REPORT

507-Mitali Bang

508-Arpita Barjibhe

524-Kalyani Fulpagare

525-Tanvi Gagan

DBMS-QUERIES

Enter password: **********

Welcome to the MySQL monitor. Commands end with ; or \g.

Your MySQL connection id is 8

Server version: 9.0.1 MySQL Community Server - GPL

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> create database miniproject;

Query OK, 1 row affected (0.01 sec)

```
mysql> use miniproject;
Database changed
mysql> CREATE TABLE Player (
    ->     id INT PRIMARY KEY AUTO_INCREMENT,
    ->     name VARCHAR(100),
    ->     dob DATE
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE TABLE Questions (
    ->     id INT PRIMARY KEY AUTO_INCREMENT,
    ->     question_text VARCHAR(255),
    ->     option_1 VARCHAR(100),
    ->     option_2 VARCHAR(100),
    ->     option_3 VARCHAR(100),
    ->     option_4 VARCHAR(100),
    ->     correct_option INT ,
    ->     difficulty_level VARCHAR(10)
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE TABLE Score (
```

```
    ->     player_id INT PRIMARY KEY,
    ->     total_score INT DEFAULT 0,
    ->     highest_score INT DEFAULT 0,
    ->     correct_answers INT DEFAULT 0,
    ->     wrong_answers INT DEFAULT 0,
    ->     attempted_questions INT DEFAULT 0,
    ->     FOREIGN KEY (player_id) REFERENCES Player(id)
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE TABLE Leaderboard (
    ->     player_id INT PRIMARY KEY,
    ->     total_score INT DEFAULT 0,
    ->     highest_score INT DEFAULT 0,
    ->     position INT DEFAULT 0,
    ->     FOREIGN KEY (player_id) REFERENCES Player(id)
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> DELIMITER $$
mysql>  CREATE PROCEDURE AnswerQuestion(
    ->      IN player_id INT,
    ->      IN question_id INT,
```

```
    ->     IN player_answer INT
    ->    )
    ->    BEGIN
    ->     DECLARE correct_option INT;
    ->     SELECT correct_option INTO correct_option
    ->     FROM questions
    ->     WHERE id = question_id;
    ->       IF player_answer = correct_option THEN
    ->
    ->         UPDATE score
    ->         SET correct_answers = correct_answers + 1,
    ->         total_score = total_score + 10,  -- Add score for correct
answer
    ->
    ->         attempted_questions = attempted_questions + 1
    ->         WHERE player_id = player_id;
    ->         ELSE
    ->
    ->         UPDATE score
    ->         SET wrong_answers = wrong_answers + 1,
    ->         attempted_questions = attempted_questions + 1
    ->         WHERE player_id = player_id;
    ->         END IF;
```

```
    ->      END $$
Query OK, 0 rows affected (0.02 sec)

mysql>  DELIMITER ;
mysql> DELIMITER $$
mysql>
mysql>  CREATE PROCEDURE ViewPlayerProfile(IN player_id INT)
    -> BEGIN
    -> SELECT
    -> p.name,
    -> p.dob,
    -> s.total_score,
    -> s.correct_answers,
    -> s.wrong_answers,
    -> s.attempted_questions
    -> FROM player p
    -> JOIN score s ON p.id = s.player_id
    -> WHERE p.id = player_id;
    -> END $$
Query OK, 0 rows affected (0.01 sec)


mysql> DELIMITER ;
```

```
mysql> drop trigger updateLeaderboard;
Query OK, 0 rows affected (0.02 sec)


mysql> DELIMITER $$
mysql> CREATE TRIGGER UpdateLeaderboard
    -> AFTER UPDATE ON score
    -> FOR EACH ROW
    -> BEGIN
    -> DECLARE player_rank INT;
    -> SET player_rank = (SELECT COUNT(*)
    -> FROM score
    -> WHERE total_score > NEW.total_score) + 1;
    -> IF EXISTS (SELECT 1 FROM leaderboard WHERE player_id =
NEW.player_id) THEN
    -> UPDATE leaderboard
    -> SET total_score = NEW.total_score,
    -> highest_score = NEW.highest_score,
    -> position = player_rank
    -> WHERE player_id = NEW.player_id;
    -> ELSE
    -> INSERT INTO leaderboard (player_id, total_score, highest_score,
position)
```

-> VALUES (NEW.player_id, NEW.total_score, NEW.highest_score, player_rank);

-> END IF;

-> END $$

Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;

mysql> INSERT INTO questions (question_text, option_1, option_2, option_3, option_4, correct_option, difficulty_level)

-> VALUES

-> ('What is the primary key in a relational database?', 'Unique identifier', 'Foreign key', 'Null value', 'Index', 1, 'easy'),

-> ('Which SQL clause is used to filter records?', 'ORDER BY', 'GROUP BY', 'HAVING', 'WHERE', 4, 'medium'),

-> ('What is the purpose of an index in a database?', 'Increase storage', 'Speed up retrieval', 'Ensure uniqueness', 'None of the above', 2, 'easy'),

-> ('Which command is used to remove a table from a database?', 'DROP', 'DELETE', 'TRUNCATE', 'REMOVE', 1, 'easy'),

-> ('What does SQL stand for?', 'Structured Query Language', 'Simple Query Language', 'Sequential Query Language', 'Structured Queue Language', 1, 'medium'),

-> ('Which keyword is used to retrieve unique values?', 'DISTINCT', 'UNIQUE', 'SELECT', 'WHERE', 1, 'easy'),

-> ('Which SQL function is used to calculate the number of records?', 'AVG()', 'SUM()', 'COUNT()', 'TOTAL()', 3, 'easy'),

-> ('What is a foreign key used for?', 'Indexing', 'Referential integrity', 'Normalization', 'Data redundancy', 2, 'easy'),

-> ('Which SQL command is used to add data to a table?', 'INSERT', 'UPDATE', 'ADD', 'MODIFY', 1, 'medium'),

-> ('What type of join returns only matching records from both tables?', 'Left Join', 'Inner Join', 'Right Join', 'Cross Join', 2, 'easy'),

-> ('How can you rename a column in SQL?', 'RENAME TO', 'ALTER COLUMN', 'CHANGE COLUMN', 'ALTER TABLE', 4, 'medium'),

-> ('Which SQL constraint ensures that all values in a column are unique?', 'PRIMARY KEY', 'UNIQUE', 'CHECK', 'NOT NULL', 2, 'easy'),

-> ('Which of these SQL clauses is used for conditional aggregation?', 'HAVING', 'GROUP BY', 'WHERE', 'ORDER BY', 1, 'medium'),

-> ('Which SQL data type is used to store large amounts of text?', 'TEXT', 'CHAR', 'VARCHAR', 'STRING', 1, 'medium'),

-> ('How do you create a view in SQL?', 'CREATE INDEX', 'CREATE VIEW', 'CREATE TABLE', 'CREATE FUNCTION', 2, 'easy'),

-> ('Which command is used to remove duplicate records in a result set?', 'UNIQUE', 'DISTINCT', 'GROUP BY', 'WHERE', 2, 'easy'),

-> ('What is a stored procedure?', 'A table in the database', 'A saved SQL query', 'A trigger', 'An index', 2, 'medium'),

-> ('What is the default sorting order in SQL?', 'Ascending', 'Descending', 'Alphabetical', 'None', 1, 'easy'),

