All SQL scripts and documentation

User Management: Support for two user roles – Client and Freelancer

```
mysql> /*user management*/
mysql> /*create table roles which include roles like Client and Freelancer*/
mysql> CREATE TABLE Roles (
    ->   role_id INT AUTO_INCREMENT PRIMARY KEY,
    ->   role_name VARCHAR(50) UNIQUE NOT NULL
    -> );
Query OK, 0 rows affected (0.05 sec)

mysql>
mysql> /*insert values into roles table*/
mysql> insert into roles(role_name) values ('Client'),('Freelancer');
Query OK, 2 rows affected (0.01 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> /*create table users which contains necessary details of users*/
mysql> CREATE TABLE Users (
    ->   user_id INT AUTO_INCREMENT PRIMARY KEY,
    ->   name VARCHAR(100) NOT NULL,
    ->   email VARCHAR(100) UNIQUE NOT NULL,
    ->   password VARCHAR(100) NOT NULL,
    ->   role_id INT NOT NULL,
    ->   created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    ->   FOREIGN KEY (role_id) REFERENCES Roles(role_id)
    -> );
Query OK, 0 rows affected (0.06 sec)

mysql> /*insert values into users table*/
mysql> INSERT INTO Users (name, email, password, role_id) VALUES
    -> ('Alice', 'alice@devifyx.com', 'alice123', 1),    -- Client
    -> ('Bob', 'bob@devifyx.com', 'bob123', 2),          -- Freelancer
    -> ('Charlie', 'charlie@devifyx.com', 'charlie123', 2); -- Freelancer
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0
```

```
mysql> /*user management query -- to get all users and their roles*/
mysql> SELECT
    ->     u.user_id,
    ->     u.name,
    ->     u.email,
    ->     r.role_name
    -> FROM Users u
    -> JOIN Roles r ON u.role_id = r.role_id;
+---------+---------+---------------------+------------+
| user_id | name    | email               | role_name  |
+---------+---------+---------------------+------------+
|       1 | Alice   | alice@devifyx.com   | Client     |
|       2 | Bob     | bob@devifyx.com     | Freelancer |
|       3 | Charlie | charlie@devifyx.com | Freelancer |
+---------+---------+---------------------+------------+
3 rows in set (0.00 sec)

mysql> /*Additional: Filter users by roles*/
mysql> /* all freelancer*/
mysql> SELECT * FROM Users u
    -> JOIN Roles r ON u.role_id = r.role_id
    -> WHERE r.role_name = 'Freelancer';
+---------+---------+---------------------+------------+---------+---------------------+---------+------------+
| user_id | name    | email               | password   | role_id | created_at          | role_id | role_name  |
+---------+---------+---------------------+------------+---------+---------------------+---------+------------+
|       2 | Bob     | bob@devifyx.com     | bob123     |       2 | 2025-06-21 23:47:49 |       2 | Freelancer |
|       3 | Charlie | charlie@devifyx.com | charlie123 |       2 | 2025-06-21 23:47:49 |       2 | Freelancer |
+---------+---------+---------------------+------------+---------+---------------------+---------+------------+
2 rows in set (0.01 sec)

mysql> /*all clients*/
mysql> SELECT * FROM Users u
    -> JOIN Roles r ON u.role_id = r.role_id
    -> WHERE r.role_name = 'Client';
+---------+-------+-------------------+----------+---------+---------------------+---------+-----------+
| user_id | name  | email             | password | role_id | created_at          | role_id | role_name |
+---------+-------+-------------------+----------+---------+---------------------+---------+-----------+
|       1 | Alice | alice@devifyx.com | alice123 |       1 | 2025-06-21 23:47:49 |       1 | Client    |
+---------+-------+-------------------+----------+---------+---------------------+---------+-----------+
1 row in set (0.00 sec)
```

