MySQL Assignment Questions

---

1. What is a Database? / Overview

1. Explain what a database is and give 3 examples of where databases are used in real life.
2. Compare SQL and NoSQL databases with examples.
3. What is ACID compliance, and why does it matter for relational databases?
4. Discuss scenarios where you would prefer NoSQL over SQL.
5. Install MySQL Workbench and connect it to a local MySQL server. Provide screenshots.
6. Compare MySQL, PostgreSQL, and SQLite—list pros and cons of each.
7. Write commands to connect to MySQL using CLI.
8. Describe the role of a RDBMS in an application stack.
9. Explain how ER diagrams help in designing databases.
10. What are the advantages of using a GUI tool like Workbench over CLI?

---

2. Database Design & Table Creation

11. Define the terms: database, table, row, column.
12. Create a database called `library_db`.
13. Create a table `books` with columns: id (PK), title, author, published_year, genre.
14. Explain primary key vs foreign key with examples.
15. What is a candidate key? Give an example.
16. Design tables for a one-to-many relationship between `customers` and `orders`.
17. Design tables for a many-to-many relationship between `students` and `courses`.
18. Write SQL to create `students` and `courses` tables with an association table `enrollments`.
19. Modify the `books` table to add a column `isbn` with UNIQUE constraint.
20. Drop the `library_db` safely after confirmation.
21. Create a table with columns having NOT NULL and DEFAULT constraints.
22. What is AUTO_INCREMENT? Create an example table that uses it.
23. Write SQL to rename a column in `books` from `genre` to `category`.
24. Alter the `books` table to change `category` data type from VARCHAR(50) to VARCHAR(100).
25. Drop the `books` table and explain the impact on related tables if foreign keys exist.

---

3. CRUD Operations (Insert, Read, Update, Delete)

26. Insert a single record into `books`.
27. Insert multiple rows into `books` in one query.
28. Insert a record without specifying all columns—use DEFAULT values.
29. Insert 500 sample records into a test table—what technique will you use for bulk insert?
30. Retrieve all records from `books`.
31. Retrieve only `title` and `author` columns.
32. Use WHERE to filter books published after 2010.
33. Retrieve books where `author` is 'Agatha Christie' OR `category` is 'Mystery'.
34. Find records where `category` is NOT NULL.
35. Select books where `title` starts with 'A'.
36. Select books published BETWEEN 2000 and 2010.
37. Update the `category` of all books by 'J.K. Rowling' to 'Fantasy'.
38. Increase the `price` of all books by 10%.
39. Update multiple columns in one UPDATE query.
40. Delete records where `published_year` is before 1990.
41. TRUNCATE the `books` table—explain the difference between TRUNCATE and DELETE.
42. Use LIMIT and OFFSET to paginate results in `books`.
43. Retrieve books ordered by `published_year` descending.
44. Retrieve top 5 most recently published books.

---

4. Querying – Sorting, Filtering, Grouping

45. Use AS to alias a column as `BookTitle`.
46. Concatenate `author` and `title` into a single column `BookInfo`.
47. Use arithmetic in SELECT to calculate discounted price.
48. Count how many books per category exist.
49. Calculate average published_year per author.
50. Use HAVING to filter groups with more than 3 books.
51. What is the difference between WHERE and HAVING? Provide examples.
52. Write a query to find the maximum published year per category.
53. Find the total number of books.
54. Retrieve all books grouped by `author`.
55. Write a query using GROUP BY and ORDER BY together.

---

5. Joins – Combining Data

56. Create `publishers` table and insert data.
57. INNER JOIN `books` and `publishers` to display book titles with publisher names.
58. LEFT JOIN to show all books even if they have no publisher.
59. RIGHT JOIN to list all publishers and their books.

60. Simulate FULL OUTER JOIN in MySQL.
61. CROSS JOIN `books` and `categories`.
62. Join 3 tables (`books`, `authors`, `publishers`) in a single query.
63. Use table aliases in JOINs.
64. Write a self-join to show books in the same category.
65. Explain when to use JOINs vs Subqueries with examples.
66. Create a subquery to find books with the maximum published_year.
67. Write a correlated subquery to get books published after the average published_year of their category.

---

6. Subqueries, Views & Indexing

68. Create a view `recent_books` for books after 2015.
69. Query the `recent_books` view.
70. Drop the `recent_books` view.
71. Explain read-only vs updatable views.
72. Create an index on `published_year`.
73. Create a composite index on `author` and `category`.
74. Drop an index.
75. Discuss how indexes affect performance with examples.
76. Use EXISTS to check if any books in 'Science' category exist.
77. Use IN with a subquery to select books by authors with more than 2 books.
78. Use ANY and ALL operators in subqueries.
79. Write a scalar subquery in SELECT to get total books per author.

---

7. Functions & Stored Procedures

80. Use CONCAT to merge first and last name of authors.
81. Use UPPER to display all titles in uppercase.
82. Extract year from `published_date`.
83. Round a numeric column in a query.
84. Create a user-defined function that returns the age of a book.
85. Create a stored procedure to insert a book record.
86. Call the stored procedure with parameters.
87. Create a procedure with OUT parameter returning total count of books.
88. Create a procedure using IF...ELSE logic.
89. Create a procedure that loops over years and prints counts.
90. Write a use case of stored procedures for admin tasks.