-> ('What does the BETWEEN operator do in SQL?', 'Finds exact values', 'Selects within a range', 'Excludes certain values', 'Joins two tables', 2, 'medium'),

-> ('Which function is used to get the current date in SQL?', 'GETDATE()', 'TODAY()', 'SYSDATE()', 'CURRENT_DATE()', 4, 'medium'),

->

-> -- Difficult questions with escaped apostrophes

-> ('What is a clustered index in SQL?', 'An index on non-primary columns', 'An index that alters the physical order of the table', 'A secondary index', 'A unique index', 2, 'difficult'),

-> ('Which SQL clause is used to combine rows from two or more tables, based on a related column between them?', 'JOIN', 'CONNECT', 'BIND', 'LINK', 1, 'difficult'),

-> ('How can you optimize a query that is running slow?', 'Use a SELECT * statement', 'Create indexes on frequently queried columns', 'Remove indexes', 'Add more columns', 2, 'difficult'),

-> ('What is the function of the WHERE clause in an UPDATE statement?', 'To specify columns to be updated', 'To filter rows to be updated', 'To group rows to be updated', 'To order rows to be updated', 2, 'difficult'),

-> ('How does a LEFT JOIN differ from an INNER JOIN?', 'Returns all records', 'Returns all records from the left table', 'Returns all records from the right table', 'Does not return unmatched rows', 2, 'difficult'),

-> ('Which of the following SQL concepts enforces rules to maintain data accuracy?', 'Normalization', 'Integrity Constraints', 'Indexing', 'Storage Procedures', 2, 'difficult'),

-> ('How does the GROUP BY clause differ from the ORDER BY clause?', 'GROUP BY groups data; ORDER BY sorts data', 'GROUP BY filters data; ORDER BY groups data', 'GROUP BY sorts data; ORDER BY filters data', 'GROUP BY and ORDER BY are the same', 1, 'difficult'),

-> ('In SQL, what does the term "ACID" stand for in database transactions?', 'Atomicity, Consistency, Isolation, Durability', 'Access, Control, Integrity, Distribution', 'Automation, Consistency, Isolation, Durability', 'Atomicity, Control, Isolation, Distribution', 1, 'difficult'),

-> ('What SQL keyword is used to perform a subquery?', 'INSERT', 'GROUP BY', 'SELECT', 'DISTINCT', 3, 'difficult');

Query OK, 29 rows affected (0.02 sec)

Records: 29 Duplicates: 0 Warnings: 0

mysql> select * from player;

```
+----+---------+------------+
| id | name    | dob        |
+----+---------+------------+
|  1 | Mitali  | 2005-06-13 |
|  2 | Arpita  | 2005-11-11 |
|  3 | tanvi   | 2001-12-30 |
|  4 | Kalyani | 1993-10-03 |
+----+---------+------------+
```

4 rows in set (0.00 sec)

mysql> select * from score;

+----------+------------+--------------+----------------+--------------+-----------------+

| player_id | total_score | highest_score | correct_answers | wrong_answers | attempted_questions |

+----------+------------+--------------+----------------+--------------+-----------------+

| 1 | 10 | 20 | 1 | 2 | 3 |

| 2 | 30 | 30 | 3 | 1 | 4 |

| 3 | 50 | 50 | 5 | 2 | 7 |

| 4 | 40 | 40 | 4 | 0 | 4 |

+----------+------------+--------------+----------------+--------------+-----------------+

4 rows in set (0.00 sec)

mysql> select * from leaderboard;

+----------+------------+--------------+----------+

| player_id | total_score | highest_score | position |

+----------+------------+--------------+----------+

| 1 | 10 | 20 | 1 |

| 2 | 30 | 30 | 1 |

| 3 | 50 | 50 | 1 |

| 4 | 40 | 40 | 2 |

+----------+------------+--------------+----------+

```
4 rows in set (0.00 sec)

mysql> SET @rank := 0;
Query OK, 0 rows affected (0.01 sec)

mysql> SET @prev_score := NULL;
Query OK, 0 rows affected (0.00 sec)

mysql> UPDATE leaderboard
    ->     JOIN (
    ->     SELECT player_id, total_score,
    ->         @rank := IF(@prev_score = total_score, @rank + 1, @rank
+ 1) AS position,
    ->         @prev_score := total_score
    ->         FROM leaderboard
    ->         ORDER BY total_score DESC
    ->     ) AS ranked
    ->     ON leaderboard.player_id = ranked.player_id
    ->     SET leaderboard.position = ranked.position;
Query OK, 2 rows affected, 2 warnings (0.01 sec)
Rows matched: 4  Changed: 2  Warnings: 2

mysql> select * from leaderboard;
```

```
+-----------+------------+--------------+----------+
| player_id | total_score | highest_score | position |
+-----------+------------+--------------+----------+
|         1 |         10 |           20 |        4 |
|         2 |         30 |           30 |        3 |
|         3 |         50 |           50 |        1 |
|         4 |         40 |           40 |        2 |
+-----------+------------+--------------+----------+
4 rows in set (0.00 sec)

mysql> SELECT * FROM player
    ->  WHERE dob BETWEEN '1999-01-01' AND '2001-12-31';
+----+-------+------------+
| id | name  | dob        |
+----+-------+------------+
|  3 | tanvi | 2001-12-30 |
+----+-------+------------+
1 row in set (0.00 sec)

mysql>  Select * from player where name like '%a%';
+----+---------+------------+
| id | name    | dob        |
+----+---------+------------+
```

```
|  1 | Mitali  | 2005-06-13 |
|  2 | Arpita  | 2005-11-11 |
|  3 | tanvi   | 2001-12-30 |
|  4 | Kalyani | 1993-10-03 |
+----+---------+------------+
4 rows in set (0.01 sec)

mysql> SELECT * FROM player
    ->  WHERE YEAR(dob) = 2001;
+----+-------+------------+
| id | name  | dob        |
+----+-------+------------+
|  3 | tanvi | 2001-12-30 |
+----+-------+------------+
1 row in set (0.00 sec)

mysql>  SELECT * FROM questions
    ->  ORDER BY difficulty_level;
+----+-------------------------------------------------------------------------
-------------------------+---------------------------------------------+---------
-----------------------------------------+-------------------------------------
------+---------------------------------------+---------------+--------------
---+
```

| id | question_text
| option_1

| option_2 | option_3
| option_4 | correct_option | difficulty_level |

+----+--+--+--+--+---------------+------------------+

| 21 | What is a clustered index in SQL?
| An index on non-primary columns

| An index that alters the physical order of the table | A secondary index | A unique index | 2 | difficult |

| 22 | Which SQL clause is used to combine rows from two or more tables, based on a related column between them? | JOIN

| CONNECT | BIND
| LINK | 1 | difficult |

| 23 | How can you optimize a query that is running slow?
| Use a SELECT * statement

| Create indexes on frequently queried columns | Remove indexes | Add more columns |
2 | difficult |

| 24 | What is the function of the WHERE clause in an UPDATE statement? | To specify columns to be updated

| To filter rows to be updated | To group rows to be updated | To order rows to be updated | 2 | difficult |

| 25 | How does a LEFT JOIN differ from an INNER JOIN? | Returns all records

| Returns all records from the left table | Returns all records from the right table | Does not return unmatched rows | 2 | difficult |

| 26 | Which of the following SQL concepts enforces rules to maintain data accuracy? | Normalization

