Project Report

Name	**Sadia**	**Sheeraz**
Roll No	**FL23745**	**FL23724**
CLASS	**BSSE 4th 'A' Morning**	
Course	**Database System**	
Submitted To	**Sir Usman Shahid**	

Catalog

Tender Management System

1. Introduction

This report presents the design and implementation of a relational database for a Tender Management System. The platform connects clients, who initiate tenders for projects, with companies that submit bids or proposals. Key features include user account management, project creation, bidding workflows, proposal evaluation, administrative review, and payment tracking.

The database has been optimized for scalability, performance, and data security. It supports concurrent operations and ensures reliable transaction handling across all modules.

2. Project Objectives

- Store and manage tender-related records efficiently.
- Provide full CRUD (Create, Read, Update, Delete) functionality for all core entities.
- Maintain data consistency using foreign keys and normalization.
- Enable meaningful filtering and reporting based on user roles.

3. Key Entities and Roles

- Users: Individuals using the system, categorized as Clients, Companies, or Admins.
- Projects: Tenders created by clients for bidding.
- Bids and Proposals: Submissions from companies to participate in projects.
- Payments: Records of payments made upon project completion.
- Ratings: Feedback exchanged between clients and companies.
- Notifications: System messages triggered by user actions.
- Meetings: Scheduled interactions for real-time bidding sessions.

4. Database Schema Overview

The schema comprises normalized tables (3NF) with well-defined relationships:

- Primary and foreign key constraints ensure referential integrity.
- User roles are defined using an ENUM column for clarity and validation.
- Tables are structured to support scalability and future extensibility.

5. Lab 01: Installation of RDBMS

MySQL Community Server and MySQL Workbench were installed. The environment was configured to create and manage the Tender Management System database. All functionalities like table creation, querying, and relationship testing were confirmed.

6. Creation of Tender Flow Database & Table

Code:

```
CREATE DATABASE TenderManagementSystem;

USE TenderManagementSystem;

CREATE TABLE Users (
    user_id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    email VARCHAR(100) UNIQUE NOT NULL,
    password VARCHAR(255) NOT NULL,
    role ENUM('Client', 'Company', 'Admin') NOT NULL,
    verified BOOLEAN DEFAULT FALSE,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
  );
CREATE TABLE CompanyDetails (
    company_id INT PRIMARY KEY,
    company_name VARCHAR(100),
    certification_url VARCHAR(255),
    blacklisted BOOLEAN DEFAULT FALSE,
    FOREIGN KEY (company_id) REFERENCES Users(user_id)
  );
CREATE TABLE Projects (
    project_id INT AUTO_INCREMENT PRIMARY KEY,
```

