SQL

* [SELECT](https://www.codecademy.com/resources/docs/sql/commands/select?page_ref=catalog) is the clause we use every time we want to query information from a database.
* [AS](https://www.codecademy.com/resources/docs/sql/commands/as?page_ref=catalog) renames a column or table.
* [DISTINCT](https://www.codecademy.com/resources/docs/sql/commands/select-distinct?page_ref=catalog) return unique values.
* [WHERE](https://www.codecademy.com/resources/docs/sql/commands/where?page_ref=catalog) is a popular command that lets you filter the results of the query based on conditions that you specify.
* [LIKE](https://www.codecademy.com/resources/docs/sql/operators/like?page_ref=catalog) and [BETWEEN](https://www.codecademy.com/resources/docs/sql/operators/between?page_ref=catalog) are special operators.
* [AND](https://www.codecademy.com/resources/docs/sql/operators/and?page_ref=catalog) and [OR](https://www.codecademy.com/resources/docs/sql/operators/or?page_ref=catalog) combines multiple conditions.
* [ORDER BY](https://www.codecademy.com/resources/docs/sql/commands/order-by?page_ref=catalog) sorts the result.
* [LIMIT](https://www.codecademy.com/resources/docs/sql/commands/limit?page_ref=catalog) specifies the maximum number of rows that the query will return.
* [CASE](https://www.codecademy.com/resources/docs/sql/commands/case?page_ref=catalog) creates different outputs.

You just learned how to use aggregate functions to perform calculations on your data. What can we generalize so far?

* [COUNT()](https://www.codecademy.com/resources/docs/sql/aggregate-functions/count?page_ref=catalog): count the number of rows
* [SUM()](https://www.codecademy.com/resources/docs/sql/aggregate-functions/sum?page_ref=catalog): the sum of the values in a column
* [MAX()](https://www.codecademy.com/resources/docs/sql/aggregate-functions/max?page_ref=catalog)/[MIN()](https://www.codecademy.com/resources/docs/sql/aggregate-functions/min?page_ref=catalog): the largest/smallest value
* [AVG()](https://www.codecademy.com/resources/docs/sql/aggregate-functions/avg?page_ref=catalog): the average of the values in a column
* [ROUND()](https://www.codecademy.com/resources/docs/sql/commands/round?page_ref=catalog): round the values in the column

*Aggregate functions* combine multiple rows together to form a single value of more meaningful information.

* [GROUP BY](https://www.codecademy.com/resources/docs/sql/commands/group-by?page_ref=catalog) is a clause used with aggregate functions to combine data from one or more columns.
* [HAVING](https://www.codecademy.com/resources/docs/sql/commands/having?page_ref=catalog) limit the results of a query based on an aggregate property.

CHEATSHEET

<https://www.codecademy.com/learn/learn-sql/modules/learn-sql-aggregate-functions/cheatsheet>

Let’s summarize what we’ve learned so far:

* [JOIN](https://www.codecademy.com/resources/docs/sql/commands/inner-join?page_ref=catalog) will combine rows from different tables if the join condition is true.
* [LEFT JOIN](https://www.codecademy.com/resources/docs/sql/commands/left-join?page_ref=catalog) will return every row in the *left* table, and if the join condition is not met, NULL values are used to fill in the columns from the *right* table.
* [*Primary key*](https://www.codecademy.com/resources/docs/sql/primary-keys?page_ref=catalog) is a column that serves a unique identifier for the rows in the table.
* [*Foreign key*](https://www.codecademy.com/resources/docs/sql/primary-keys?page_ref=catalog) is a column that contains the primary key to another table.
* [CROSS JOIN](https://www.codecademy.com/resources/docs/sql/joins?page_ref=catalog) lets us combine all rows of one table with all rows of another table.
* [UNION](https://www.codecademy.com/resources/docs/sql/commands/union?page_ref=catalog) stacks one dataset on top of another.
* [WITH](https://www.codecademy.com/resources/docs/sql/commands/with?page_ref=catalog) allows us to define one or more temporary tables that can be used in the final query.