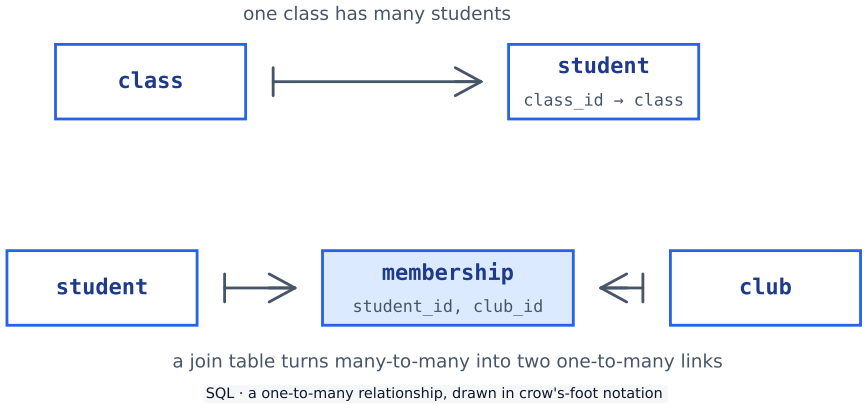
数据库设计

SQL Reference

Relationships & ER diagrams

表通过关系 (relationship) 相互连接。一对多 (one-to-many) 最常见: 一个班有许多学生。多对多 (many-to-many) 需要中间有一张连接表 (join table)。实体关系图 (entity-relationship diagram,ER 图) 把每个实体 (entity) 画成一个方框, 把每个关系画成一条线。

关系	例子
一对一	一个人和他的护照
一对多	一个班和它的学生
多对多	学生和社团



“ 多 ”的一侧携带外键; 多对多关系需要一张连接表

exercises.lol exercises.com.cn

连接表每行对应一个链接，其各列均为外键。其主键是**配对** (`student_id, club_id`)，属于复合键，因此同一学生不能重复加入同一社团。两列均为 `NOT NULL`，故每条成员关系指定一名学生和一处社团。此处梅参加了两个社团，国际象棋社有两名成员：

```
CREATE TABLE student (id INTEGER PRIMARY KEY, name TEXT);
CREATE TABLE club (id INTEGER PRIMARY KEY, name TEXT);
CREATE TABLE membership (
  student_id INTEGER NOT NULL REFERENCES student(id),
  club_id INTEGER NOT NULL REFERENCES club(id),
  PRIMARY KEY (student_id, club_id)
);
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

范式化 (normalisation) 把表组织好, 以避免冗余 (redundancy)(同样的数据重复) 以及它带来的更新错误。前三个范式 (normal form):

- **1NF**: 每个单元格只放一个原子 (atomic) 值 — 单元格里不放列表。
- **2NF**: 没有列只依赖复合主键 (composite key) 的一部分。
- **3NF**: 没有列依赖另一个非键列。

未范式化(差)	已范式化(更好)
`student(name, club1, club2)`	`student(name)` + `membership(student, club)`

常见错误

- 把重复的组拆到单独的表里 (规范化), 而不是用许多相似的列。
- 每张表只描述**一种**事物。
- 用外键把表连接起来, 外键指向另一张表的主键。