| Integrity Constraints | Indexing | Storage Procedures | 2 | difficult |

| 27 | How does the GROUP BY clause differ from the ORDER BY clause? | GROUP BY groups data; ORDER BY sorts data | GROUP BY filters data; ORDER BY groups data | GROUP BY sorts data; ORDER BY filters data | GROUP BY and ORDER BY are the same | 1 | difficult |

| 28 | In SQL, what does the term "ACID" stand for in database transactions? | Atomicity, Consistency, Isolation, Durability | Access, Control, Integrity, Distribution | Automation, Consistency, Isolation, Durability | Atomicity, Control, Isolation, Distribution | 1 | difficult |

| 29 | What SQL keyword is used to perform a subquery? | INSERT

| GROUP BY | SELECT | DISTINCT | 3 | difficult |

| 1 | What is the primary key in a relational database? | Unique identifier

| Foreign key | Null value
| Index | 1 | easy |

| 3 | What is the purpose of an index in a database?
| Increase storage

| Speed up retrieval | Ensure uniqueness
| None of the above | 2 | easy |

| 4 | Which command is used to remove a table from a database?
| DROP

| DELETE | TRUNCATE
| REMOVE | 1 | easy |

| 6 | Which keyword is used to retrieve unique values?
| DISTINCT

| UNIQUE | SELECT
| WHERE | 1 | easy |

| 7 | Which SQL function is used to calculate the number of records?
| AVG()

| SUM() | COUNT()
| TOTAL() | 3 | easy |

| 8 | What is a foreign key used for?
| Indexing

| Referential integrity | Normalization
| Data redundancy | 2 | easy |

| 10 | What type of join returns only matching records from both tables?
| Left Join

| Inner Join | Right Join
| Cross Join | 2 | easy |

| 12 | Which SQL constraint ensures that all values in a column are unique? | PRIMARY KEY | UNIQUE | CHECK | NOT NULL | 2 | easy |

| 15 | How do you create a view in SQL? | CREATE INDEX | CREATE VIEW | CREATE TABLE | CREATE FUNCTION | 2 | easy |

| 16 | Which command is used to remove duplicate records in a result set? | UNIQUE | DISTINCT | GROUP BY | WHERE | 2 | easy |

| 18 | What is the default sorting order in SQL? | Ascending | Descending | Alphabetical | None | 1 | easy |

| 2 | Which SQL clause is used to filter records? | ORDER BY | GROUP BY | HAVING | WHERE | 4 | medium |

| 5 | What does SQL stand for? | Structured Query Language | Simple Query Language | Sequential Query Language | Structured Queue Language | 1 | medium |

| 9 | Which SQL command is used to add data to a table?
| INSERT

| UPDATE | ADD
| MODIFY | 1 | medium |

| 11 | How can you rename a column in SQL?
| RENAME TO

| ALTER COLUMN | CHANGE COLUMN
| ALTER TABLE | 4 | medium |

| 13 | Which of these SQL clauses is used for conditional aggregation?
| HAVING

| GROUP BY | WHERE
| ORDER BY | 1 | medium |

| 14 | Which SQL data type is used to store large amounts of text?
| TEXT

| CHAR | VARCHAR
| STRING |

1 | medium |

| 17 | What is a stored procedure?
| A table in the database

| A saved SQL query | A trigger
| An index | 2 | medium |

| 19 | What does the BETWEEN operator do in SQL?
| Finds exact values

| Selects within a range | Excludes certain values
| Joins two tables | 2 | medium |

```
| 20 | Which function is used to get the current date in SQL?
| GETDATE()
    | TODAY()                                    | SYSDATE()
| CURRENT_DATE()                          |             4 | medium          |
+----+--------------------------------------------------------------------------------------------------------+---------------------------------------------+---------------------------------------------------------------------+-----------------------------------------------------+-------------------------------------------+----------------+------------------+
29 rows in set (0.00 sec)
```

```
mysql> SELECT YEAR(dob) AS birth_year, COUNT(*) AS player_count
    ->  FROM player
    ->  GROUP BY birth_year;
+------------+--------------+
| birth_year | player_count |
+------------+--------------+
|       2005 |            2 |
|       2001 |            1 |
|       1993 |            1 |
+------------+--------------+
3 rows in set (0.00 sec)
```

```
mysql> Select player.name as player_name,dob,score.total_score,score.correct_answers,score.wrong_answers,score.attempted_questions
    ->      from player
    ->      INNER JOIN score on player.id=score.player_id;
+-------------+------------+-------------+-----------------+--------------+--------------------+
| player_name | dob        | total_score | correct_answers | wrong_answers | attempted_questions |
+-------------+------------+-------------+-----------------+--------------+--------------------+
| Mitali      | 2005-06-13 |          10 |               1 |            2 |                  3 |
| Arpita      | 2005-11-11 |          30 |               3 |            1 |                  4 |
| tanvi       | 2001-12-30 |          50 |               5 |            2 |                  7 |
| Kalyani     | 1993-10-03 |          40 |               4 |            0 |                  4 |
+-------------+------------+-------------+-----------------+--------------+--------------------+
4 rows in set (0.00 sec)

mysql> SELECT player.name,
    ->          score.total_score,
    ->          score.correct_answers,
    ->          leaderboard.position
    ->      FROM player
```

```
    ->      LEFT JOIN score ON player.id = score.player_id
    ->     LEFT JOIN leaderboard ON player.id = leaderboard.player_id;
+---------+------------+----------------+----------+
| name    | total_score | correct_answers | position |
+---------+------------+----------------+----------+
| Mitali  |          10 |              1 |        4 |
| Arpita  |          30 |              3 |        3 |
| tanvi   |          50 |              5 |        1 |
| Kalyani |          40 |              4 |        2 |
+---------+------------+----------------+----------+
4 rows in set (0.00 sec)

mysql> SELECT player.name, score.total_score
    ->    FROM player
    ->    RIGHT JOIN score ON player.id = score.player_id;
+---------+------------+
| name    | total_score |
+---------+------------+
| Mitali  |          10 |
| Arpita  |          30 |
| tanvi   |          50 |
| Kalyani |          40 |
+---------+------------+
```

