

Your overall mark for the paper is **43/120**. This result indicates a need to revisit fundamental concepts, especially in SQL syntax and relational design theory.

Where You Went Wrong (Detailed Breakdown)

This section identifies the critical errors in each question, which resulted in significant mark loss.

Question 1: Entity Relationship Diagram (ERD) - (7/20 Marks)

Requirement	Student Error
M:M Relationship Resolution	<p>[cite_start]The most critical error was the failure to correctly resolve the Many-to-Many (M:M) relationship between Medication and Active Ingredients[cite: 622, 623]. You showed a direct M:M relationship with confusing multiplicities, and an incorrect associative entity was implied or named (Active Ingredients Category).</p>
Associative Entity/Attributes	<p>[cite_start]You missed the associative entity (Medication_Ingredient or similar) which is necessary to store the non-key attribute Quantity[cite: 623], as this attribute depends on both the MedicationID and the Active IngredientsID.</p>
UML Notation/Multiplicity	<p>The notation used (e.g., 1..* or 1...1) was inconsistent with standard UML notation for multiplicity (which uses 1..1, 1..*, 0..1, etc.) and was incorrectly applied to the relationships.</p>

Missing Foreign Keys	While the primary keys were identified, the final placement of the Foreign Keys (FKs) in the derived 1:M relationships (e.g., from Category to Medication, and from Category to Active Ingredients) was not correctly represented in the decomposition.
-----------------------------	--

Question 2: Normalisation (2NF & 3NF) - (8/20 Marks)

Requirement	Student Error
Identifying the Primary Key (1NF)	[cite_start]You did not explicitly or correctly identify the Primary Key for the initial 1NF relation $R(\underline{\text{PharmacyID}}, \text{PharmacyName}, \text{GroupID}, \text{GroupName}, \text{PharmacistID}, \text{PharmacistName})$ [cite: 635, 641]. Based on the sample data, PharmacyID is the unique identifier (the key).
2NF: Definition of Partial Dependency	Assuming a single key ($\underline{\text{PharmacyID}}$), the relation is technically already in 2NF, as a partial dependency only occurs when a non-key attribute depends on <i>part</i> of a composite primary key. You incorrectly identified partial dependencies when you should have moved directly to identifying transitive dependencies (3NF).
3NF: Transitive Dependencies	While you correctly identified the functional dependencies: $\text{GroupID} \rightarrow \text{GroupName}$ and $\text{PharmacistID} \rightarrow \text{PharmacistName}$ (these are transitive dependencies in 2NF), you failed to

	preserve the original relation structure.
Decomposition and Referencing	Your final 3NF decomposition was incomplete. You successfully created separate tables for Group and Pharmacist , but you neglected to leave the Foreign Keys (GroupID and PharmacistID) in the main Pharmacy table to maintain referential integrity.

Question 3: MySQL (SQL) - (15/65 Marks)

This section suffered heavily from **critical syntax and logic errors**, preventing the execution of the script.

Q#	Requirement	Student Error
3.1, 3.2	CREATE TABLE	[cite_start] Critical Syntax Error: The column definitions and constraints were placed <i>outside</i> the main CREATE TABLE (...) parentheses, which causes the entire statement to fail[cite: 360, 368, 369].
3.3	CREATE Appointments	[cite_start] Critical Syntax Error: The second FOREIGN KEY reference (REFERENCES Doctor (DoctorID)) was placed <i>outside</i> the table definition parentheses[cite: 380, 382]. [cite_start]Also, a spelling error in the foreign key reference: Pait**ie**ntID instead of

		Pait**e**ntID[cite: 378].
3.4	INSERT Appointments	<p>[cite_start]</p> <p>Logic Error: The order of the PatientID and DoctorID values in your VALUES list did not match the expected data from the question paper[cite: 389, 687].</p> <p>[cite_start]Also, poor formatting with quotation marks and line breaks would likely cause a syntax error in MySQL[cite: 392, 394].</p>
3.5	SELECT Appointments	<p>Logic Error: The filter condition was incorrect: ... AND AppointmentDate >= "2024-01-20". This finds appointments <i>after</i> Jan 20th. [cite_start]For an inclusive range, you must use ... AND AppointmentDate <= "2024-01-20", or use the BETWEEN operator[cite: 418, 419].</p>
3.6, 3.7	JOIN Conditions	<p>Critical Syntax Error: You frequently omitted the equality operator (=) in your JOIN conditions, e.g., P. [cite_start]PaitentID A. PaitentID[cite: 422, 429]. This is a fatal syntax error.</p>
3.6, 3.7	ORDER BY	<p>[cite_start]</p> <p>Logic Error: You were required to sort in descending order for Q3.6 [cite: 693][cite_start]and</p>

		Q3.7[cite: 696], but you failed to add the DESC keyword, resulting in the default ascending sort.
3.6	GROUP BY	Syntax/Logic Error: When using the GROUP BY clause, all non-aggregated columns in the SELECT list must be included. [cite_start]You only grouped by Paitent Surname, but PaitentName was also selected[cite: 423].
3.8	CREATE VIEW	[cite_start] Logic Error: You included A.DoctorID in the SELECT list when the requirement strictly asks for just the name and surname of the patients [cite: 699]. [cite_start]Incorrect JOIN on AppointmentID = PaitentID instead of the FKs[cite: 435].
3.9	STORED PROCEDURE	[cite_start] Logic/Design Error: You introduced unnecessary variable declaration and SELECT INTO logic that was confusing and did not achieve the required filter [cite: 445-449]. [cite_start]You also entirely missed the final WHERE clause needed to filter the results by the input AppointmentDate[cite: 706].