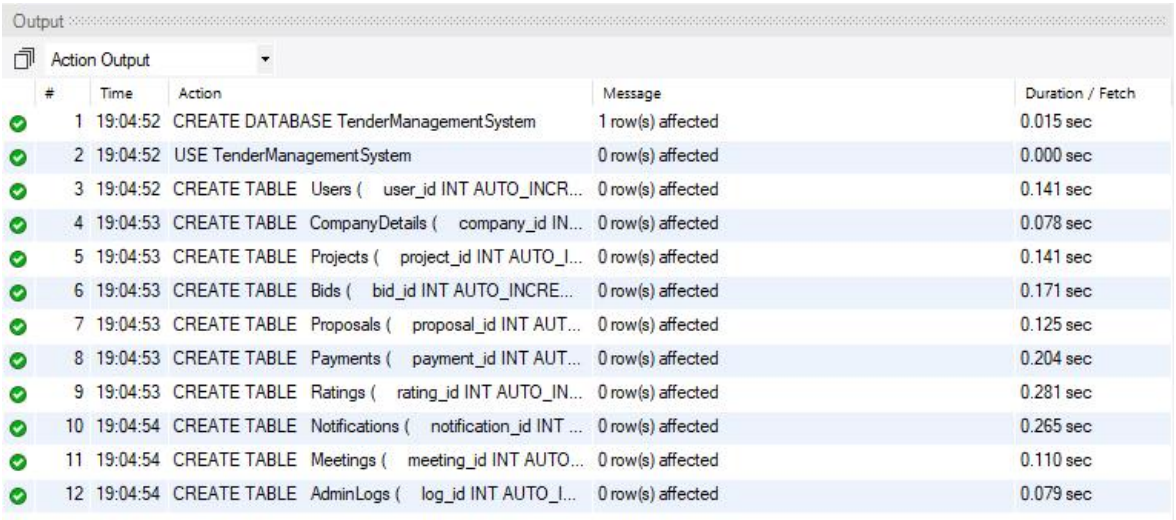
```
    client_id INT NOT NULL,
    title VARCHAR(150) NOT NULL,
    description TEXT,
    category VARCHAR(100),
    budget DECIMAL(12, 2),
    bid_type ENUM('Real-Time', 'Proposal-Based'),
    status ENUM('Pending', 'Approved', 'Rejected', 'Completed') DEFAULT 'Pending',
    posted_on TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (client_id) REFERENCES Users(user_id)
  );
CREATE TABLE Bids (
    bid_id INT AUTO_INCREMENT PRIMARY KEY,
    project_id INT NOT NULL,
    company_id INT NOT NULL,
    amount DECIMAL(12, 2),
    bid_time DATETIME DEFAULT CURRENT_TIMESTAMP,
    status ENUM('Submitted', 'Accepted', 'Rejected') DEFAULT 'Submitted',
    FOREIGN KEY (project_id) REFERENCES Projects(project_id),
    FOREIGN KEY (company_id) REFERENCES Users(user_id)
  );
CREATE TABLE Proposals (
    proposal_id INT AUTO_INCREMENT PRIMARY KEY,
    project_id INT NOT NULL,
    company_id INT NOT NULL,
    proposal_text TEXT,
    submission_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    status ENUM('Submitted', 'Accepted', 'Rejected') DEFAULT 'Submitted',
```

```
    FOREIGN KEY (project_id) REFERENCES Projects(project_id),
    FOREIGN KEY (company_id) REFERENCES Users(user_id)
  );
CREATE TABLE Payments (
    payment_id INT AUTO_INCREMENT PRIMARY KEY,
    project_id INT NOT NULL,
    client_id INT NOT NULL,
    company_id INT NOT NULL,
    amount DECIMAL(12, 2),
    method ENUM('Bank Transfer', 'Cheque'),
    status ENUM('Pending', 'Completed', 'Failed') DEFAULT 'Pending',
    paid_on TIMESTAMP,
    FOREIGN KEY (project_id) REFERENCES Projects(project_id),
    FOREIGN KEY (client_id) REFERENCES Users(user_id),
    FOREIGN KEY (company_id) REFERENCES Users(user_id)
  );
CREATE TABLE Ratings (
    rating_id INT AUTO_INCREMENT PRIMARY KEY,
    project_id INT NOT NULL,
    rated_by INT NOT NULL,
    rated_user INT NOT NULL,
    role ENUM('Client', 'Company'),
    rating_value INT CHECK (rating_value BETWEEN 1 AND 5),
    feedback TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (project_id) REFERENCES Projects(project_id),
    FOREIGN KEY (rated_by) REFERENCES Users(user_id),
```

```
    FOREIGN KEY (rated_user) REFERENCES Users(user_id)
  );
CREATE TABLE Notifications (
    notification_id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT NOT NULL,
    message TEXT,
    seen BOOLEAN DEFAULT FALSE,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES Users(user_id)
  );
CREATE TABLE Meetings (
    meeting_id INT AUTO_INCREMENT PRIMARY KEY,
    project_id INT NOT NULL,
    company_id INT NOT NULL,
    scheduled_on DATETIME,
    meeting_link VARCHAR(255),
    status ENUM('Scheduled', 'Completed', 'Cancelled') DEFAULT 'Scheduled',
    FOREIGN KEY (project_id) REFERENCES Projects(project_id),
    FOREIGN KEY (company_id) REFERENCES Users(user_id)
  );
CREATE TABLE AdminLogs (
    log_id INT AUTO_INCREMENT PRIMARY KEY,
    admin_id INT NOT NULL,
    action_type VARCHAR(100),
    description TEXT,
    logged_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (admin_id) REFERENCES Users(user_id)
```