4 rows in set (0.00 sec)

```
mysql> SELECT name FROM player
    ->      WHERE id IN (SELECT player_id FROM score WHERE
total_score > (SELECT AVG(total_score) FROM score));
+---------+
| name    |
+---------+
| tanvi   |
| Kalyani |
+---------+
2 rows in set (0.01 sec)

mysql> CREATE VIEW leaderboard_view AS
    ->      SELECT player.name, leaderboard.position,
leaderboard.total_score
    ->      FROM player
    ->      JOIN leaderboard ON player.id = leaderboard.player_id;
Query OK, 0 rows affected (0.01 sec)

mysql> SET profiling = 1;
Query OK, 0 rows affected, 1 warning (0.00 sec)

mysql> SELECT * FROM score WHERE total_score = 10;
```

```
+-----------+------------+--------------+----------------+--------------+-------------------+
| player_id | total_score | highest_score | correct_answers | wrong_answers | attempted_questions |
+-----------+------------+--------------+----------------+--------------+-------------------+
|         1 |          10 |           20 |              1 |            2 |                 3 |
+-----------+------------+--------------+----------------+--------------+-------------------+
1 row in set (0.00 sec)

mysql> show profile;
+------------------------------+----------+
| Status                       | Duration |
+------------------------------+----------+
| starting                     | 0.000099 |
| Executing hook on transaction  | 0.000002 |
| starting                     | 0.000007 |
| checking permissions           | 0.000004 |
| Opening tables                 | 0.000046 |
| init                         | 0.000003 |
| System lock                    | 0.000007 |
| optimizing                     | 0.000008 |
| statistics                   | 0.000017 |
```

```
| preparing                    | 0.000020 |
| executing                    | 0.000038 |
| end                          | 0.000003 |
| query end                    | 0.000002 |
| waiting for handler commit   | 0.000008 |
| closing tables               | 0.000005 |
| freeing items                | 0.000054 |
| cleaning up                  | 0.000019 |
+------------------------------+----------+
17 rows in set, 1 warning (0.01 sec)

mysql>   CREATE INDEX idx_total_score ON score(total_score);
Query OK, 0 rows affected (0.05 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> SELECT * FROM score WHERE total_score = 10;
+-----------+-------------+---------------+-----------------+---------------+-------
---------------+
| player_id | total_score | highest_score | correct_answers |
wrong_answers | attempted_questions |
+-----------+-------------+---------------+-----------------+---------------+-------
---------------+
|         1 |          10 |           20 |               1 |             2 |                  3 |
```

```
+----------+------------+--------------+----------------+--------------+------
--------------+
1 row in set (0.00 sec)

mysql> show profile;
+------------------------------+----------+
| Status                       | Duration |
+------------------------------+----------+
| starting                     | 0.000124 |
| Executing hook on transaction  | 0.000003 |
| starting                     | 0.000007 |
| checking permissions         | 0.000005 |
| Opening tables               | 0.000063 |
| init                         | 0.000004 |
| System lock                  | 0.000008 |
| optimizing                   | 0.000029 |
| statistics                   | 0.000086 |
| preparing                    | 0.000034 |
| executing                    | 0.000038 |
| end                          | 0.000003 |
| query end                    | 0.000002 |
| waiting for handler commit     | 0.000009 |
| closing tables               | 0.000007 |
```

```
| freeing items                  | 0.000059 |
| cleaning up                    | 0.000010 |
+--------------------------------+----------+
17 rows in set, 1 warning (0.00 sec)

mysql> CALL ViewPlayerProfile(1);
+--------+------------+-------------+----------------+--------------+-------------------+
| name   | dob        | total_score | correct_answers | wrong_answers | attempted_questions |
+--------+------------+-------------+----------------+--------------+-------------------+
| Mitali | 2005-06-13 |          10 |              1 |            2 |                 3 |
+--------+------------+-------------+----------------+--------------+-------------------+
1 row in set (0.01 sec)

Query OK, 0 rows affected (0.01 sec)

mysql>  SELECT AVG(total_score) AS average_score FROM score;
+---------------+
| average_score |
+---------------+
|       32.5000 |
```

```
+--------------+
1 row in set (0.00 sec)

mysql>  SELECT MAX(total_score) AS highest_score FROM score;
+--------------+
| highest_score |
+--------------+
|           50 |
+--------------+
1 row in set (0.00 sec)

mysql> SELECT player.name, score.total_score
    ->     FROM player
    ->     JOIN score ON player.id = score.player_id
    ->     WHERE score.total_score BETWEEN 20 AND 50;
+---------+-------------+
| name    | total_score |
+---------+-------------+
| Arpita  |          30 |
| Kalyani |          40 |
| tanvi   |          50 |
+---------+-------------+
3 rows in set (0.00 sec)
```

```
mysql> SELECT
    ->         CASE
    ->             WHEN total_score < 20 THEN 'Low'
    ->             WHEN total_score BETWEEN 20 AND 50 THEN
'Medium'
    ->             ELSE 'High'
    ->         END AS score_level, COUNT(*)
    ->      FROM score
    ->      GROUP BY score_level;
+-------------+----------+
| score_level | COUNT(*) |
+-------------+----------+
| Low         |        1 |
| Medium      |        3 |
+-------------+----------+
2 rows in set (0.00 sec)

mysql>
```

JDBC CODE:

```
package gui;



import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
```

```
import java.awt.event.ActionListener;
import java.sql.*;
import java.text.ParseException;
import java.text.SimpleDateFormat;



public class dbmsproject{

 static final String JDBC_DRIVER = "com.mysql.cj.jdbc.Driver";
 static final String DB_URL = "jdbc:mysql://localhost:3306/miniproject"; //
Replace with your DB name
 static final String USER = "root";
 static final String PASS = "Mitali@123";

private Connection conn;
public dbmsproject() {

 try {
Class.forName(JDBC_DRIVER); // Load JDBC driver
conn = DriverManager.getConnection(DB_URL, USER, PASS);
createLoginWindow();
} catch (SQLException | ClassNotFoundException e) {



JOptionPane.showMessageDialog(null, "Failed to connect to DB: " +
e.getMessage());
 }
 }

 private void createLoginWindow() {
JFrame frame = new JFrame("Game Login");
 frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
 frame.setSize(300, 200);
frame.setLayout(new GridLayout(3, 2));
JLabel userLabel = new JLabel("Username:");
JTextField userField = new JTextField();
JLabel dobLabel = new JLabel("DOB (YYYY-MM-DD):");
JTextField dobField = new JTextField();
JButton loginButton = new JButton("Login");
JButton signUpButton = new JButton("Sign Up");
```

```
 loginButton.addActionListener(e -> login(userField.getText(),
dobField.getText()));
signUpButton.addActionListener(e -> signUp(userField.getText(),
dobField.getText()));

 frame.add(userLabel);
frame.add(userField);
frame.add(dobLabel);
frame.add(dobField);
frame.add(loginButton);
frame.add(signUpButton);
frame.setVisible(true);
}

private boolean isValidDate(String dateStr) {
 SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
sdf.setLenient(false);

try {
sdf.parse(dateStr);
return true;
} catch (ParseException e) {

return false;
 }
}

 private void login(String username, String dob) {

if (username.isEmpty() || dob.isEmpty()) {
JOptionPane.showMessageDialog(null, "Both fields are required.");
return;
}

if (!isValidDate(dob)) {
JOptionPane.showMessageDialog(null, "Invalid date format. Please use YYYY-MM-
DD.");
return;
}

// SQL query to check if the username and DOB match in the player table
String sql = "SELECT * FROM player WHERE name = ? AND dob = ?";
```

```
    try (PreparedStatement pstmt = conn.prepareStatement(sql)) {
    pstmt.setString(1, username);
    pstmt.setDate(2, Date.valueOf(dob));
    ResultSet rs = pstmt.executeQuery();

    if (rs.next()) {
    // User found, login successful
    JOptionPane.showMessageDialog(null, "Login successful!");
    createGameMenu(username); // Proceed to game menu after login
    } else {
    // User not found or incorrect credentials
    JOptionPane.showMessageDialog(null, "Invalid username or date of birth.");

    }
    } catch (SQLException e) {
    JOptionPane.showMessageDialog(null, "Error: " + e.getMessage());

    }   }


private void signUp(String username, String dob) {
    if (username.isEmpty() || dob.isEmpty()) {
    JOptionPane.showMessageDialog(null, "All fields are required.");
    return;
    }

    if (!isValidDate(dob)) {
    JOptionPane.showMessageDialog(null, "Invalid date format. Please use YYYY-MM-
DD.");
    return;
    }

    // Check if player already exists
    String checkPlayerSql = "SELECT * FROM player WHERE name = ?";

    try (PreparedStatement pstmt = conn.prepareStatement(checkPlayerSql)) {
    pstmt.setString(1, username);
    ResultSet rs = pstmt.executeQuery();

        if (rs.next()) {
    // Player already exists
```

```
    JOptionPane.showMessageDialog(null, "Username already exists. Please try a
different one.");
    return;
    }
    } catch (SQLException e) {
    JOptionPane.showMessageDialog(null, "Error: " + e.getMessage());
    return;
    }

    // Insert new player
    String insertPlayerSql = "INSERT INTO player (name, dob) VALUES (?, ?)";
    String insertScoreSql = "INSERT INTO score (player_id, total_score,
correct_answers, wrong_answers, attempted_questions, highest_score) VALUES (?, 0,
0, 0, 0, 0)";

    try {

    // Insert player data
    try (PreparedStatement pstmt = conn.prepareStatement(insertPlayerSql,
Statement.RETURN_GENERATED_KEYS)) {
    pstmt.setString(1, username);
    pstmt.setDate(2, Date.valueOf(dob));
    pstmt.executeUpdate();

    // Get the generated player ID
    ResultSet rs = pstmt.getGeneratedKeys();
    if (rs.next()) {
    int playerId = rs.getInt(1);
    // Insert score data for the player

    try (PreparedStatement scorePstmt = conn.prepareStatement(insertScoreSql)) {
scorePstmt.setInt(1, playerId);
scorePstmt.executeUpdate();

    }}}

JOptionPane.showMessageDialog(null, "Sign up successful!");
} catch (SQLException e) {
JOptionPane.showMessageDialog(null, "Error: " + e.getMessage());

    }}
```

```
 private void createGameMenu(String username) {
 JFrame frame = new JFrame("Game Menu");
frame.setSize(300, 300);
frame.setLayout(new GridLayout(4, 1));

 JButton playQuizButton = new JButton("Play Quiz");
 JButton viewProfileButton = new JButton("View Profile");
JButton leaderboardButton = new JButton("Leaderboard");
JButton logoutButton = new JButton("Logout");



playQuizButton.addActionListener(e -> playQuiz(username));
viewProfileButton.addActionListener(e -> viewProfile(username));
leaderboardButton.addActionListener(e -> viewLeaderboard());
logoutButton.addActionListener(e -> frame.dispose());

frame.add(playQuizButton);
frame.add(viewProfileButton);
frame.add(leaderboardButton);
frame.add(logoutButton);
frame.setVisible(true);
}



 private void playQuiz(String username) {
String sql = "SELECT id, question_text, option_1, option_2, option_3, option_4,
correct_option FROM questions ORDER BY RAND() LIMIT 1";

try (PreparedStatement pstmt = conn.prepareStatement(sql)) {

 ResultSet rs = pstmt.executeQuery();

 if (rs.next()) {

String question = rs.getString("question_text");

String option1 = rs.getString("option_1");

String option2 = rs.getString("option_2");
```

