Joining Data in SQL: Takeaways ⬈

Syntax

- Joining tables using an INNER JOIN:

```
SELECT [column_names] FROM [table_name_one]

INNER JOIN [table_name_two] ON [join_constraint];
```

- Joining tables using a LEFT JOIN:

```
SELECT * FROM facts

LEFT JOIN cities ON cities.facts_id = facts.id;
```

- Joining tables using a RIGHT JOIN:

```
SELECT f.name country, c.name city

FROM cities c

RIGHT JOIN facts f ON f.id = c.facts_id

LIMIT 5;
```

- Joining tables using a FULL OUTER JOIN:

```
SELECT f.name country, c.name city

FROM cities c

FULL OUTER JOIN facts f ON f.id = c.facts_id

LIMIT 5;
```

- Sorting a column without specifying a column name:

```
SELECT name, migration_rate FROM FACTS

ORDER BY 2 desc; # 2 refers to migration_rate column
```

- Using a join within a subquery:

```
SELECT c.name capital_city, f.name country

FROM facts f

INNER JOIN (

      SELECT * FROM cities

      WHERE capital = 1

    ) c ON c.facts_id = f.id

LIMIT 10;
```

Concepts

- We use joins to combine multiple tables within a query.
- A schema diagram shows the tables in the database, the columns within the them, and how they are connected.
- The `ON` statement tells the SQL engine what columns to use to join the tables.
- Joins come after the `FROM` clause.
- An inner join is the most common way to join data using SQL. An inner join includes only rows that have a match as specified by the `ON` clause.
- A left join includes all rows from an inner join, plus any rows from the first table that don't have a match in the second table.
- A right join includes all rows from the second table that don't have a match in the first table, plus any rows from an inner join.
- A full outer join includes all rows from both joined tables.
- SQLite doesn't support full outer joins or right joins.

Resources

- [SQL Joins](#)
- [Quora – Difference in Joins](#)