);

Output:



Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	19:04:52	CREATE DATABASE TenderManagementSystem	1 row(s) affected	0.015 sec
2	19:04:52	USE TenderManagementSystem	0 row(s) affected	0.000 sec
3	19:04:52	CREATE TABLE Users (user_id INT AUTO_INCR...	0 row(s) affected	0.141 sec
4	19:04:53	CREATE TABLE CompanyDetails (company_id IN...	0 row(s) affected	0.078 sec
5	19:04:53	CREATE TABLE Projects (project_id INT AUTO_I...	0 row(s) affected	0.141 sec
6	19:04:53	CREATE TABLE Bids (bid_id INT AUTO_INCRE...	0 row(s) affected	0.171 sec
7	19:04:53	CREATE TABLE Proposals (proposal_id INT AUT...	0 row(s) affected	0.125 sec
8	19:04:53	CREATE TABLE Payments (payment_id INT AUT...	0 row(s) affected	0.204 sec
9	19:04:53	CREATE TABLE Ratings (rating_id INT AUTO_IN...	0 row(s) affected	0.281 sec
10	19:04:54	CREATE TABLE Notifications (notification_id INT ...	0 row(s) affected	0.265 sec
11	19:04:54	CREATE TABLE Meetings (meeting_id INT AUTO...	0 row(s) affected	0.110 sec
12	19:04:54	CREATE TABLE AdminLogs (log_id INT AUTO_I...	0 row(s) affected	0.079 sec

7. Inserting Data into All Tables

Code:

```
INSERT INTO Users (name, email, password, role, verified) VALUES

('Ali Raza', 'ali@example.com', 'hashed_pass1', 'Client', TRUE),

('TechCorp Ltd.', 'techcorp@example.com', 'hashed_pass2', 'Company', TRUE),

('BuildX Solutions', 'buildx@example.com', 'hashed_pass3', 'Company', TRUE),

('Admin Huma', 'admin@example.com', 'hashed_pass4', 'Admin', TRUE);

INSERT INTO CompanyDetails (company_id, company_name, certification_url, blacklisted) VALUES

(2, 'TechCorp Ltd.', 'http://certs.example.com/techcorp', FALSE),

(3, 'BuildX Solutions', 'http://certs.example.com/buildx', FALSE);

INSERT INTO Projects (client_id, title, description, category, budget, bid_type, status) VALUES

(1, 'Website Development', 'Create a responsive and SEO-friendly company website', 'Web Design', 80000, 'Proposal-Based', 'Approved'),

(1, 'Mobile App Prototype', 'Develop a cross-platform MVP', 'Mobile Development', 100000, 'Real-Time', 'Pending');

INSERT INTO Bids (project_id, company_id, amount, bid_time, status) VALUES

(2, 2, 95000, NOW(), 'Submitted'),

(2, 3, 90000, NOW(), 'Submitted');

INSERT INTO Proposals (project_id, company_id, proposal_text, submission_date, status) VALUES

(1, 2, 'We propose to use ReactJS and NodeJS with a 4-week timeline.', NOW(), 'Submitted'),

(1, 3, 'BuildX will deliver the project using Laravel and Vue.js stack.', NOW(), 'Submitted');

INSERT INTO Payments (project_id, client_id, company_id, amount, method, status, paid_on) VALUES

(1, 1, 3, 78000, 'Bank Transfer', 'Completed', NOW());

INSERT INTO Ratings (project_id, rated_by, rated_user, role, rating_value, feedback, created_at) VALUES

(1, 1, 3, 'Company', 5, 'BuildX completed the project on time and met all expectations.', NOW()),
```

(1, 3, 1, 'Client', 4, 'Client provided clear instructions and prompt feedback.', NOW());

INSERT INTO Notifications (user_id, message, seen, created_at) VALUES

(2, 'You have been invited to a new bidding session.', FALSE, NOW()),

(1, 'Your project \"Mobile App Prototype\" has received new bids.', FALSE, NOW());

