

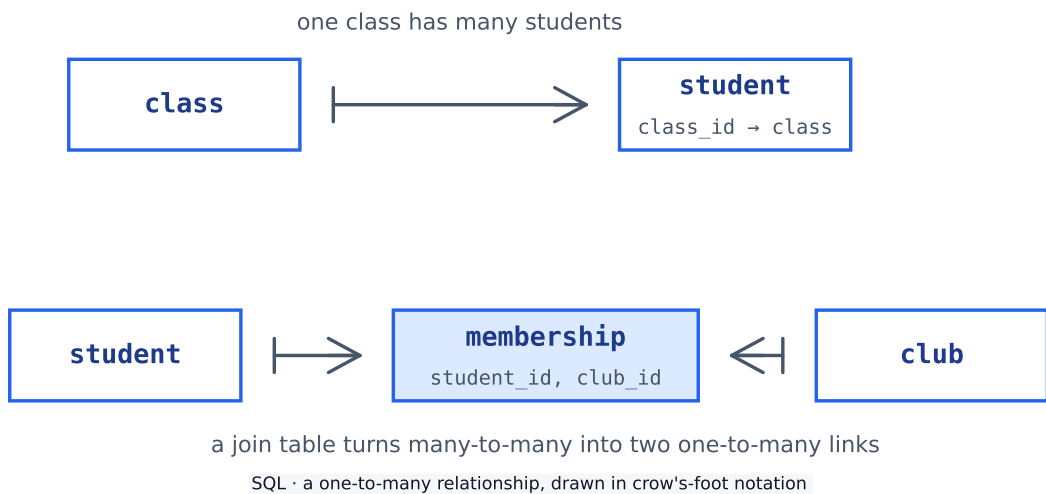
Database design

SQL Reference

Relationships & ER diagrams

Tables connect through relationships . One-to-many is the most common: one class has many students. Many-to-many needs a join table in the middle. An entity-relationship diagram (ER diagram) draws each entity as a box and each relationship as a line.

Relationship	Example
one-to-one	a person and their passport
one-to-many	a class and its students
many-to-many	students and clubs



The “many” side carries the foreign key; a many-to-many link needs a join table

The join table holds one row per link. Here Mei is in two clubs, and Chess has two members:

```
CREATE TABLE student (id INTEGER PRIMARY KEY, name TEXT);
CREATE TABLE club (id INTEGER PRIMARY KEY, name TEXT);
CREATE TABLE membership (student_id INTEGER, club_id INTEGER);
INSERT INTO student VALUES (1, 'Mei'), (2, 'Sam');
INSERT INTO club VALUES (1, 'Chess'), (2, 'Art');
INSERT INTO membership VALUES (1, 1), (1, 2), (2, 1);
SELECT s.name, c.name AS club
FROM membership m
INNER JOIN student s ON m.student_id = s.id
INNER JOIN club c ON m.club_id = c.id;
```

Normalisation

Normalisation organises tables to avoid redundancy (the same data repeated) and the update mistakes it causes. The first three normal forms :

- **1NF**: every cell holds one atomic value — no lists inside a cell.
- **2NF**: no column depends on only part of a composite key .
- **3NF**: no column depends on another non-key column.

Unnormalised (bad)	Normalised (better)
student(name, club1, club2)	student(name) + membership(student, club)

Common mistakes

- Split repeating groups into their own table (normalisation) instead of many similar columns.
- Each table should describe ONE kind of thing.
- Link tables with a foreign key that points to another table’s primary key.