Project Posting: Clients can post new projects with details such as title, description, budget, deadline, and required skills

```
mysql> /*project posting*/
mysql> /*create project table which stores project information posted by clients*/
mysql> CREATE TABLE Projects (
    ->   project_id INT AUTO_INCREMENT PRIMARY KEY,
    ->   client_id INT NOT NULL,
    ->   title VARCHAR(255),
    ->   description TEXT,
    ->   budget DECIMAL(10,2),
    ->   deadline DATE,
    ->   status ENUM('Open', 'In Progress', 'Completed', 'Cancelled') DEFAULT 'Open',
    ->   created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    ->   FOREIGN KEY (client_id) REFERENCES Users(user_id)
    -> );
Query OK, 0 rows affected (0.05 sec)

mysql> /*create table skills which store the skills required for the project*/
mysql> CREATE TABLE Skills (
    ->     skill_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     skill_name VARCHAR(100) NOT NULL
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> /*create table projectSkills to map each project to required skills(many-to-many relationship)*/
mysql> CREATE TABLE ProjectSkills (
    ->   project_id INT,
    ->   skill_id INT,
    ->   PRIMARY KEY (project_id, skill_id),
    ->   FOREIGN KEY (project_id) REFERENCES Projects(project_id),
    ->   FOREIGN KEY (skill_id) REFERENCES Skills(skill_id)
    -> );
Query OK, 0 rows affected (0.06 sec)

mysql> /*insert values into project table*/
mysql> insert into projects(client_id,title,description,budget,deadline)
    -> values(1,'Build REST API','Need a python + MYSQL backend',1000.00,'2025-06-20'),
    -> (2,'Build Frondend project','Need a React',1100.00,'2025-06-21');
Query OK, 2 rows affected (0.01 sec)
Records: 2  Duplicates: 0  Warnings: 0
```

```
mysql> /*insert values into skills table*/
mysql> insert into skills(skill_name) values ('Python'),('MYSQL'),('React.js');
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> /*insert into projectSkills table*/
mysql> insert into projectSkills(project_id,skill_id) values
    -> (1,1),
    -> (1,2),
    -> (2,3);
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> /*view all posted projects with skills*/
mysql> SELECT
    ->   p.project_id,
    ->   p.title,
    ->   p.description,
    ->   p.budget,
    ->   p.deadline,
    ->   u.name AS client_name,
    ->   GROUP_CONCAT(s.skill_name SEPARATOR ', ') AS required_skills
    -> FROM Projects p
    -> JOIN Users u ON p.client_id = u.user_id
    -> JOIN ProjectSkills ps ON p.project_id = ps.project_id
    -> JOIN Skills s ON ps.skill_id = s.skill_id
    -> GROUP BY p.project_id;
+------------+-----------------------+-------------------------------+---------+------------+-------------+-----------------+
| project_id | title                 | description                   | budget  | deadline   | client_name | required_skills |
+------------+-----------------------+-------------------------------+---------+------------+-------------+-----------------+
|          1 | Build REST API        | Need a python + MYSQL backend | 1000.00 | 2025-06-20 | Alice       | Python, MYSQL   |
|          2 | Build Frondend project | Need a React                  | 1100.00 | 2025-06-21 | Bob         | React.js        |
+------------+-----------------------+-------------------------------+---------+------------+-------------+-----------------+
2 rows in set (0.01 sec)
```

Bidding System: Freelancers can place bids on available projects, specifying their proposed amount and timeline

```
mysql> /*bidding system*/
mysql> /*create table for bids*/
mysql> CREATE TABLE Bids (
    ->   bid_id INT AUTO_INCREMENT PRIMARY KEY,
    ->   project_id INT NOT NULL,
    ->   freelancer_id INT NOT NULL,
    ->   amount DECIMAL(10,2),
    ->   timeline_days INT,
    ->   status ENUM('Pending', 'Accepted', 'Rejected') DEFAULT 'Pending',
    ->   bid_time TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    ->   FOREIGN KEY (project_id) REFERENCES Projects(project_id),
    ->   FOREIGN KEY (freelancer_id) REFERENCES Users(user_id)
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> /* insert data --- freelancer bidding*/
mysql> INSERT INTO Bids (project_id, freelancer_id, amount, timeline_days)
    -> VALUES (1, 2, 950.00, 7);
Query OK, 1 row affected (0.01 sec)

mysql> /*view all bids for a project*/
mysql> SELECT
    ->   b.bid_id,
    ->   u.name AS freelancer_name,
    ->   b.amount,
    ->   b.timeline_days,
    ->   b.status,
    ->   b.bid_time
    -> FROM Bids b
    -> JOIN Users u ON b.freelancer_id = u.user_id
    -> WHERE b.project_id = 1  -- replace with project_id of interest
    -> ORDER BY b.bid_time;
+--------+-----------------+--------+---------------+---------+---------------------+
| bid_id | freelancer_name | amount | timeline_days | status  | bid_time            |
+--------+-----------------+--------+---------------+---------+---------------------+
|      1 | Bob             | 950.00 |             7 | Pending | 2025-06-22 00:23:22 |
+--------+-----------------+--------+---------------+---------+---------------------+
1 row in set (0.00 sec)
```