INSERT INTO Meetings (project_id, company_id, scheduled_on, meeting_link, status) VALUES

(2, 2, '2025-06-10 10:00:00', 'https://meet.example.com/session123', 'Scheduled'),

(2, 3, '2025-06-10 11:00:00', 'https://meet.example.com/session456', 'Scheduled');

INSERT INTO AdminLogs (admin_id, action_type, description, logged_at) VALUES

(4, 'Project Approval', 'Approved project ID 1 by client Ali Raza.', NOW()),

(4, 'Bid Review', 'Reviewed bids for project ID 2.', NOW());

Output:

13	19:11:38	INSERT INTO Users (name, email, password, role, verified) VALUES ('Ali Raza', 'ali@example.c...	4 row(s) affected Records: 4	Duplicates: 0	Warnings: 0	0.047 sec
14	19:11:38	INSERT INTO CompanyDetails (company_id, company_name, certification_url, blacklisted) VAL...	2 row(s) affected Records: 2	Duplicates: 0	Warnings: 0	0.031 sec
15	19:11:38	INSERT INTO Projects (client_id, title, description, category, budget, bid_type, status) VALUES ...	2 row(s) affected Records: 2	Duplicates: 0	Warnings: 0	0.031 sec
16	19:11:38	INSERT INTO Bids (project_id, company_id, amount, bid_time, status) VALUES (2, 2, 95000, N...	2 row(s) affected Records: 2	Duplicates: 0	Warnings: 0	0.016 sec
17	19:11:38	INSERT INTO Proposals (project_id, company_id, proposal_text, submission_date, status) VAL...	2 row(s) affected Records: 2	Duplicates: 0	Warnings: 0	0.031 sec
18	19:11:38	INSERT INTO Payments (project_id, client_id, company_id, amount, method, status, paid_on) V...	1 row(s) affected			0.031 sec
19	19:11:38	INSERT INTO Ratings (project_id, rated_by, rated_user, role, rating_value, feedback, created_...	2 row(s) affected Records: 2	Duplicates: 0	Warnings: 0	0.032 sec
20	19:11:38	INSERT INTO Notifications (user_id, message, seen, created_at) VALUES (2, 'You have been i...	2 row(s) affected Records: 2	Duplicates: 0	Warnings: 0	0.047 sec
21	19:11:38	INSERT INTO Meetings (project_id, company_id, scheduled_on, meeting_link, status) VALUES...	2 row(s) affected Records: 2	Duplicates: 0	Warnings: 0	0.047 sec
22	19:11:38	INSERT INTO AdminLogs (admin_id, action_type, description, logged_at) VALUES (4, 'Project ...	2 row(s) affected Records: 2	Duplicates: 0	Warnings: 0	0.047 sec

8. Lab 02: Retrieving Data Using SELECT

SQL SELECT statements were used to retrieve data from the system

Code:

SELECT * FROM Users;

SELECT name, email, role FROM Users;

SELECT * FROM Projects;

SELECT title, category, budget FROM Projects WHERE status = 'Approved';

Output:

	title	category	budget
▶	Website Development	Web Design	80000.00

9. Lab 03: WHERE Clause and ORDER BY

Filtering and sorting examples

Code:

SELECT * FROM Projects WHERE budget > 50000 ORDER BY budget DESC;

SELECT * FROM Users WHERE name LIKE 'A%';

Output:

user_id	name	email	password	role	verified	created_at
1	Ali Raza	ali@example.com	hashed_pass1	Client	1	2025-06-04 19:11:38
4	Admin Huma	admin@example.com	hashed_pass4	Admin	1	2025-06-04 19:11:38
NULL	NULL	NULL	NULL	NULL	NULL	NULL

10. Lab 04: DDL Statements

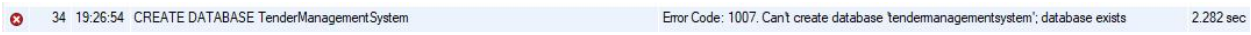
Tables were created using DDL commands. Each table includes appropriate data types and constraints.

