

Querying basics

SQL Reference

SELECT and FROM

A database keeps data in tables . A query reads data with **SELECT**. **SELECT *** returns every column; **FROM** names the table.

```
CREATE TABLE student (id INTEGER, name TEXT, score INTEGER);
INSERT INTO student VALUES (1, 'Mei', 88), (2, 'Sam', 71);
SELECT * FROM student;
```

table: student

id	name	score
1	'Mei'	88
2	'Sam'	71
3	'Ana'	95

a column
(one field)

a row
(one record)

01-table-anatomy

A table stores records: each row is one record, each column is one field

- Each row is one record . SQL keywords are written in UPPERCASE by habit.
- A statement ends with a semicolon ;.

Choosing columns

List the columns you want, separated by commas. Use **AS** to rename a column in the result (an alias).

```
CREATE TABLE student (id INTEGER, name TEXT, score INTEGER);
INSERT INTO student VALUES (1, 'Mei', 88), (2, 'Sam', 71);
SELECT name, score AS mark FROM student;
```

A selected column can also be a calculation — pair it with **AS** to name the new column:

```
CREATE TABLE student (id INTEGER, name TEXT, score INTEGER);
INSERT INTO student VALUES (1, 'Mei', 88), (2, 'Sam', 71);
SELECT name, score + 5 AS bonus FROM student;
```

- **SELECT DISTINCT col** removes duplicate values from the result.

Filtering with WHERE

WHERE keeps only the rows that match a condition . Compare with **=**, **<>** (not equal), **<**, **>**, **<=**, **>=**. Put text in single quotes.

```
CREATE TABLE student (id INTEGER, name TEXT, score INTEGER);
INSERT INTO student VALUES (1, 'Mei', 88), (2, 'Sam', 71), (3, 'Ana',
↪ 95);
SELECT name, score FROM student WHERE score >= 80;
```

- SQL runs the parts in this order: **FROM** → **WHERE** → **SELECT**.

Common mistakes

- Put text values in single quotes: **WHERE name = 'Ann'**; column names have no quotes.
- **SELECT *** returns every column — name only the columns you need.
- **WHERE** filters rows; it comes after **FROM**.