Bid History: Track all bids placed on each project, including bid status (pending, accepted, rejected).

```
mysql> /*Bid history*/
mysql> SELECT
   ->     p.project_id,
   ->     p.title AS project_title,
   ->     b.bid_id,
   ->     u.user_id AS freelancer_id,
   ->     u.name AS freelancer_name,
   ->     b.amount AS bid_amount,
   ->     b.timeline_days AS proposed_timeline,
   ->     b.status AS bid_status,
   ->     b.bid_time
   -> FROM Bids b
   -> JOIN Projects p ON b.project_id = p.project_id
   -> JOIN Users u ON b.freelancer_id = u.user_id
   -> ORDER BY p.project_id, b.bid_time;
+------------+---------------+--------+--------------+-----------------+------------+-------------------+------------+---------------------+
| project_id | project_title | bid_id | freelancer_id | freelancer_name | bid_amount | proposed_timeline | bid_status | bid_time            |
+------------+---------------+--------+--------------+-----------------+------------+-------------------+------------+---------------------+
|          1 | Build REST API |      1 |            2 | Bob             |     950.00 |                 7 | Pending    | 2025-06-22 00:23:22 |
+------------+---------------+--------+--------------+-----------------+------------+-------------------+------------+---------------------+
1 row in set (0.00 sec)
```

Project Assignment: Clients can accept a bid, assigning the project to a freelancer and updating project status.

```
mysql> /*Project assignment*/
mysql> ALTER TABLE Projects
    -> ADD COLUMN assigned_freelancer_id INT DEFAULT NULL,
    -> ADD FOREIGN KEY (assigned_freelancer_id) REFERENCES Users(user_id);
Query OK, 2 rows affected (0.06 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> DELIMITER $$
mysql>
mysql> CREATE PROCEDURE AssignProjectToFreelancer (
    ->     IN p_bid_id INT
    -> )
    -> BEGIN
    ->     DECLARE v_project_id INT;
    ->     DECLARE v_freelancer_id INT;
    ->
    ->     -- Get project_id and freelancer_id from the accepted bid
    ->     SELECT project_id, freelancer_id
    ->     INTO v_project_id, v_freelancer_id
    ->     FROM Bids
    ->     WHERE bid_id = p_bid_id;
    ->
    ->     -- Step 1: Update selected bid to 'Accepted'
    ->     UPDATE Bids
    ->     SET status = 'Accepted'
    ->     WHERE bid_id = p_bid_id;
    ->
    ->     -- Step 2: Reject other bids on the same project
    ->     UPDATE Bids
    ->     SET status = 'Rejected'
    ->     WHERE project_id = v_project_id AND bid_id != p_bid_id;
    ->
    ->     -- Step 3: Assign freelancer to the project and update status
    ->     UPDATE Projects
    ->     SET assigned_freelancer_id = v_freelancer_id,
    ->         status = 'In Progress'
    ->     WHERE project_id = v_project_id;
    -> END $$
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> DELIMITER ;
mysql> CALL AssignProjectToFreelancer(5);
Query OK, 0 rows affected (0.00 sec)

mysql> select
    -> p.project_id,
    -> p.title,
    -> p.status,
    -> u.name AS assigned_freelancer
    -> from projects p
    -> left join users u on p.assigned_freelancer_id=u.user_id
    -> where p.project_id=1;
+------------+----------------+--------+---------------------+
| project_id | title          | status | assigned_freelancer |
+------------+----------------+--------+---------------------+
|          1 | Build REST API | Open   | NULL                |
+------------+----------------+--------+---------------------+
1 row in set (0.02 sec)

mysql> CALL AssignProjectToFreelancer(1);
Query OK, 1 row affected (0.02 sec)

mysql> SELECT
    ->     p.project_id,
    ->     p.title,
    ->     p.status,
    ->     u.name AS assigned_freelancer
    -> FROM Projects p
    -> LEFT JOIN Users u ON p.assigned_freelancer_id = u.user_id
    -> WHERE p.project_id = 1;  -- Change project_id as needed
+------------+----------------+-------------+---------------------+
| project_id | title          | status      | assigned_freelancer |
+------------+----------------+-------------+---------------------+
|          1 | Build REST API | In Progress | Bob                 |
+------------+----------------+-------------+---------------------+
1 row in set (0.00 sec)

mysql> /* the above query is to see the project with assigned freelancer*/
mysql>
```