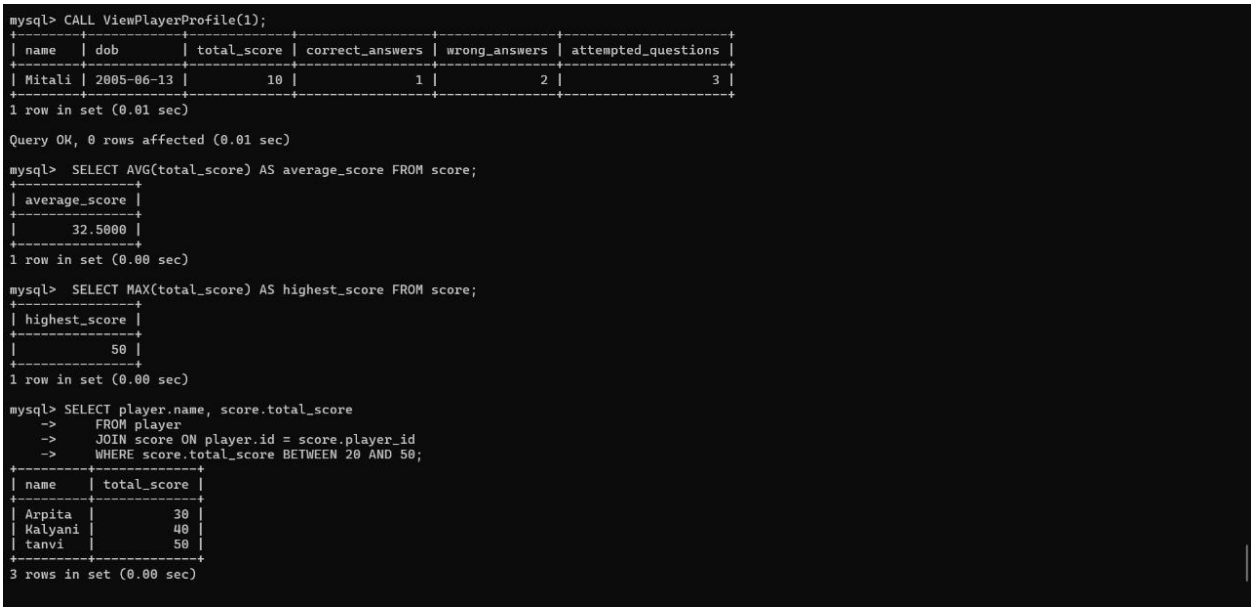
```
String option3 = rs.getString("option_3");

String option4 = rs.getString("option_4");

 int correctOption = rs.getInt("correct_option");

int questionId = rs.getInt("id");
String userAnswer = JOptionPane.showInputDialog(
question + "\n1. " + option1 + "\n2. " + option2 + "\n3. " + option3 + "\n4. " +
option4 + "\nEnter your option (1-4):"
);



if (userAnswer != null && Integer.parseInt(userAnswer) == correctOption) {
JOptionPane.showMessageDialog(null, "Correct Answer!");
updateScore(username, questionId, 10, true); // Correct answer logic
 } else {
 JOptionPane.showMessageDialog(null, "Wrong Answer!");
 updateScore(username, questionId, 0, false); // Wrong answer logic
 }
 } else {
JOptionPane.showMessageDialog(null, "No questions available.");
}
} catch (SQLException | NumberFormatException e) {




 JOptionPane.showMessageDialog(null, "Quiz Error: " + e.getMessage());
}}

 private void updateScore(String username, int questionId, int points, boolean
isCorrect) {

// Step 1: Calculate the total score

    String sql = "UPDATE score s JOIN player p ON s.player_id = p.id " +

    "SET s.total_score = s.total_score + ?, " +

    "s.correct_answers = s.correct_answers + ?, " +
```

```
    "s.wrong_answers = s.wrong_answers + ?, " +

    "s.attempted_questions = s.attempted_questions + 1, " +

    "s.highest_score = CASE " +

    "WHEN s.total_score + ? > s.highest_score THEN s.total_score + ? " +

    "ELSE s.highest_score END " + // Update highest score only if new total score
exceeds previous highest

    "WHERE p.name = ?";



try (PreparedStatement pstmt = conn.prepareStatement(sql)) {

pstmt.setInt(1, points); // Add the points earned in the current session to the
total score
pstmt.setInt(2, isCorrect ? 1 : 0); // Increment correct answers if the answer
was correct
pstmt.setInt(3, isCorrect ? 0 : 1); // Increment wrong answers if the answer was
incorrect
pstmt.setInt(4, points); // The points to compare with the current total score
pstmt.setInt(5, points); // Same value for comparison and update if higher
pstmt.setString(6, username); // Identify the player by username
pstmt.executeUpdate();

} catch (SQLException e) {
JOptionPane.showMessageDialog(null, "Error updating score: " + e.getMessage());
}
}



private void viewProfile(String username) {
String sql = "SELECT s.total_score, s.correct_answers, s.wrong_answers,
s.attempted_questions, s.highest_score "
+ "FROM player p JOIN score s ON p.id = s.player_id WHERE p.name = ?";

 try (PreparedStatement pstmt = conn.prepareStatement(sql)) {
```

