Databases

IB DP Computer Science A3 Databases full package

One spreadsheet, many problems

> A database stores each fact once.

- Theme A3: relational databases, design and **normalisation**, SQL.
- Prerequisites: data types - integer, string, Boolean, real.

Why a relational database?

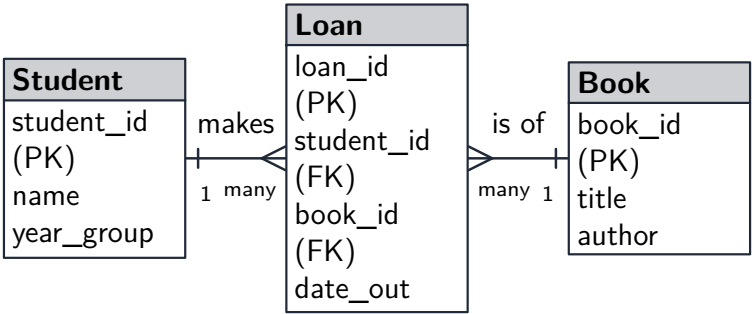
Flat-file problem	**Relational fix**
redundancy: data stored many times	each fact stored once
inconsistency: copies disagree	one copy, nothing to disagree
hard to search and combine	SQL joins tables
no control of who sees what	accounts and permissions

Primary and foreign keys

> A **foreign key** links a **record** to another table.

- **Composite key**: a primary key made of two or more fields.
- Referential integrity: a non-null foreign key must match an existing key.

Entities and relationships

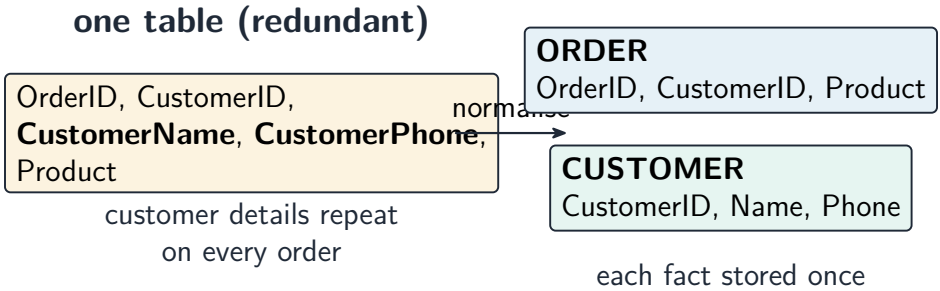


A **link table** turns many-to-many into two one-to-manys.

Normalising an order table

Form	Rule	Fix
1NF	one atomic value per **field**	one row per item
2NF	no field depends on part of the key	customer to Order; price to Item
3NF	no field depends on a non-key field	phone to Customer

The tables in 3NF



> Li Na's phone number is now stored once.

SELECT, WHERE, ORDER BY

Result: Chen, Ana.

- SELECT name FROM Student WHERE year_group = 11 ORDER BY name DESC;
- WHERE filters rows; DESC sorts Z to A.

Joining three tables

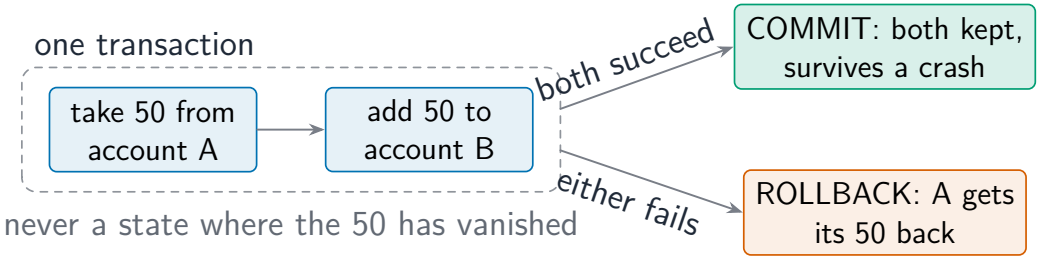
> Ben has no loans, so he does not appear.

- **JOIN** Student ON Loan.student_id = Student.student_id
- JOIN Book ON Loan.book_id = Book.book_id

Groups and aggregates

Clause	Use
WHERE	filter rows: $=$, $<>$, $>$, LIKE 'D%'
ORDER BY	sort, ASC or DESC
GROUP BY	make groups for aggregates
HAVING	filter groups

Changing data safely



No WHERE, and every row changes.

Transactions and ACID

Property	**Meaning**
atomicity	all of it happens, or none of it
consistency	one valid state to another
isolation	transactions at once do not interfere
durability	once committed, it survives a crash

The order SQL works in

the order the database runs a grouped query



COUNT(*) only exists after GROUP BY, so “more than 20 students” must be a HAVING filter.

> SELECT returns the answer

Practice

Your turn

Try these:

A. Explain why students and books need a Loan table.
B. Normalise (booking_id, room, room_capacity, date).
C. Write SQL for the titles of all books by Austen.
D. Explain atomicity with a bank transfer.

Check yourself

