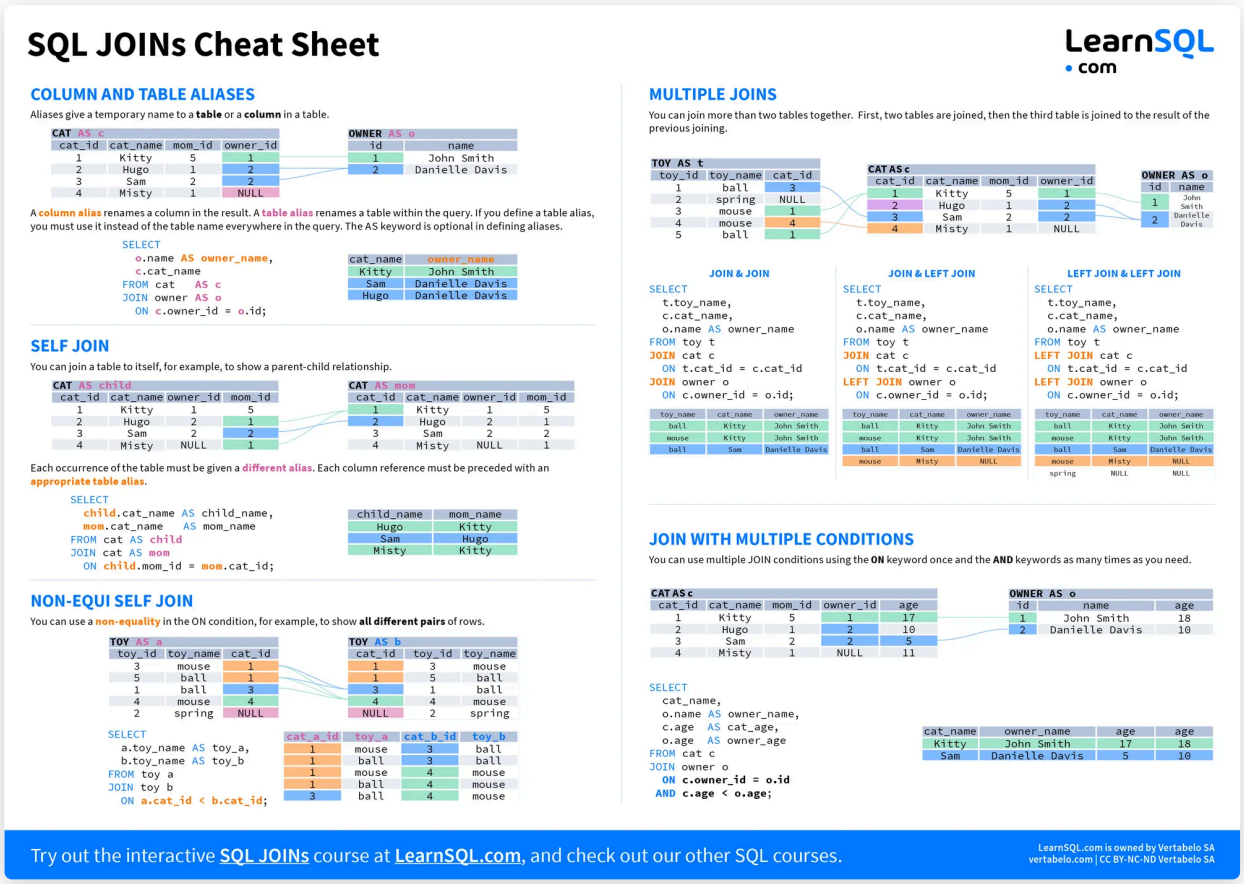
```
SELECT *
FROM toy
CROSS JOIN cat;
```

Other syntax:

```
SELECT *
FROM toy, cat;
```

toy_id	toy_name	cat_id	cat_id	cat_name
1	ball	3	1	Kitty
2	spring	NULL	1	Kitty
3	mouse	1	1	Kitty
4	mouse	4	1	Kitty
5	ball	1	1	Kitty
1	ball	3	2	Hugo
2	spring	NULL	2	Hugo
3	mouse	1	2	Hugo
4	mouse	4	2	Hugo
5	ball	1	2	Hugo
1	ball	3	3	Sam
...	...	...	...	...

Try out the interactive **SQL JOINs** course at **LearnSQL.com**, and check out our other SQL courses.



JOINING TABLES

`JOIN` combines data from two tables.