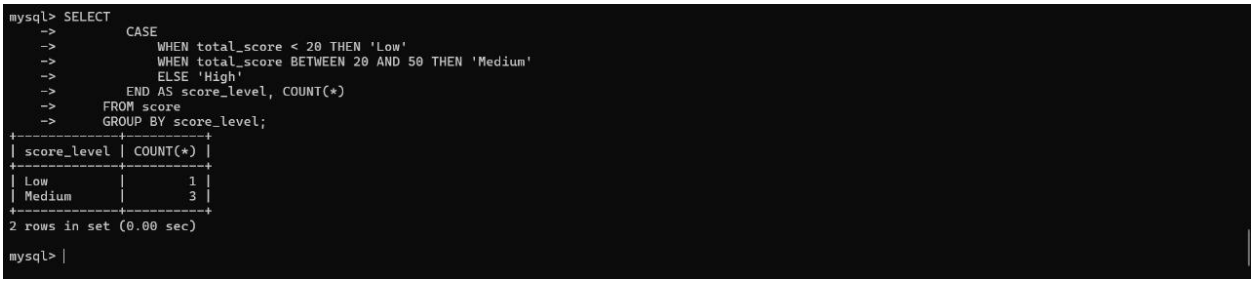
```
pstmt.setString(1, username);
ResultSet rs = pstmt.executeQuery();

 if (rs.next()) {
 int totalScore = rs.getInt("total_score");
 int correctAnswers = rs.getInt("correct_answers");
 int wrongAnswers = rs.getInt("wrong_answers");
 int attemptedQuestions = rs.getInt("attempted_questions");
 int highestScore = rs.getInt("highest_score");
 JOptionPane.showMessageDialog(null,
 "Player Profile:\n" +
"Total Score: " + totalScore + "\n" +
 "Correct Answers: " + correctAnswers + "\n" +
 "Wrong Answers: " + wrongAnswers + "\n" +
 "Attempted Questions: " + attemptedQuestions + "\n" +
 "Highest Score: " + highestScore);
 } else {
 JOptionPane.showMessageDialog(null, "Profile not found.");
 }
 } catch (SQLException e) {
JOptionPane.showMessageDialog(null, "Error: " + e.getMessage());
 }
 }

private void viewLeaderboard() {
String sql = "SELECT p.name, s.total_score FROM player p JOIN score s ON p.id =
s.player_id ORDER BY s.total_score DESC";
try (PreparedStatement pstmt = conn.prepareStatement(sql); ResultSet rs =
pstmt.executeQuery()) {
 StringBuilder leaderboard = new StringBuilder("Leaderboard:\n");

 int rank = 1;

while (rs.next()) {
 leaderboard.append("Rank ").append(rank++)
 .append(": ").append(rs.getString("name"))
.append(" - ").append(rs.getInt("total_score")).append("\n");

}
JOptionPane.showMessageDialog(null, leaderboard.toString());
} catch (SQLException e) {
JOptionPane.showMessageDialog(null, "Leaderboard Error: " + e.getMessage());
```

```
}
 }

 public static void main(String[] args) {

 SwingUtilities.invokeLater(dbmsproject::new);
 }
}
```