Project Status Tracking: Projects should have statuses such as Open, In Progress, Completed, Cancelled.

```
mysql> /*Project status tracking*/
mysql> /*Mark project as Completed*/
mysql> UPDATE Projects
    -> SET status = 'Completed'
    -> WHERE project_id = 1;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> /*Cancel a project*/
mysql> update projects
    -> set status='Cancelled'
    -> where project_id=2;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> /*query project by status*/
mysql> SELECT * FROM Projects
    -> WHERE status IN ('Open', 'In Progress');
Empty set (0.00 sec)

mysql> /* as no project is added as open or in progress that's why it is showing empty set*/
mysql> /*for references if I change the status to open and then run*/
mysql> update projects
    -> set status='open'
    -> where project_id=1;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT * FROM Projects
    -> WHERE status IN ('Open', 'In Progress');
+------------+-----------+---------------+-----------------------------+---------+------------+--------+---------------------+-----------------------+
| project_id | client_id | title         | description                 | budget  | deadline   | status | created_at          | assigned_freelancer_id |
+------------+-----------+---------------+-----------------------------+---------+------------+--------+---------------------+-----------------------+
|          1 |         1 | Build REST API | Need a python + MYSQL backend | 1000.00 | 2025-06-20 | Open   | 2025-06-22 00:14:00 |                     2 |
+------------+-----------+---------------+-----------------------------+---------+------------+--------+---------------------+-----------------------+
1 row in set (0.00 sec)

mysql> /*now it is giving the data for that*/
mysql>
mysql> /*let's change it back*/
mysql> update projects
    -> set status='Completed'
    -> where project_id=1;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

```
mysql> /*Now all completed projects with freelancer Info*/
mysql> SELECT
    ->     p.project_id,
    ->     p.title,
    ->     u.name AS freelancer,
    ->     p.status
    -> FROM Projects p
    -> LEFT JOIN Users u ON p.assigned_freelancer_id = u.user_id
    -> WHERE p.status = 'Completed';
+------------+----------------+------------+-----------+
| project_id | title          | freelancer | status    |
+------------+----------------+------------+-----------+
|          1 | Build REST API | Bob        | Completed |
+------------+----------------+------------+-----------+
1 row in set (0.00 sec)

mysql>
mysql> /*If check for cancelled*/
mysql> select
    -> p.project_id,
    -> p.title,
    -> u.name as freelancer,
    -> p.status
    -> from projects p
    -> left join users u on p.assigned_freelancer_id=u.user_id
    -> where p.status='Cancelled';
+------------+------------------------+------------+-----------+
| project_id | title                  | freelancer | status    |
+------------+------------------------+------------+-----------+
|          2 | Build Frondend project | NULL       | Cancelled |
+------------+------------------------+------------+-----------+
1 row in set (0.00 sec)
```

Review and Rating: After project completion, both clients and freelancers can leave reviews and ratings for each other

```
mysql> /*Review and rating*/
mysql> /*Create review table*/
mysql> CREATE TABLE Reviews (
    ->     review_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     project_id INT NOT NULL,
    ->     reviewer_id INT NOT NULL,
    ->     reviewee_id INT NOT NULL,
    ->     rating INT CHECK (rating BETWEEN 1 AND 5),
    ->     review_text TEXT,
    ->     created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    ->     FOREIGN KEY (project_id) REFERENCES Projects(project_id),
    ->     FOREIGN KEY (reviewer_id) REFERENCES Users(user_id),
    ->     FOREIGN KEY (reviewee_id) REFERENCES Users(user_id)
    -> );
Query OK, 0 rows affected (0.08 sec)

mysql> /*insert values into review table*/
mysql> INSERT INTO Reviews (project_id, reviewer_id, reviewee_id, rating, review_text)
    -> VALUES (1, 1, 2, 5, 'Great work, delivered on time!');
Query OK, 1 row affected (0.01 sec)

