Databases & Web Programming

Sample Paper 2 — Practice Exam

University of Birmingham

School of Computer Science

Module:	Databases & Web Programming
Exam:	Sample Paper 2 (Practice)
Duration:	2 hours
Total marks:	100

Instructions to candidates

- Answer **all three** questions.
- Marks for each part are shown in **red** on the right.
- This paper covers the same syllabus as Paper 1 with different scenarios.
- Where a curl command is asked for, assume the server is on `localhost:8080` unless told otherwise.

Question 1 — Databases

Total: 30 marks

(a) Schema Design from Scratch

A gym keeps its bookings in one big table called `Booking` with columns: `BookingID`, `MemberID`, `MemberName`, `MemberEmail`, `ClassName`, `ClassRoom`, `Instructor`, `InstructorPhone`, `StartTime`, `Duration`. The gym has noticed that when an instructor's phone number changes they have to update it on dozens of rows.

(i) Identify the **type of anomaly** the gym is currently suffering and one further anomaly that this design will produce.

[4 marks]

(ii) Write an SQL **CREATE TABLE** statement for a redesigned `Instructor` table that fixes the problem. Include: primary key, NOT NULL on the name, UNIQUE on the phone number, and an appropriate data type for each column.

[6 marks]

(b) From 1NF to 3NF

Consider the following table that a hotel uses to record stays. A guest can stay in multiple rooms over time. Each room has exactly one type (e.g. “Deluxe”), and each type has a fixed nightly rate.

StayID	GuestName	GuestPhone	RoomNo	RoomType	NightlyRate	CheckIn	Nights
1001	Aisha Lee	07700-1	204	Deluxe	150.00	2026-04-01	3
1002	Aisha Lee	07700-1	510	Suite	260.00	2026-05-12	2
1003	Tom Hart	07700-2	204	Deluxe	150.00	2026-05-15	1
...	...	...	...	...	...	...	...

Normalise this table into **3NF**. For each resulting table, list its columns and clearly mark Primary Keys with **(PK)** and Foreign Keys with **(FK)**. State which **functional dependency** drove each decomposition step.

[10 marks]

(c) SQL Queries

Two tables exist in an online-bookstore database:

```
Author(AuthorID PK, Name, Country)
Book(BookID PK, AuthorID FK, Title, Genre, Price)
```

(i) Write a SQL query that returns the **Name** of every author together with the **number of books** they have written. **Include authors who have written zero books.** Sort results by the book count, highest first.

[6 marks]

(ii) Briefly explain why an **INNER JOIN** would not give the right result for this query.

[4 marks]

Question 2 — Web Development Principles

Total: 30 marks

(a) Path Variables vs Request Parameters

An online-shopping API uses two endpoints:

```
GET /api/v1/products/2025
GET /api/v1/products?category=phones&maxPrice=500
```

(i) Explain the difference between a **path variable** and a **request parameter**, using these two examples.

[6 marks]

(ii) Give a scenario where you must use a path variable, and a scenario where a request parameter is clearly the right choice. Justify each.

[4 marks]

(b) HTTP Methods & Status Codes

(i) A developer issues `POST /api/v1/products/2025` intending to update an existing product. The server rejects it. State the **most likely 4xx error code** and explain why.

[4 marks]

(ii) Compare **PUT** and **PATCH**. Give one situation where PATCH is preferable to PUT.

[4 marks]

(c) Spring Boot Project Structure

In a generated JHipster Spring Boot project under `src/main/java`, four folders matter for entity-driven features: `domain/`, `repository/`, `service/`, `web/rest/`.

Briefly explain the responsibility of each folder, name **one** typical annotation you would see in each, and describe what a single feature looks like when sliced vertically through them.

[12 marks]

Question 3 — Web Development Practice

Total: 40 marks

(a) JDL Modelling — Many-to-Many

You are modelling a conference-management app. Write JDL that:

- Defines a **Conference** entity with: `name` (String, required), `startDate` (LocalDate, required), `endDate` (LocalDate, required), `website` (String) and a `summary` (TextBlob).
- Defines a **Speaker** entity with: `firstName` (String, required), `lastName` (String, required), `bio` (TextBlob).
- A Speaker can speak at many Conferences; a Conference can have many Speakers (many-to-many).
- Every Conference is created by exactly one organising User (built-in entity).
- Set the application to use **pagination** for both entities and a **service class** for Conference.

[15 marks]

(b) cURL & HTTP

Using the model from (a), write the cURL command that would:

(i) **POST** a new Conference titled “Spring Connect 2026” starting `2026-09-12` and ending `2026-09-14`. Include the `Authorization: Bearer <TOKEN>` header.

[5 marks]

(ii) The team sends `DELETE /api/v1/conferences/77` but the server returns 405. Explain what **405 Method Not Allowed** means and what is most likely wrong with the backend.

[5 marks]

(c) Custom Repository Query

You need an endpoint that lists every Conference happening in a given year, sorted by start date.

(i) Write the **Spring Data JPA** repository method signature using a derived query name (no `@Query`).

[5 marks]

(ii) Now write the equivalent method using a **JPQL** `@Query` annotation. Explain when you would prefer JPQL over a derived query name.

[5 marks]

(d) Liquibase & Schema Migration

Your app is in production. You need to add a `maxAttendees` (Integer) column to the `Conference` table without losing existing data.

(i) Describe in 3-4 sentences how **Liquibase** handles this safely. What is the role of the `master.xml` changelog and the per-change `changeSet` id/author tuple?

[5 marks]

— End of Sample Paper 2 —