I/P & O/P Screenshots:



```
mysql> CALL ViewPlayerProfile(1);
+--------+------------+-------------+-----------------+---------------+----------------------+
| name   | dob        | total_score | correct_answers | wrong_answers | attempted_questions |
+--------+------------+-------------+-----------------+---------------+----------------------+
| Mitali | 2005-06-13 |          10 |               1 |             2 |                    3 |
+--------+------------+-------------+-----------------+---------------+----------------------+
1 row in set (0.01 sec)

Query OK, 0 rows affected (0.01 sec)

mysql>  SELECT AVG(total_score) AS average_score FROM score;
+---------------+
| average_score |
+---------------+
|       32.5000 |
+---------------+
1 row in set (0.00 sec)

mysql>  SELECT MAX(total_score) AS highest_score FROM score;
+---------------+
| highest_score |
+---------------+
|            50 |
+---------------+
1 row in set (0.00 sec)

mysql> SELECT player.name, score.total_score
    ->     FROM player
    ->     JOIN score ON player.id = score.player_id
    ->     WHERE score.total_score BETWEEN 20 AND 50;
+---------+-------------+
| name    | total_score |
+---------+-------------+
| Arpita  |          30 |
| Kalyani |          40 |
| tanvi   |          50 |
+---------+-------------+
3 rows in set (0.00 sec)
```



```
mysql> SELECT
    ->         CASE
    ->             WHEN total_score < 20 THEN 'Low'
    ->             WHEN total_score BETWEEN 20 AND 50 THEN 'Medium'
    ->             ELSE 'High'
    ->         END AS score_level, COUNT(*)
    ->      FROM score
    ->      GROUP BY score_level;
+-------------+----------+
| score_level | COUNT(*) |
+-------------+----------+
| Low         |        1 |
| Medium      |        3 |
+-------------+----------+
2 rows in set (0.00 sec)

mysql>
```

```
     | Selects within a range                          | Excludes certain values                      | Joins two tables                          |
              2 | medium          |
| 20 | Which function is used to get the current date in SQL?                                                  |    GETDATE()
     | TODAY()                                         | SYSDATE()                                    | CURRENT_DATE()                            |
              4 | medium          |
+----+-------------------------------------------------------------------------------------------------------+-----------------------------------------
-----+-----------------------------------------------+----------------------------------------------+-------------------------------------------+
-----------------+------------------+
29 rows in set (0.00 sec)

mysql> SELECT YEAR(dob) AS birth_year, COUNT(*) AS player_count
    ->  FROM player
    ->  GROUP BY birth_year;
+------------+--------------+
| birth_year | player_count |
+------------+--------------+
|       2005 |            2 |
|       2001 |            1 |
|       1993 |            1 |
+------------+--------------+
3 rows in set (0.00 sec)

mysql> Select player.name as player_name,dob,score.total_score,score.correct_answers,score.wrong_answers,score.attempted_questions
    ->      from player
    ->      INNER JOIN score on player.id=score.player_id;
+-------------+------------+-------------+-----------------+---------------+----------------------+
| player_name | dob        | total_score | correct_answers | wrong_answers | attempted_questions |
+-------------+------------+-------------+-----------------+---------------+----------------------+
| Mitali      | 2005-06-13 |          10 |               1 |             2 |                    3 |
| Arpita      | 2005-11-11 |          30 |               3 |             1 |                    4 |
| tanvi       | 2001-12-30 |          50 |               5 |             2 |                    7 |
| Kalyani     | 1993-10-03 |          40 |               4 |             0 |                    4 |
+-------------+------------+-------------+-----------------+---------------+----------------------+
4 rows in set (0.00 sec)
```

```
mysql> SET @rank := 0;
Query OK, 0 rows affected (0.01 sec)

mysql> SET @prev_score := NULL;
Query OK, 0 rows affected (0.00 sec)

mysql> UPDATE leaderboard
    ->      JOIN (
    ->      SELECT player_id, total_score,
    ->            @rank := IF(@prev_score = total_score, @rank + 1, @rank + 1) AS position,
    ->            @prev_score := total_score
    ->            FROM leaderboard
    ->            ORDER BY total_score DESC
    ->      ) AS ranked
    ->      ON leaderboard.player_id = ranked.player_id
    ->      SET leaderboard.position = ranked.position;
Query OK, 2 rows affected, 2 warnings (0.01 sec)
Rows matched: 4  Changed: 2  Warnings: 2

mysql> select * from leaderboard;
+-----------+-------------+---------------+----------+
| player_id | total_score | highest_score | position |
+-----------+-------------+---------------+----------+
|         1 |          10 |            20 |        4 |
|         2 |          30 |            30 |        3 |
|         3 |          50 |            50 |        1 |
|         4 |          40 |            40 |        2 |
+-----------+-------------+---------------+----------+
4 rows in set (0.00 sec)

mysql> SELECT * FROM player
    ->  WHERE dob BETWEEN '1999-01-01' AND '2001-12-31';
+----+-------+------------+
| id | name  | dob        |
+----+-------+------------+
|  3 | tanvi | 2001-12-30 |
+----+-------+------------+
1 row in set (0.00 sec)
```

```
mysql> select * from player;
+----+---------+------------+
| id | name    | dob        |
+----+---------+------------+
|  1 | Mitali  | 2005-06-13 |
|  2 | Arpita  | 2005-11-11 |
|  3 | tanvi   | 2001-12-30 |
|  4 | Kalyani | 1993-10-03 |
+----+---------+------------+
4 rows in set (0.00 sec)

mysql> select * from score;
+-----------+-------------+---------------+-----------------+---------------+----------------------+
| player_id | total_score | highest_score | correct_answers | wrong_answers | attempted_questions |
+-----------+-------------+---------------+-----------------+---------------+----------------------+
|         1 |          10 |            20 |               1 |             2 |                    3 |
|         2 |          30 |            30 |               3 |             1 |                    4 |
|         3 |          50 |            50 |               5 |             2 |                    7 |
|         4 |          40 |            40 |               4 |             0 |                    4 |
+-----------+-------------+---------------+-----------------+---------------+----------------------+
4 rows in set (0.00 sec)

mysql> select * from leaderboard;
+-----------+-------------+---------------+----------+
| player_id | total_score | highest_score | position |
+-----------+-------------+---------------+----------+
|         1 |          10 |            20 |        1 |
|         2 |          30 |            30 |        1 |
|         3 |          50 |            50 |        1 |
|         4 |          40 |            40 |        2 |
+-----------+-------------+---------------+----------+
4 rows in set (0.00 sec)
```

```
mysql>   CREATE INDEX idx_total_score ON score(total_score);
Query OK, 0 rows affected (0.05 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> SELECT * FROM score WHERE total_score = 10;
+-----------+-------------+---------------+-----------------+---------------+----------------------+
| player_id | total_score | highest_score | correct_answers | wrong_answers | attempted_questions |
+-----------+-------------+---------------+-----------------+---------------+----------------------+
|         1 |          10 |            20 |               1 |             2 |                    3 |
+-----------+-------------+---------------+-----------------+---------------+----------------------+
1 row in set (0.00 sec)

mysql> show profile;
+------------------------------+----------+
| Status                       | Duration |
+------------------------------+----------+
| starting                     | 0.000124 |
| Executing hook on transaction  | 0.000003 |
| starting                     | 0.000007 |
| checking permissions         | 0.000005 |
| Opening tables               | 0.000063 |
| init                         | 0.000004 |
| System lock                  | 0.000008 |
| optimizing                   | 0.000029 |
| statistics                   | 0.000086 |
| preparing                    | 0.000034 |
| executing                    | 0.000038 |
| end                          | 0.000003 |
| query end                    | 0.000002 |
| waiting for handler commit   | 0.000009 |
| closing tables               | 0.000007 |
| freeing items                | 0.000059 |
| cleaning up                  | 0.000010 |
+------------------------------+----------+
17 rows in set, 1 warning (0.00 sec)
```

```
           2 | easy            |
| 12 | Which SQL constraint ensures that all values in a column are unique?                                    | PRIMARY KEY
    | UNIQUE                                          | CHECK                                  | NOT NULL                                |
           2 | easy            |
| 15 | How do you create a view in SQL?                                                                        | CREATE INDEX
    | CREATE VIEW                                     | CREATE TABLE                           | CREATE FUNCTION                         |
           2 | easy            |
| 16 | Which command is used to remove duplicate records in a result set?                                      | UNIQUE
    | DISTINCT                                        | GROUP BY                               | WHERE                                   |
           2 | easy            |
| 18 | What is the default sorting order in SQL?                                                               | Ascending
    | Descending                                      | Alphabetical                           | None                                    |
           1 | easy            |
|  2 | Which SQL clause is used to filter records?                                                             | ORDER BY
    | GROUP BY                                        | HAVING                                 | WHERE                                   |
           4 | medium          |
|  5 | What does SQL stand for?                                                                                | Structured Query Language
    | Simple Query Language                           | Sequential Query Language              | Structured Queue Language               |
           1 | medium          |
|  9 | Which SQL command is used to add data to a table?                                                       | INSERT
    | UPDATE                                          | ADD                                    | MODIFY                                  |
           1 | medium          |
| 11 | How can you rename a column in SQL?                                                                     | RENAME TO
    | ALTER COLUMN                                    | CHANGE COLUMN                          | ALTER TABLE                             |
           4 | medium          |
| 13 | Which of these SQL clauses is used for conditional aggregation?                                         | HAVING
    | GROUP BY                                        | WHERE                                  | ORDER BY                                |
           1 | medium          |
