3.2 SQL queries and transactions

Name: _____ Class: _____ Date: _____ **Total: 22 marks**

KEY WORDS join • aggregate function • transaction

Objective

Build the skills to answer exam questions on **SQL** – selecting, filtering, sorting, joining and grouping, changing data, and transactions.

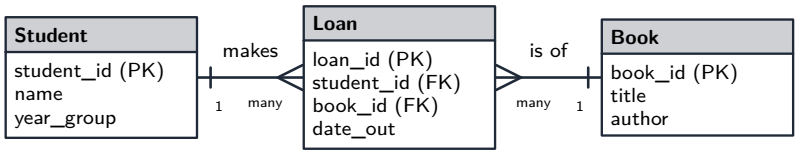
You must be able to:

- write `SELECT` queries with `WHERE`, `ORDER BY`, `LIKE` and Boolean conditions
- **join** two or three tables on matching keys
- use **aggregate functions** with `GROUP BY` and `HAVING`
- write `INSERT`, `UPDATE` and `DELETE` statements
- explain a **transaction** and the ACID properties

1 Key knowledge and worked examples

Study these first. Each one shows the method for a question type used later.

■ Key knowledge: the tables



Student		
student_id	name	year_group
1	Ana	11
2	Ben	12
3	Chen	11

Book		
book_id	title	author
10	Dune	Herbert
11	Emma	Austen
12	Holes	Sachar

Loan			
loan_id	student_id	book_id	date_out
100	1	10	2026-09-01
101	1	12	2026-09-05
102	3	11	2026-09-06

■ Worked example: a join with grouping

List each student's name and number of loans, most loans first.

```
SELECT Student.name, COUNT(*) AS loans
FROM Student
JOIN Loan ON Student.student_id = Loan.student_id
GROUP BY Student.name
ORDER BY loans DESC;
```

1. `JOIN … ON` matches each loan to its student: three joined rows.
2. `GROUP BY` makes one group per name; `COUNT(*)` counts its rows.
3. Result: (Ana, 2), (Chen, 1). Ben has no loans, so he has no joined row.

■ Key knowledge: changing data and ACID

- `INSERT INTO t (a, b) VALUES (1, 'x');` `UPDATE t SET a = 2 WHERE …;` `DELETE FROM t WHERE …;`
- Without `WHERE`, `UPDATE` and `DELETE` affect every row.
- ACID: **a**tomicity (all or nothing), **c**onsistency (valid to valid), **i**solation (no interference), **d**urability (survives a crash).

2 Practice

2.1 Write a query to list the names of students in year 12. [2]

2.2 Write a query to list book titles that start with the letter H, in alphabetical order. [3]

2.3 State the result of `SELECT title FROM Book WHERE book_id <> 11;`. [1]

2.4 Write a statement to add the book 13, 'Wonder', by Palacio. [2]

3 Exam-style questions

3.1 Which clause filters **groups** after `GROUP BY`? [1]

- ☐ **A** `WHERE`
- **B** `HAVING`
- **C** `ORDER BY`
- **D** `SELECT`

3.2 What does `DELETE FROM Loan;` do? [1]

- ☐ **A** deletes the Loan table's design
- **B** deletes every row of Loan
- **C** deletes nothing without `WHERE`
- **D** deletes the last row

3.3 Which ACID property ensures a committed change survives a power cut? [1]

- ☐ **A** atomicity
- **B** consistency
- **C** isolation
- **D** durability

3.4 Use the Student, Book and Loan tables.

(a) Write a query to list the title of every book borrowed by Ana. [4]

(b) State the output of your query. [1]

(c) Book 10 is returned. Write a statement to remove its loan record. [2]

(d) Chen moves to year 12. Write a statement to record this. [2]

(e) A transfer between two library accounts is processed as a transaction. Explain why atomicity matters. [2]

Solutions

Each answer shows the steps, then the **result**.

2.1

- `SELECT name FROM Student`
- `WHERE year_group = 12;`

2.2

- `SELECT title FROM Book`
- `WHERE title LIKE 'H%'`
- `ORDER BY title;`

2.3

- **Dune, Holes**

2.4

- `INSERT INTO Book (book_id, title, author)`
- `VALUES (13, 'Wonder', 'Palacio');`

3.1 B

- `WHERE` filters rows before grouping

3.2 B

- no `WHERE`, so every row

3.3 D

- durability

3.4

(a) One answer:

```
SELECT Book.title
FROM Loan
JOIN Student ON Loan.student_id = Student.student_id
JOIN Book ON Loan.book_id = Book.book_id
WHERE Student.name = 'Ana';
```

- selects `Book.title`
- joins Loan to Student on `student_id`
- joins Loan to Book on `book_id`
- `WHERE` name is `'Ana'`

(b)

- **Dune, Holes**

(c)

- `DELETE FROM Loan`
- `WHERE book_id = 10;` (or `loan_id = 100`)

(d)

- `UPDATE Student SET year_group = 12`
- `WHERE student_id = 3;`

(e)

- both the withdrawal and the deposit must happen, or neither
- if the system fails part-way, the transaction is rolled back, so no money is lost or created