mysql> /*the above for the client reveiws freelancer*/
mysql>
mysql> /*Now - freelancer reviews client*/
mysql> INSERT INTO Reviews (project_id, reviewer_id, reviewee_id, rating, review_text)
    -> VALUES (1, 2, 1, 4, 'Clear requirements and good communication.');
Query OK, 1 row affected (0.01 sec)

mysql> /*view all reviews for a user*/
mysql> SELECT
    ->     r.review_id,
    ->     u1.name AS reviewer,
    ->     u2.name AS reviewee,
    ->     r.rating,
    ->     r.review_text,
    ->     r.created_at,
    ->     p.title AS project_title
    -> FROM Reviews r
    -> JOIN Users u1 ON r.reviewer_id = u1.user_id
    -> JOIN Users u2 ON r.reviewee_id = u2.user_id
    -> JOIN Projects p ON r.project_id = p.project_id
    -> WHERE r.reviewee_id = 2; -- reviews received by user_id 2 (freelancer)
+-----------+----------+----------+--------+------------------------------+---------------------+---------------+
| review_id | reviewer | reviewee | rating | review_text                  | created_at          | project_title |
+-----------+----------+----------+--------+------------------------------+---------------------+---------------+
|         1 | Alice    | Bob      |      5 | Great work, delivered on time! | 2025-06-22 01:08:34 | Build REST API |
+-----------+----------+----------+--------+------------------------------+---------------------+---------------+
1 row in set (0.01 sec)

mysql> select
    -> r.review_id,
    -> u1.name as reviewer,
    -> u2.name as reviewee,
    -> r.rating,
    -> r.review_text,
    -> r.created_at,
    -> p.title as project_title
    -> from reviews r
    -> JOIN Users u1 ON r.reviewer_id = u1.user_id
    -> JOIN Users u2 ON r.reviewee_id = u2.user_id
    -> JOIN Projects p ON r.project_id = p.project_id
    -> WHERE r.reviewee_id = 1; -- reviews received by user_id 1 (client)
+-----------+----------+----------+--------+-----------------------------------------+---------------------+---------------+
| review_id | reviewer | reviewee | rating | review_text                             | created_at          | project_title |
+-----------+----------+----------+--------+-----------------------------------------+---------------------+---------------+
|         2 | Bob      | Alice    |      4 | Clear requirements and good communication. | 2025-06-22 01:10:04 | Build REST API |
+-----------+----------+----------+--------+-----------------------------------------+---------------------+---------------+
1 row in set (0.00 sec)
```

Skill Management: Projects and freelancers can be associated with multiple skills.

```
mysql> /*Skill management*/
mysql> /*create freelancerSkills table to link freelancers to their skills*/
mysql> CREATE TABLE FreelancerSkills (
    ->     user_id INT NOT NULL,
    ->     skill_id INT NOT NULL,
    ->     PRIMARY KEY (user_id, skill_id),
    ->     FOREIGN KEY (user_id) REFERENCES Users(user_id),
    ->     FOREIGN KEY (skill_id) REFERENCES Skills(skill_id)
    -> );
Query OK, 0 rows affected (0.07 sec)

mysql> /*insert values into freelancerSkills table*/
mysql> insert into freelancerSkills(user_id,skill_id)
    -> values(1,1),(1,2),(2,3);
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> /*Query to get freelancer's skills*/
mysql> SELECT s.skill_name
    -> FROM FreelancerSkills fs
    -> JOIN Skills s ON fs.skill_id = s.skill_id
    -> WHERE fs.user_id = 2;
+------------+
| skill_name |
+------------+
| React.js   |
+------------+
1 row in set (0.00 sec)
```

Bonus Features

- **Messaging system between clients and freelancers (schema only).**