| 14 | Which SQL data type is used to store large amounts of text?                                             | TEXT
    | CHAR                                            | VARCHAR                                | STRING                                  |
           1 | medium          |
| 17 | What is a stored procedure?                                                                             | A table in the database
    | A saved SQL query                               | A trigger                              | An index                                |
           2 | medium          |
| 19 | What does the BETWEEN operator do in SQL?                                                               | Finds exact values
    | Selects within a range                          | Excludes certain values                | Joins two tables                        |
           2 | medium          |
| 20 | Which function is used to get the current date in SQL?                                                  | GETDATE()
    | TODAY()                                         | SYSDATE()                              | CURRENT_DATE()                          |
           4 | medium          |
```

```
mysql> SELECT player.name,
    ->          score.total_score,
    ->          score.correct_answers,
    ->          leaderboard.position
    ->      FROM player
    ->      LEFT JOIN score ON player.id = score.player_id
    ->     LEFT JOIN leaderboard ON player.id = leaderboard.player_id;
+---------+-------------+-----------------+----------+
| name    | total_score | correct_answers | position |
+---------+-------------+-----------------+----------+
| Mitali  |          10 |               1 |        4 |
| Arpita  |          30 |               3 |        3 |
| tanvi   |          50 |               5 |        1 |
| Kalyani |          40 |               4 |        2 |
+---------+-------------+-----------------+----------+
4 rows in set (0.00 sec)

mysql> SELECT player.name, score.total_score
    ->    FROM player
    ->    RIGHT JOIN score ON player.id = score.player_id;
+---------+-------------+
| name    | total_score |
+---------+-------------+
| Mitali  |          10 |
| Arpita  |          30 |
| tanvi   |          50 |
| Kalyani |          40 |
+---------+-------------+
4 rows in set (0.00 sec)

mysql> SELECT name FROM player
    ->      WHERE id IN (SELECT player_id FROM score WHERE total_score > (SELECT AVG(total_score) FROM score));
+---------+
| name    |
+---------+
| tanvi   |
| Kalyani |
+---------+
2 rows in set (0.01 sec)
```

```
mysql> CREATE VIEW leaderboard_view AS
    ->      SELECT player.name, leaderboard.position, leaderboard.total_score
    ->      FROM player
    ->      JOIN leaderboard ON player.id = leaderboard.player_id;
Query OK, 0 rows affected (0.01 sec)

mysql> SET profiling = 1;
Query OK, 0 rows affected, 1 warning (0.00 sec)

mysql> SELECT * FROM score WHERE total_score = 10;
+-----------+-------------+---------------+-----------------+---------------+----------------------+
| player_id | total_score | highest_score | correct_answers | wrong_answers | attempted_questions |
+-----------+-------------+---------------+-----------------+---------------+----------------------+
|         1 |          10 |            20 |               1 |             2 |                    3 |
+-----------+-------------+---------------+-----------------+---------------+----------------------+
1 row in set (0.00 sec)

mysql> show profile;
+--------------------------------+----------+
| Status                         | Duration |
+--------------------------------+----------+
| starting                       | 0.000099 |
| Executing hook on transaction  | 0.000002 |
| starting                       | 0.000007 |
| checking permissions           | 0.000004 |
| Opening tables                 | 0.000046 |
| init                           | 0.000003 |
| System lock                    | 0.000007 |
| optimizing                     | 0.000008 |
| statistics                     | 0.000017 |
| preparing                      | 0.000020 |
| executing                      | 0.000038 |
| end                            | 0.000003 |
| query end                      | 0.000002 |
| waiting for handler commit     | 0.000008 |
| closing tables                 | 0.000005 |
| freeing items                  | 0.000054 |
| cleaning up                    | 0.000019 |
+--------------------------------+----------+
17 rows in set, 1 warning (0.01 sec)
```

```
    | Create indexes on frequently queried columns         | Remove indexes                                | Add more columns                                |
            2 | difficult       |
| 24 | What is the function of the WHERE clause in an UPDATE statement?                                                 | To specify columns to be updated
    | To filter rows to be updated                         | To group rows to be updated                   | To order rows to be updated                     |
            2 | difficult       |
| 25 | How does a LEFT JOIN differ from an INNER JOIN?                                                                  | Returns all records
    | Returns all records from the left table              | Returns all records from the right table      | Does not return unmatched rows                  |
            2 | difficult       |
| 26 | Which of the following SQL concepts enforces rules to maintain data accuracy?                                    | Normalization
    | Integrity Constraints                                | Indexing                                      | Storage Procedures                              |
            2 | difficult       |
| 27 | How does the GROUP BY clause differ from the ORDER BY clause?                                                    | GROUP BY groups data; ORDER BY sorts data
    | GROUP BY filters data; ORDER BY groups data          | GROUP BY sorts data; ORDER BY filters data     | GROUP BY and ORDER BY are the same              |
            1 | difficult       |
| 28 | In SQL, what does the term "ACID" stand for in database transactions?                                            | Atomicity, Consistency, Isolation, Durabi
lity | Access, Control, Integrity, Distribution             | Automation, Consistency, Isolation, Durability | Atomicity, Control, Isolation, Distribution |
            1 | difficult       |
| 29 | What SQL keyword is used to perform a subquery?                                                                  | INSERT
    | GROUP BY                                             | SELECT                                        | DISTINCT                                        |
            3 | difficult       |
|  1 | What is the primary key in a relational database?                                                                | Unique identifier
    | Foreign key                                          | Null value                                    | Index                                           |
            1 | easy            |
|  3 | What is the purpose of an index in a database?                                                                   | Increase storage
    | Speed up retrieval                                   | Ensure uniqueness                             | None of the above                               |
            2 | easy            |
|  4 | Which command is used to remove a table from a database?                                                         | DROP
    | DELETE                                               | TRUNCATE                                      | REMOVE                                          |
            1 | easy            |
|  6 | Which keyword is used to retrieve unique values?                                                                 | DISTINCT
    | UNIQUE                                               | SELECT                                        | WHERE                                           |
            1 | easy            |
|  7 | Which SQL function is used to calculate the number of records?                                                   | AVG()
    | SUM()                                                | COUNT()                                       | TOTAL()                                         |
            3 | easy            |
|  8 | What is a foreign key used for?                                                                                  | Indexing
    | Referential integrity                                | Normalization                                 | Data redundancy                                 |
            2 | easy            |
| 10 | What type of join returns only matching records from both tables?                                                | Left Join
    | Inner Join                                           | Right Join                                    | Cross Join                                      |
            2 | easy            |
```

```
mysql>  Select * from player where name like '%a%';
+----+---------+------------+
| id | name    | dob        |
+----+---------+------------+
|  1 | Mitali  | 2005-06-13 |
|  2 | Arpita  | 2005-11-11 |
|  3 | tanvi   | 2001-12-30 |
|  4 | Kalyani | 1993-10-03 |
+----+---------+------------+
4 rows in set (0.01 sec)

mysql> SELECT * FROM player
    ->  WHERE YEAR(dob) = 2001;
+----+-------+------------+
| id | name  | dob        |
+----+-------+------------+
|  3 | tanvi | 2001-12-30 |
+----+-------+------------+
1 row in set (0.00 sec)

mysql>  SELECT * FROM questions
    ->  ORDER BY difficulty_level;
+----+-------------------------------------------------------------------------------------------------------------------+-----------------------------------------------
-----+-------------------------------------------------------+-----------------------------------------------+-----------------------------------------+
----------------+------------------+
| id | question_text                                                                                                     | option_1
    | option_2                                              | option_3                                      | option_4                                |
 correct_option | difficulty_level |
+----+-------------------------------------------------------------------------------------------------------------------+-----------------------------------------------
-----+-------------------------------------------------------+-----------------------------------------------+-----------------------------------------+
----------------+------------------+
| 21 | What is a clustered index in SQL?                                                                                 | An index on non-primary columns
    | An index that alters the physical order of the table | A secondary index                             | A unique index                          |
            2 | difficult       |
| 22 | Which SQL clause is used to combine rows from two or more tables, based on a related column between them? | JOIN
    | CONNECT                                              | BIND                                          | LINK                                    |
            1 | difficult       |
| 23 | How can you optimize a query that is running slow?                                                                | Use a SELECT * statement
    | Create indexes on frequently queried columns         | Remove indexes                                | Add more columns                        |
            2 | difficult       |
```