Code:

CREATE DATABASE TenderManagementSystem;

USE TenderManagementSystem;

Output:



11. Lab 05: Schema Constraints

Constraints used:

- PRIMARY KEY on each table
- FOREIGN KEY to ensure relational integrity
- CHECK (rating_value BETWEEN 1 AND 5) to validate feedback
- UNIQUE on email in Users table
- ENUM to restrict values in role and status columns

Code:

CREATE DATABASE TenderManagementSystem;

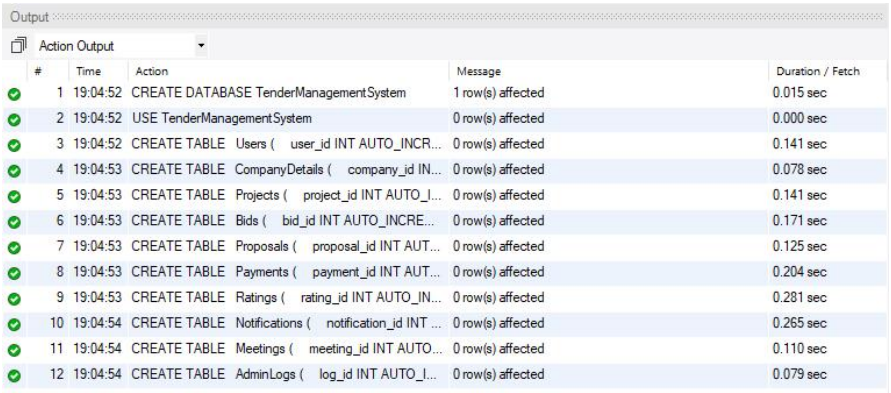
```
USE TenderManagementSystem;
CREATE TABLE Users (
    user_id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    email VARCHAR(100) UNIQUE NOT NULL,
    password VARCHAR(255) NOT NULL,
    role ENUM('Client', 'Company', 'Admin') NOT NULL,
    verified BOOLEAN DEFAULT FALSE,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
  );
CREATE TABLE CompanyDetails (
    company_id INT PRIMARY KEY,
    company_name VARCHAR(100),
    certification_url VARCHAR(255),
    blacklisted BOOLEAN DEFAULT FALSE,
    FOREIGN KEY (company_id) REFERENCES Users(user_id)
  );
CREATE TABLE Projects (
    project_id INT AUTO_INCREMENT PRIMARY KEY,
    client_id INT NOT NULL,
    title VARCHAR(150) NOT NULL,
    description TEXT,
    category VARCHAR(100),
    budget DECIMAL(12, 2),
    bid_type ENUM('Real-Time', 'Proposal-Based'),
    status ENUM('Pending', 'Approved', 'Rejected', 'Completed') DEFAULT 'Pending',
    posted_on TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
```

```
    FOREIGN KEY (client_id) REFERENCES Users(user_id)
  );
CREATE TABLE Bids (
    bid_id INT AUTO_INCREMENT PRIMARY KEY,
    project_id INT NOT NULL,
    company_id INT NOT NULL,
    amount DECIMAL(12, 2),
    bid_time DATETIME DEFAULT CURRENT_TIMESTAMP,
    status ENUM('Submitted', 'Accepted', 'Rejected') DEFAULT 'Submitted',
    FOREIGN KEY (project_id) REFERENCES Projects(project_id),
    FOREIGN KEY (company_id) REFERENCES Users(user_id)
  );
CREATE TABLE Proposals (
    proposal_id INT AUTO_INCREMENT PRIMARY KEY,
    project_id INT NOT NULL,
    company_id INT NOT NULL,
    proposal_text TEXT,
    submission_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    status ENUM('Submitted', 'Accepted', 'Rejected') DEFAULT 'Submitted',
    FOREIGN KEY (project_id) REFERENCES Projects(project_id),
    FOREIGN KEY (company_id) REFERENCES Users(user_id)
  );
CREATE TABLE Payments (
    payment_id INT AUTO_INCREMENT PRIMARY KEY,
    project_id INT NOT NULL,
    client_id INT NOT NULL,
    company_id INT NOT NULL,
```

```
    amount DECIMAL(12, 2),
    method ENUM('Bank Transfer', 'Cheque'),
    status ENUM('Pending', 'Completed', 'Failed') DEFAULT 'Pending',
    paid_on TIMESTAMP,
    FOREIGN KEY (project_id) REFERENCES Projects(project_id),
    FOREIGN KEY (client_id) REFERENCES Users(user_id),
    FOREIGN KEY (company_id) REFERENCES Users(user_id)
  );
CREATE TABLE Ratings (
    rating_id INT AUTO_INCREMENT PRIMARY KEY,
    project_id INT NOT NULL,
    rated_by INT NOT NULL,
    rated_user INT NOT NULL,
    role ENUM('Client', 'Company'),
    rating_value INT CHECK (rating_value BETWEEN 1 AND 5),
    feedback TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (project_id) REFERENCES Projects(project_id),
    FOREIGN KEY (rated_by) REFERENCES Users(user_id),
    FOREIGN KEY (rated_user) REFERENCES Users(user_id)
  );
CREATE TABLE Notifications (
    notification_id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT NOT NULL,
    message TEXT,
    seen BOOLEAN DEFAULT FALSE,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
```

```
    FOREIGN KEY (user_id) REFERENCES Users(user_id)
  );
CREATE TABLE Meetings (
    meeting_id INT AUTO_INCREMENT PRIMARY KEY,
    project_id INT NOT NULL,
    company_id INT NOT NULL,
    scheduled_on DATETIME,
    meeting_link VARCHAR(255),
    status ENUM('Scheduled', 'Completed', 'Cancelled') DEFAULT 'Scheduled',
    FOREIGN KEY (project_id) REFERENCES Projects(project_id),
    FOREIGN KEY (company_id) REFERENCES Users(user_id)
  );
CREATE TABLE AdminLogs (
    log_id INT AUTO_INCREMENT PRIMARY KEY,
    admin_id INT NOT NULL,
    action_type VARCHAR(100),
    description TEXT,
    logged_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (admin_id) REFERENCES Users(user_id)
  );
```