Question 4: MongoDB (NoSQL) - (13/15 Marks)

Q#	Requirement	Student Error
4.2	insertMany	[cite_start] Minor Syntax Error: Your initial insertMany attempt failed because the documents were not enclosed in a single array [] [cite: 555]. While you successfully used two insertOne commands, mastering insertMany is the more professional and efficient approach for bulk loading.
4.2	Naming	[cite_start] Spelling Error: The collection was named Paitents instead of patients [cite: 551]. Consistency in naming is crucial in professional environments.
4.4	find().pretty()	[cite_start] Minor Syntax Error: Your initial query attemptedpretty without the function call parenthesis () [cite: 581], which is a syntax error in the shell. The second attempt was correct, but you should always aim for the fully formatted command: .pretty().

How to Improve Your Mark 🚀

To turn your grade around, focus on these three core areas:

1. Mastering SQL Syntax and Execution (Q.3):

- **Prioritise DDL:** You must be able to create tables correctly. A single missing parenthesis or misplaced clause (like in Q.3.1, 3.2, 3.3) means all subsequent queries (Q.3.4 - Q.3.9) automatically fail.
 - **Rule:** Every table, every column, and every constraint **must** be inside the main CREATE TABLE (...) block.
- **Master JOIN Operators:** Missing the = operator in your JOIN conditions is a critical error. A JOIN is useless without the explicit condition that links the two tables' foreign and primary keys (ON TableA.FK = TableB.PK).
- **Use Specific Keywords:** When a question specifies sorting, you *must* use DESC for descending order. The default is always ascending (ASC).

2. Solidifying Relational Design Concepts (Q.1 & Q.2):

- **Q.1 M:M Resolution:** This is a foundation of relational design. An M:M relationship **must** be resolved by creating a new **associative entity**. This new entity will have a **composite primary key** consisting of the two FKs from the original entities, and it will hold any non-key attributes (like **Quantity**) that depend on the combination of the two.
- **Q.2 Normal Forms:** Understand the exact **definition** of each normal form:
 - **2NF:** No non-key attribute is dependent on only *part* of a **composite** primary key.
 - **3NF:** No non-key attribute is dependent on another **non-key** attribute (no transitive dependency).

3. NoSQL Flexibility (Q.4):

- Be aware of NoSQL data loading conventions. For multiple documents, always format the data as a JavaScript array [] within the insertMany() command to maximise efficiency.

Guideline to Tackle Further Questions (Strategy) 💡

Adopt this structured approach for any future database assessment:

1. Relational Design (ERD & Normalisation)

Step	Action	Check
Identify Entities/Attributes	List all key nouns and their associated data (attributes) from the business rules.	Have you included all required attributes?
Identify Relationships	Determine the connections between the entities (e.g., Medication <i>belongs to</i> Category).	Have you named all relationships clearly?
Resolve M:M (Q.1)	CRITICALLY: Convert all M:M relationships into two 1:M relationships connected by a new associative entity.	Does the associative entity have a composite primary key? Does it hold the necessary non-key attributes?
Identify Dependencies (Q.2)	Key Identification: Determine the Primary Key (PK) of the initial relation. Functional Dependency: Identify all \$A \to B\$ dependencies.	Is the PK a single column or a composite key? Are there any non-key attributes depending on other non-key attributes?
Decompose to 3NF	For 3NF: Remove the transitive dependencies by creating new tables (e.g., GroupID \$\to\$	Are your final relations free of partial and transitive dependencies? Have you

	GroupName). The original table MUST retain the GroupID as a Foreign Key (FK) to link back to the new table.	maintained all FKs?
--	--	---------------------

2. SQL Implementation (Q.3)

Step	Action	Check
DDL (CREATE)	Write your CREATE TABLE statements. Define columns, data types, NOT NULL constraints, and the PRIMARY KEY inside the (...). Define FOREIGN KEY constraints at the end of the column definitions, still inside the (...).	Is all code within the parentheses? Have you checked for spelling/typos in table and column names?
DML (INSERT)	Write your INSERT statements. Pay close attention to the order of columns in the column list and the order of values in the VALUES list.	Do the column count and value count match? Does the data type match (e.g., integers for IDs, dates for DOB)?
Querying (SELECT/JOIN)	For complex queries, draft the steps: 1. Tables: Which tables are needed? 2. JOIN: How do I link them (using PK = FK and the equality operator =!)? 3. FILTER: What WHERE conditions are required? 4. GROUP: Do I need aggregation? 5. SORT:	Is the JOIN condition correct? Have I used DESC if the question requires descending order?

	What is the final ORDER BY and is it ASC or DESC?	
--	---	--

3. NoSQL (MongoDB) (Q.4)

Step	Action	Check
Context	Use the use command to switch to the required database.	Is the database name correct and in the required format?
Data Loading	Use db.<collectionName>.insert Many([<doc1>, <doc2>, ...]).	Are all documents enclosed in a single array []? Have you checked for field name spelling?
Querying	Use \$gt (greater than), \$lt (less than), \$gte (greater than or equal), etc., for numerical/date comparison queries. Remember to enclose the field name in quotes if it contains spaces (e.g., "Date Of Birth").	Is the operator correct for the required condition (e.g., \$gt for <i>after</i> , not including)? Have you added .pretty() for clean, readable output?