TOY

toy_id	toy_name	cat_id
1	ball	3
2	spring	NULL
3	mouse	1
4	mouse	4
5	ball	1

CAT

cat_id	cat_name
1	Kitty
2	Hugo
3	Sam
4	Misty

`JOIN` typically combines rows with equal values for the specified columns. **Usually**, one table contains a **primary key**, which is a column or columns that uniquely identify rows in the table (the `cat_id` column in the `cat` table). The

other table has a column or columns that **refer to the primary key columns** in the first table (the `cat_id` column in the `toy` table). Such columns are **foreign keys**. The `JOIN` condition is the equality between the primary key columns in one table and columns referring to them in the other table.

> Do you want to practice SQL JOINs? Check out our SQL JOINs course!

JOIN

`JOIN` returns all rows that match the `ON` condition. `JOIN` is also called `INNER JOIN`

```
SELECT *
FROM toy
JOIN cat
  ON toy.cat_id = cat.cat_id;
```

toy_id	toy_name	cat_id	cat_id	cat_name
5	ball	1	1	Kitty
3	mouse	1	1	Kitty
1	ball	3	3	Sam
4	mouse	4	4	Misty

There is also another, older syntax, but it **isn't recommended**.
List joined tables in the `FROM` clause, and place the conditions in the `WHERE` clause.

```
SELECT *
FROM toy, cat
WHERE toy.cat_id = cat.cat_id;
```

JOIN CONDITIONS

The `JOIN` condition doesn't have to be an equality – it can be any condition you want. `JOIN` doesn't interpret the `JOIN` condition, it only checks if the rows satisfy the given condition.

To refer to a column in the `JOIN` query, you have to use the full column name: first the table name, then a dot (`.`) and the column name:

```
  ON cat.cat_id = toy.cat_id
```

You can omit the table name and use just the column name if the name of the column is unique within all columns in the joined tables.

NATURAL JOIN

If the tables have columns with the same name, you can use `NATURAL JOIN` instead of `JOIN`.

cat_id	toy_id	toy_name	cat_name
1	5	ball	Kitty
1	3	mouse	Kitty
3	1	ball	Sam
4	4	mouse	Misty

```
SELECT *
FROM toy
NATURAL JOIN cat;
```

The common column appears only once in the result table.
Note: `NATURAL JOIN` is rarely used in real life.

LEFT JOIN

`LEFT JOIN` returns all rows from the **left table** with matching rows from the right table. Rows without a match are filled with `NULL` s. `LEFT JOIN` is also called `LEFT OUTER JOIN`.

```
SELECT *
FROM toy
LEFT JOIN cat
  ON toy.cat_id = cat.cat_id;
```

toy_id	toy_name	cat_id	cat_id	cat_name
5	ball	1	1	Kitty
3	mouse	1	1	Kitty
1	ball	3	3	Sam
4	mouse	4	4	Misty
2	spring	NULL	NULL	NULL

whole left table

RIGHT JOIN

`RIGHT JOIN` returns all rows from the **right table** with matching rows from the left table. Rows without a match are filled with `NULL` s. `RIGHT JOIN` is also called `RIGHT OUTER JOIN` .

```
SELECT *
FROM toy
RIGHT JOIN cat
  ON toy.cat_id = cat.cat_id;
```

toy_id	toy_name	cat_id	cat_id	cat_name
5	ball	1	1	Kitty
3	mouse	1	1	Kitty
NULL	NULL	NULL	2	Hugo
1	ball	3	3	Sam
4	mouse	4	4	Misty
			whole right table	

FULL JOIN

`FULL JOIN` returns all rows from the **left table** and all rows from the **right table**. It fills the non-matching rows with `NULL` s. `FULL JOIN` is also called `FULL OUTER JOIN` .

```
SELECT *
FROM toy
FULL JOIN cat
  ON toy.cat_id = cat.cat_id;
```

toy_id	toy_name	cat_id	cat_id	cat_name
5	ball	1	1	Kitty
3	mouse	1	1	Kitty
NULL	NULL	NULL	2	Hugo
1	ball	3	3	Sam
4	mouse	4	4	Misty
2	spring	NULL	NULL	NULL
whole left table			whole right table	

CROSS JOIN

`CROSS JOIN` returns **all possible combinations** of rows from the left and right tables.

```
SELECT *
FROM toy
CROSS JOIN cat;
```

Other syntax:

```
SELECT *
FROM toy, cat;
```

toy_id	toy_name	cat_id	cat_id	cat_name
1	ball	3	1	Kitty
2	spring	NULL	1	Kitty
3	mouse	1	1	Kitty
4	mouse	4	1	Kitty
5	ball	1	1	Kitty
1	ball	3	2	Hugo
2	spring	NULL	2	Hugo
3	mouse	1	2	Hugo
4	mouse	4	2	Hugo
5	ball	1	2	Hugo
1	ball	3	3	Sam
...	...	...	...	...