```
mysql> /*Bonus feature*/
mysql> /*Messaging system between clients and freelancers (schema only).*/
mysql> /*create table for messages*/
mysql> CREATE TABLE Messages (
    ->     message_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     sender_id INT NOT NULL,
    ->     receiver_id INT NOT NULL,
    ->     project_id INT, -- Optional: to link messages to a project
    ->     message_text TEXT NOT NULL,
    ->     sent_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    ->     is_read BOOLEAN DEFAULT FALSE,
    ->
    ->     FOREIGN KEY (sender_id) REFERENCES Users(user_id),
    ->     FOREIGN KEY (receiver_id) REFERENCES Users(user_id),
    ->     FOREIGN KEY (project_id) REFERENCES Projects(project_id)
    -> );
Query OK, 0 rows affected (0.07 sec)

mysql> /*Insert values into messages table*/
mysql> INSERT INTO Messages (sender_id, receiver_id, project_id, message_text)
    -> VALUES (1, 2, 3, 'Hi, I'm interested in your project.');
ERROR 1452 (23000): Cannot add or update a child row: a foreign key constraint fails (`devifyx`.`messages`, CONSTRAINT `messages_ibfk_3` FOREIGN KEY (`proje
ct_id`) REFERENCES `projects` (`project_id`))
mysql> INSERT INTO Messages (sender_id, receiver_id, project_id, message_text)
    -> VALUES (1, 2, 2, 'Hi, I'm interested in your project.');
Query OK, 1 row affected (0.01 sec)

mysql> /*Fetch chat between two users*/
mysql> SELECT
    ->     m.message_id,
    ->     u1.name AS sender,
    ->     u2.name AS receiver,
    ->     m.message_text,
    ->     m.sent_at
    -> FROM Messages m
    -> JOIN Users u1 ON m.sender_id = u1.user_id
    -> JOIN Users u2 ON m.receiver_id = u2.user_id
    -> WHERE (m.sender_id = 1 AND m.receiver_id = 2)
    ->    OR (m.sender_id = 2 AND m.receiver_id = 1)
    -> ORDER BY m.sent_at;
+------------+--------+----------+-----------------------------------+---------------------+
| message_id | sender | receiver | message_text                      | sent_at             |
+------------+--------+----------+-----------------------------------+---------------------+
|          2 | Alice  | Bob      | Hi, I?m interested in your project. | 2025-06-22 01:20:17 |
+------------+--------+----------+-----------------------------------+---------------------+
```

- **Notification system for important events (e.g., bid placed, bid accepted).**

```
mysql> /*NoNotification system for important events (e.g., bid placed, bid accepted).*/
mysql> /*create table notification*/
mysql> CREATE TABLE Notifications (
    ->     notification_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     user_id INT NOT NULL,
    ->     message TEXT NOT NULL,
    ->     is_read BOOLEAN DEFAULT FALSE,
    ->     created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    ->     FOREIGN KEY (user_id) REFERENCES Users(user_id)
    -> );
Query OK, 0 rows affected (0.06 sec)

mysql> /*insert into notification table*/
mysql> INSERT INTO Notifications (user_id, message)
    -> VALUES (1, 'A new bid has been placed on your project: "Build Website".');
Query OK, 1 row affected (0.01 sec)

mysql> /*the above query for notification for bid placed(to client)*/
mysql>
mysql> /*now for notification for bid Accepted(to freelancer)*/
mysql> INSERT INTO Notifications (user_id, message)
    -> VALUES (2, 'Your bid on project "Build Website" has been accepted.');
Query OK, 1 row affected (0.01 sec)

mysql> /*query unread notification*/
mysql> SELECT message, created_at
    -> FROM Notifications
    -> WHERE user_id = 2 AND is_read = FALSE
    -> ORDER BY created_at DESC;
+-----------------------------------------------------+---------------------+
| message                                             | created_at          |
+-----------------------------------------------------+---------------------+
| Your bid on project "Build Website" has been accepted. | 2025-06-22 01:23:42 |
+-----------------------------------------------------+---------------------+
1 row in set (0.00 sec)

mysql> SELECT message, created_at
    -> FROM Notifications
    -> WHERE user_id = 1 AND is_read = FALSE
    -> ORDER BY created_at DESC;
+-----------------------------------------------------------+---------------------+
| message                                                   | created_at          |
+-----------------------------------------------------------+---------------------+
| A new bid has been placed on your project: "Build Website". | 2025-06-22 01:22:24 |
+-----------------------------------------------------------+---------------------+
```

- **Ability to attach files to projects and bids (schema only).**