---

8. Triggers, Transactions & Security

91. Create a BEFORE INSERT trigger to set `created_at` timestamp.
92. Create an AFTER UPDATE trigger to log changes in a log table.
93. Write a trigger that prevents deletion of books published after 2015.
94. Explain use cases of triggers for auditing.
95. Start a transaction to insert multiple records.
96. Rollback a transaction after inserting records.
97. Use SAVEPOINT and ROLLBACK TO SAVEPOINT.
98. Commit a transaction.
99. Show how to create a user `report_user`.
100. Grant SELECT privileges to `report_user`.
101. Revoke privileges from `report_user`.
102. Explain secure password policies in MySQL.

---

9. Integration & Tools

103. Write a Node.js script to connect to MySQL and query `books`.
104. Write a Python script using `mysql.connector` to insert a record.
105. Write a PHP script to select all records.
106. Demonstrate a parameterized query to prevent SQL injection.
107. Export `books` table to CSV.
108. Import CSV data into `books` table.
109. Use `mysqldump` to backup the `library_db`.
110. Restore the database from a dump file.
111. Enable slow query log.
112. Use EXPLAIN to analyze a query.
113. Optimize a slow query with an index.
114. Write a query to benchmark performance of COUNT vs EXISTS.

---

10. Real-World Projects

Project 1: Inventory Management DB

115. Design tables: `products`, `categories`, `suppliers`.
116. Create triggers to update stock quantity after sales.
117. Create a view `monthly_sales_report`.
118. Insert sample data into `products` and `suppliers`.
119. Write queries to retrieve low stock products.
120. Generate a report showing top-selling products.

Project 2: User Authentication System

121. Create `users` and `login_log` tables.
122. Write a stored procedure `log_login_attempt`.
123. Insert a login attempt record via the procedure.
124. Assign roles to users.
125. Write a query to retrieve all failed login attempts.
126. Create a trigger to lock an account after 3 failed attempts.
127. Demonstrate use of transactions in recording logins.

---

11. Advanced SQL (Challenging)

128. Write a query to find the second highest price in `products`.
129. Write a recursive CTE to generate dates of the past 30 days.
130. Use window functions to rank books by published_year.
131. Use JSON columns and functions in MySQL 8.
132. Write a stored procedure that dynamically builds and executes a query.
133. Design a schema to track hierarchical categories (adjacency list).
134. Use REGEXP to filter titles starting with a digit.
135. Write a query to pivot sales data (simulate PIVOT).
136. Create a full-text index and search for keywords.
137. Demonstrate partitioning a large table by range.
138. Use generated columns to store derived values.
139. Write a procedure that accepts JSON input.
140. Create a procedure for batch insert with error handling.

---

12. Miscellaneous

141. Compare INNER JOIN and LEFT JOIN performance.
142. Describe pros and cons of denormalization.
143. Implement a simple audit trail for `orders`.
144. Write a backup and restore strategy for production.
145. Configure user roles and access control for different teams.
146. Discuss ACID properties in the context of banking transactions.
147. Design indexes for optimal performance in a reporting database.
148. Create an ER diagram for an e-commerce database.
149. Write a script to anonymize user data.
150. Discuss common pitfalls when migrating from SQL Server to MySQL.

---

13. Extra Practice Queries

151. Count books per year and filter only years with more than 5 books.
152. List categories with no books.
153. Update prices by 15% only for books older than 10 years.
154. Find authors with multiple categories.
155. Retrieve books with titles containing 'Guide'.
156. Delete all books with NULL category.
157. Show books with the longest titles.
158. Group by author and show min and max published year.
159. List books sorted by length of title.
160. Write a query to get books where the title has more than 3 words.
161. Add a column `last_modified` with default CURRENT_TIMESTAMP.
162. Update `last_modified` automatically on record update (trigger).
163. Write a query to calculate age of books.
164. Use DATE_FORMAT to display date in 'Month-Year'.
165. Use CASE to label books as 'Classic' if before 1980, else 'Modern'.

---

14. Integration & Security (Advanced)

166. Create a user for an application with limited privileges.
167. Demonstrate using SSL for secure connections.
168. Create a stored procedure to batch update prices.
169. Write a procedure to archive old records into an `archive_books` table.
170. Create a trigger to prevent updating ISBN once set.
171. Demonstrate error handling in stored procedures.
172. Discuss benefits and risks of dynamic SQL.
173. Write a script to rotate logs periodically.
174. Use performance schema to analyze slow queries.
175. Show how to enable binary logging.

---

15. Data Export & Import

176. Export data to JSON format.
177. Import JSON data into a table.
178. Use LOAD DATA INFILE to bulk import CSV.
179. Compare `mysqldump` and `mysqlpump`.
180. Explain replication basics and set up a simple master-slave replication.

---

16. Advanced Projects

181. Design a content management system schema.
182. Create an audit log system with triggers and views.
183. Design a ticket booking system.
184. Create a stored procedure to generate monthly invoices.
185. Build a report showing sales trends over time.
186. Implement versioning for records.
187. Create a REST API using Node.js to query MySQL.
188. Write a script to migrate data from SQLite to MySQL.
189. Create complex views combining multiple tables.
190. Create a procedure to detect duplicate records.

---

17. Bonus Challenges

191. Create a stored procedure to send email notifications (using UDF or external integration).
192. Write queries using common table expressions (CTEs) for hierarchical data.
193. Implement row-level security.
194. Design a partitioning strategy for large time-series data.
195. Use MySQL event scheduler to automate cleanup tasks.
196. Create a function to validate email formats.
197. Write a query to find gaps in a sequence of IDs.
198. Use window functions to calculate running totals.
199. Discuss scaling MySQL for high traffic applications.
200. Explain strategies for zero-downtime migrations.