COLUMN AND TABLE ALIASES

Aliases give a temporary name to a **table** or a **column** in a table.

CAT AS c

cat_id	cat_name	mom_id	owner_id
1	Kitty	5	1
2	Hugo	1	2
3	Sam	2	2
4	Misty	1	NULL

OWNER AS o

id	name
1	John Smith
2	Danielle Davis

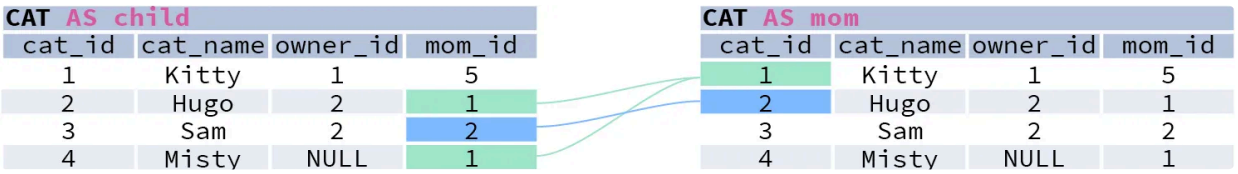
A **column alias** renames a column in the result. A **table alias** renames a table within the query. If you define a table alias, you must use it instead of the table name everywhere in the query. The AS keyword is optional in defining aliases.

```
SELECT
  o.name AS owner_name,
  c.cat_name
FROM cat   AS c
JOIN owner AS o
  ON c.owner_id = o.id;
```

cat_name	owner_name
Kitty	John Smith
Sam	Danielle Davis
Hugo	Danielle Davis

SELF-JOIN

You can join a table to itself, for example, to show a parent-child relationship.



CAT AS child

cat_id	cat_name	owner_id	mom_id
1	Kitty	1	5
2	Hugo	2	1
3	Sam	2	2
4	Misty	NULL	1

CAT AS mom

cat_id	cat_name	owner_id	mom_id
1	Kitty	1	5
2	Hugo	2	1
3	Sam	2	2
4	Misty	NULL	1

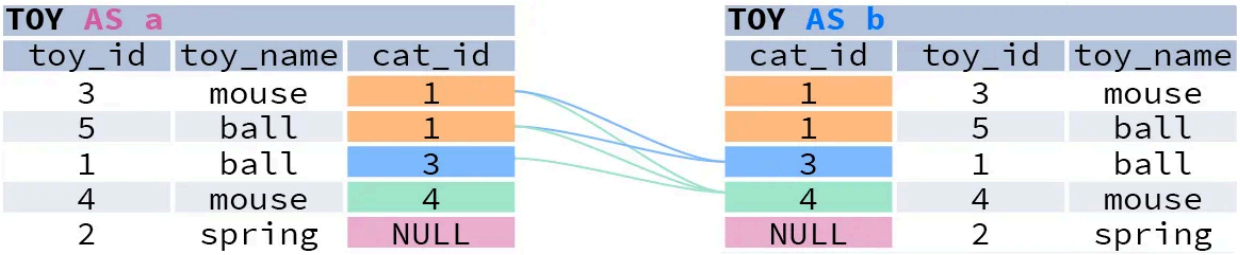
Each occurrence of the table must be given a **different alias**. Each column reference must be preceded with an **appropriate table alias**.

```
SELECT
  child.cat_name AS child_name,
  mom.cat_name   AS mom_name
FROM cat AS child
JOIN cat AS mom
  ON child.mom_id = mom.cat_id;
```

child_name	mom_name
Hugo	Kitty
Sam	Hugo
Misty	Kitty

NON-EQUI SELF-JOIN

You can use a **non-equality** in the `ON` condition, for example, to show **all different pairs** of rows.



TOY AS a

toy_id	toy_name	cat_id
3	mouse	1
5	ball	1
1	ball	3
4	mouse	4
2	spring	NULL

TOY AS b

cat_id	toy_id	toy_name
1	3	mouse
1	5	ball
3	1	ball
4	4	mouse
NULL	2	spring

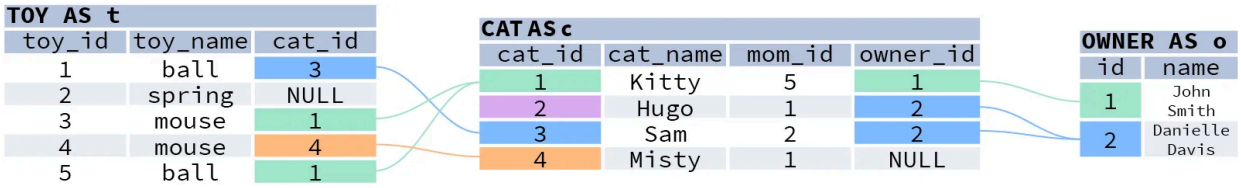
```
SELECT
  a.toy_name AS toy_a,
  b.toy_name AS toy_b
FROM toy a
JOIN toy b
  ON a.cat_id < b.cat_id;
```

cat_a_id	toy_a	cat_b_id	toy_b
1	mouse	3	ball
1	ball	3	ball
1	mouse	4	mouse
1	ball	4	mouse
3	ball	4	mouse

Practice SQL JOINs with our interactive SQL JOINs course.