```
mysql> /*Ability to attach files to projects and bids (schema only).*/
mysql> /*create table ProjectFiles*/
mysql> CREATE TABLE ProjectFiles (
    ->     file_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     project_id INT NOT NULL,
    ->     file_name VARCHAR(255) NOT NULL,
    ->     file_path VARCHAR(500) NOT NULL, -- or file_url if hosted externally
    ->     uploaded_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    ->     FOREIGN KEY (project_id) REFERENCES Projects(project_id)
    -> );
Query OK, 0 rows affected (0.07 sec)

mysql> /*create bid table*/
mysql> /*Stores file references attached to a bid.*/
mysql> CREATE TABLE BidFiles (
    ->     file_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     bid_id INT NOT NULL,
    ->     file_name VARCHAR(255) NOT NULL,
    ->     file_path VARCHAR(500) NOT NULL,
    ->     uploaded_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    ->     FOREIGN KEY (bid_id) REFERENCES Bids(bid_id)
    -> );
Query OK, 0 rows affected (0.05 sec)
```

```
mysql> /*Insert into projectFiles table*/
mysql> INSERT INTO ProjectFiles (project_id, file_name, file_path)
    -> VALUES (1, 'requirements.pdf', '/uploads/projects/1/requirements.pdf');
Query OK, 1 row affected (0.01 sec)

mysql> /*Insert into Bid table*/
mysql> INSERT INTO BidFiles (bid_id, file_name, file_path)
    -> VALUES (1, 'proposal.docx', '/uploads/bids/10/proposal.docx');
Query OK, 1 row affected (0.01 sec)

mysql> insert into bids(bid_id,project_id,freelancer_id,amount,timeline_days,status,bid_time)
    -> values(10,2,2,100.00,5,'Pending',NOW());
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO BidFiles (bid_id, file_name, file_path)
    -> VALUES (10, 'proposal.docx', '/uploads/bids/10/proposal.docx');
Query OK, 1 row affected (0.01 sec)

mysql> select * from projectFiles;
+---------+------------+------------------+-------------------------------------+---------------------+
| file_id | project_id | file_name        | file_path                           | uploaded_at         |
+---------+------------+------------------+-------------------------------------+---------------------+
|       1 |          1 | requirements.pdf | /uploads/projects/1/requirements.pdf | 2025-06-22 01:30:04 |
+---------+------------+------------------+-------------------------------------+---------------------+
1 row in set (0.00 sec)

mysql> select * from bidFiles;
+---------+--------+---------------+-------------------------------+---------------------+
| file_id | bid_id | file_name     | file_path                     | uploaded_at         |
+---------+--------+---------------+-------------------------------+---------------------+
|       2 |      1 | proposal.docx | /uploads/bids/10/proposal.docx | 2025-06-22 01:31:43 |
|       3 |     10 | proposal.docx | /uploads/bids/10/proposal.docx | 2025-06-22 01:34:24 |
+---------+--------+---------------+-------------------------------+---------------------+
2 rows in set (0.00 sec)

mysql> /*List all files uploaded for a given project*/
mysql> SELECT file_name, file_path, uploaded_at
    -> FROM ProjectFiles
    -> WHERE project_id = 1;
+------------------+-------------------------------------+---------------------+
| file_name        | file_path                           | uploaded_at         |
+------------------+-------------------------------------+---------------------+
| requirements.pdf | /uploads/projects/1/requirements.pdf | 2025-06-22 01:30:04 |
+------------------+-------------------------------------+---------------------+
1 row in set (0.00 sec)

mysql> /*Join with Projects table to show project title + file*/
mysql> SELECT p.title, pf.file_name, pf.file_path, pf.uploaded_at
    -> FROM ProjectFiles pf
    -> JOIN Projects p ON pf.project_id = p.project_id;
+----------------+------------------+-------------------------------------+---------------------+
| title          | file_name        | file_path                           | uploaded_at         |
+----------------+------------------+-------------------------------------+---------------------+
| Build REST API | requirements.pdf | /uploads/projects/1/requirements.pdf | 2025-06-22 01:30:04 |
+----------------+------------------+-------------------------------------+---------------------+
1 row in set (0.00 sec)

mysql> /*Total number of files per project*/
mysql> SELECT project_id, COUNT(*) AS total_files
    -> FROM ProjectFiles
    -> GROUP BY project_id;
+------------+-------------+
| project_id | total_files |
+------------+-------------+
|          1 |           1 |
+------------+-------------+
1 row in set (0.00 sec)
```

- **Analytics views (e.g., top freelancers, most active clients).**

