8 Databases – Part 2

Name: _____ Class: _____ Date: _____

Vocabulary

Write each word three times.

English	**Write it 3 times** (English)		
entity			
entity-relationship diagram			
cardinality			
link table			
normal forms			
Data Definition Language			
Data Manipulation Language			
join			
aggregate functions			

Recall

In **Part 1** you met the relational model (tables, records, fields, primary and foreign keys), referential integrity, and a simple `SELECT` query. Part 2 adds diagramming a database, tidying it up, and more SQL.

New ideas

Read these first. Each idea has a short worked example. Then do the Practice.

1 Entity-relationship (E-R) diagrams

An **entity** is a thing we store data about (a `CUSTOMER`). An **entity-relationship diagram** draws each entity as a box joined by relationship lines, marked with the **cardinality**:

- *one-to-one*, *one-to-many* (one customer has many orders), *many-to-many*.



A many-to-many relationship can't be stored directly – you add a **link table** holding the two foreign keys (e.g. an `ENROLMENT` table between `STUDENT` and `COURSE`).

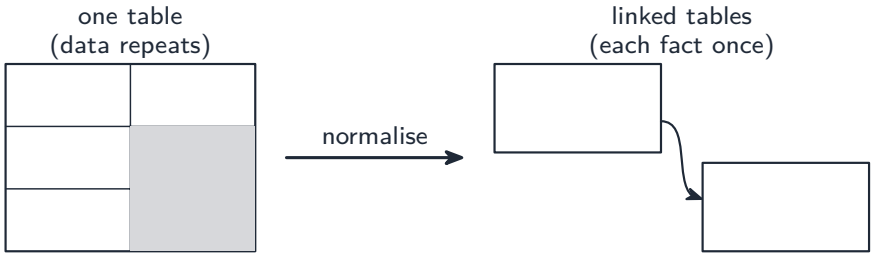
Worked example. `STUDENT` and `COURSE` are many-to-many (a student takes many courses; a course has many students), so an `ENROLMENT` link table holds both keys.

2 Normalisation

Normalisation organises tables into **normal forms** to remove repeated data:

- **1NF** – every field holds a single value (no repeating groups), with a primary key;
- **2NF** – no field depends on only *part* of a composite key;
- **3NF** – no non-key field depends on another non-key field.

Aim for **3NF**: each fact is stored once, so updates can't make copies disagree.

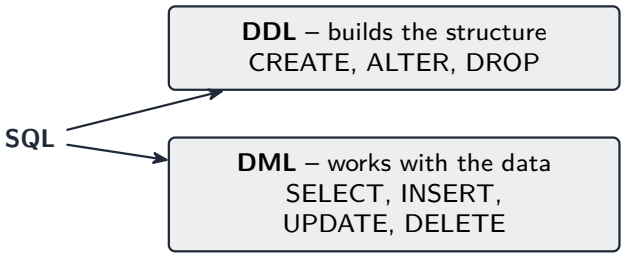


Worked example. A table that repeats a customer's name beside each of their orders is moved towards 3NF by holding the name once in a `CUSTOMER` table and linking by key.

3 SQL: DDL and DML

SQL has two halves: **Data Definition Language** (DDL) builds the *structure*, and **Data Manipulation Language** (DML) works with the *data*. A **join** combines two tables on a key; **aggregate functions** (`COUNT`, `SUM`, `AVG`) summarise rows.

exercises.lol exercises.com.cn




```
SELECT Name, Phone
FROM   CUSTOMER
WHERE  City = 'London'
ORDER BY Name;
```

Worked example. `CREATE TABLE` (DDL) builds an empty table; `INSERT INTO` (DML) then adds rows of data into it.

Watch out: a `JOIN` needs a key that appears in *both* tables – with no shared key field you cannot link them.

Practice

8.1 A customer can place many orders, but each order belongs to one customer. State the cardinality of this relationship. [1]

8.2 State whether `CREATE TABLE` is a DDL or a DML statement. [1]

8.3 State how a many-to-many relationship is stored in a relational database. [1]

8.4 State what it means for a table to be in first normal form (1NF). [2]

8.5 Write a SQL query that selects the `Name` of every row in a table `STUDENT` where the `Grade` is greater than 80. [3]

8.6 Explain *why* aiming for 3NF (each fact stored once) makes a database safer to update. [2]

8.7 Challenge. Write a SQL query that counts how many rows are in a `CUSTOMER` table. [2]

exercises.lol exercises.com.cn