MULTIPLE JOINS

You can join more than two tables together. First, two tables are joined, then the third table is joined to the result of the previous joining.



JOIN & JOIN

```
SELECT
  t.toy_name,
  c.cat_name,
  o.name AS owner_name
FROM toy t
JOIN cat c
  ON t.cat_id = c.cat_id
JOIN owner o
  ON c.owner_id = o.id;
```

toy_name	cat_name	owner_name
ball	Kitty	John Smith
mouse	Kitty	John Smith
ball	Sam	Danielle Davis

JOIN & LEFT JOIN

```
SELECT
  t.toy_name,
  c.cat_name,
  o.name AS owner_name
FROM toy t
JOIN cat c
  ON t.cat_id = c.cat_id
LEFT JOIN owner o
  ON c.owner_id = o.id;
```

toy_name	cat_name	owner_name
ball	Kitty	John Smith
mouse	Kitty	John Smith
ball	Sam	Danielle Davis
mouse	Misty	NULL

LEFT JOIN & LEFT JOIN

```
SELECT
  t.toy_name,
  c.cat_name,
  o.name AS owner_name
FROM toy t
LEFT JOIN cat c
  ON t.cat_id = c.cat_id
LEFT JOIN owner o
  ON c.owner_id = o.id;
```

toy_name	cat_name	owner_name
ball	Kitty	John Smith
mouse	Kitty	John Smith
ball	Sam	Danielle Davis
mouse	Misty	NULL
spring	NULL	NULL

JOIN WITH MULTIPLE CONDITIONS

You can use multiple `JOIN` conditions using the `ON` keyword once and the `AND` keywords as many times as you need.

CAT AS c

cat_id	cat_name	mom_id	owner_id	age
1	Kitty	5	1	17
2	Hugo	1	2	10
3	Sam	2	2	5
4	Misty	1	NULL	11

OWNER AS o

id	name	age
1	John Smith	18
2	Danielle Davis	10

```
SELECT
  cat_name,
  o.name AS owner_name,
  c.age  AS cat_age,
  o.age  AS owner_age
FROM cat c
JOIN owner o
  ON c.owner_id = o.id
 AND c.age < o.age;
```

cat_name	owner_name	age	age
Kitty	John Smith	17	18
Sam	Danielle Davis	5	10

Try out the interactive SQL JOINs course at LearnSQL.com, and check out our other SQL courses.

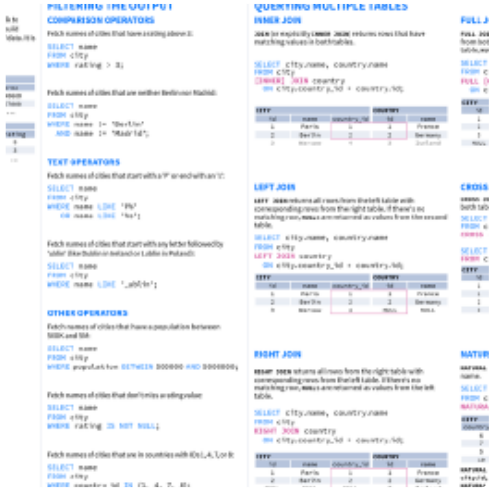
Tags: Cheat Sheet JOIN

You may also like

RIGHT JOIN in SQL: A Beginner's Tutorial

Explore RIGHT JOIN in SQL in this beginner's guide. Learn to merge tables and enhance database queries efficiently.

Read more ›



SQL Basics Cheat Sheet

SQL made simple: Download this beginner-friendly cheat sheet in A4, Letter, or mobile format. All the basics, syntax, and examples in one place.

Read more ›

Subscribe to our newsletter

Join our monthly newsletter to be notified about the latest posts.

Email address

Subscribe

Quick links

Courses
Pricing
For Students
Affiliate Program
Blog
Cookbook
LearnPython.com
Vertabelo.com

Assistance

Need assistance? Drop us a line at **contact@learnsql.com**

Write to us

Follow us