```
mysql> CREATE OR REPLACE VIEW TopFreelancers AS
    -> SELECT
    ->     u.user_id,
    ->     u.name AS freelancer_name,
    ->     AVG(r2.rating) AS average_rating,
    ->     COUNT(r2.review_id) AS total_reviews
    -> FROM Users u
    -> JOIN Roles r ON u.role_id = r.role_id
    -> JOIN Reviews r2 ON u.user_id = r2.reviewee_id
    -> JOIN Projects p ON p.project_id = r2.project_id
    -> WHERE r.role_name = 'Freelancer'
    ->   AND p.status = 'Completed'
    -> GROUP BY u.user_id, u.name
    -> ORDER BY average_rating DESC
    -> LIMIT 10;
Query OK, 0 rows affected (0.02 sec)
```

```
mysql> CREATE OR REPLACE VIEW MostActiveClients AS
    -> SELECT
    ->     u.user_id,
    ->     u.name AS client_name,
    ->     COUNT(p.project_id) AS total_projects
    -> FROM Users u
    -> JOIN Roles r ON u.role_id = r.role_id
    -> JOIN Projects p ON u.user_id = p.client_id
    -> WHERE r.role_name = 'Client'
    -> GROUP BY u.user_id, u.name
    -> ORDER BY total_projects DESC
    -> LIMIT 10;
Query OK, 0 rows affected (0.01 sec)

mysql> CREATE OR REPLACE VIEW MostBiddedProjects AS
    -> SELECT
    ->     p.project_id,
    ->     p.title,
    ->     COUNT(b.bid_id) AS total_bids
    -> FROM Projects p
    -> JOIN Bids b ON p.project_id = b.project_id
    -> GROUP BY p.project_id, p.title
    -> ORDER BY total_bids DESC
    -> LIMIT 10;
Query OK, 0 rows affected (0.01 sec)

mysql> CREATE OR REPLACE VIEW AverageBidAmount AS
    -> SELECT
    ->     p.project_id,
    ->     p.title,
    ->     ROUND(AVG(b.amount), 2) AS avg_bid_amount,
    ->     COUNT(b.bid_id) AS bid_count
    -> FROM Projects p
    -> JOIN Bids b ON p.project_id = b.project_id
    -> GROUP BY p.project_id, p.title;
Query OK, 0 rows affected (0.01 sec)

mysql> select * from TopFreelancers;
+---------+-----------------+---------------+---------------+
| user_id | freelancer_name | average_rating | total_reviews |
+---------+-----------------+---------------+---------------+
|       2 | Bob             |         5.0000 |             1 |
+---------+-----------------+---------------+---------------+
1 row in set (0.00 sec)
```

Write at least 3 complex queries (e.g., top freelancers by rating, projects with most bids, average bid per project).

```
mysql> -- Get top freelancers
mysql> SELECT * FROM TopFreelancers;
+---------+-----------------+----------------+---------------+
| user_id | freelancer_name | average_rating | total_reviews |
+---------+-----------------+----------------+---------------+
|       2 | Bob             |         5.0000 |             1 |
+---------+-----------------+----------------+---------------+
1 row in set (0.01 sec)

mysql>
mysql> -- See most active clients
mysql> SELECT * FROM MostActiveClients;
+---------+-------------+----------------+
| user_id | client_name | total_projects |
+---------+-------------+----------------+
|       1 | Alice       |              1 |
+---------+-------------+----------------+
1 row in set (0.00 sec)

mysql>
mysql> -- List top 5 most bidded projects
mysql> SELECT * FROM MostBiddedProjects LIMIT 5;
+------------+------------------------+------------+
| project_id | title                  | total_bids |
+------------+------------------------+------------+
|          1 | Build REST API         |          1 |
|          2 | Build Frondend project |          1 |
+------------+------------------------+------------+
2 rows in set (0.00 sec)

mysql> select * from AverageBidAmount;
+------------+------------------------+----------------+-----------+
| project_id | title                  | avg_bid_amount | bid_count |
+------------+------------------------+----------------+-----------+
|          1 | Build REST API         |         950.00 |         1 |
|          2 | Build Frondend project |         100.00 |         1 |
+------------+------------------------+----------------+-----------+
2 rows in set (0.00 sec)
```