Output:



Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	19:04:52	CREATE DATABASE TenderManagementSystem	1 row(s) affected	0.015 sec
2	19:04:52	USE TenderManagementSystem	0 row(s) affected	0.000 sec
3	19:04:52	CREATE TABLE Users (user_id INT AUTO_INCR...	0 row(s) affected	0.141 sec
4	19:04:53	CREATE TABLE CompanyDetails (company_id IN...	0 row(s) affected	0.078 sec
5	19:04:53	CREATE TABLE Projects (project_id INT AUTO_I...	0 row(s) affected	0.141 sec
6	19:04:53	CREATE TABLE Bids (bid_id INT AUTO_INCRE...	0 row(s) affected	0.171 sec
7	19:04:53	CREATE TABLE Proposals (proposal_id INT AUT...	0 row(s) affected	0.125 sec
8	19:04:53	CREATE TABLE Payments (payment_id INT AUT...	0 row(s) affected	0.204 sec
9	19:04:53	CREATE TABLE Ratings (rating_id INT AUTO_IN...	0 row(s) affected	0.281 sec
10	19:04:54	CREATE TABLE Notifications (notification_id INT ...	0 row(s) affected	0.265 sec
11	19:04:54	CREATE TABLE Meetings (meeting_id INT AUTO...	0 row(s) affected	0.110 sec
12	19:04:54	CREATE TABLE AdminLogs (log_id INT AUTO_I...	0 row(s) affected	0.079 sec

12. Lab 06: Character Functions

Examples using character functions

Code:

```
SELECT name, UPPER(name), LENGTH(email) FROM Users;

SELECT CONCAT(name, ' (', role, ')') AS user_info FROM Users;
```

Output:

user_info
Ali Raza (Client)
TechCorp Ltd. (Company)
BuildX Solutions (Company)
Admin Huma (Admin)

13. Lab 07: Date Functions & Type Conversion

Date functions used

Code:

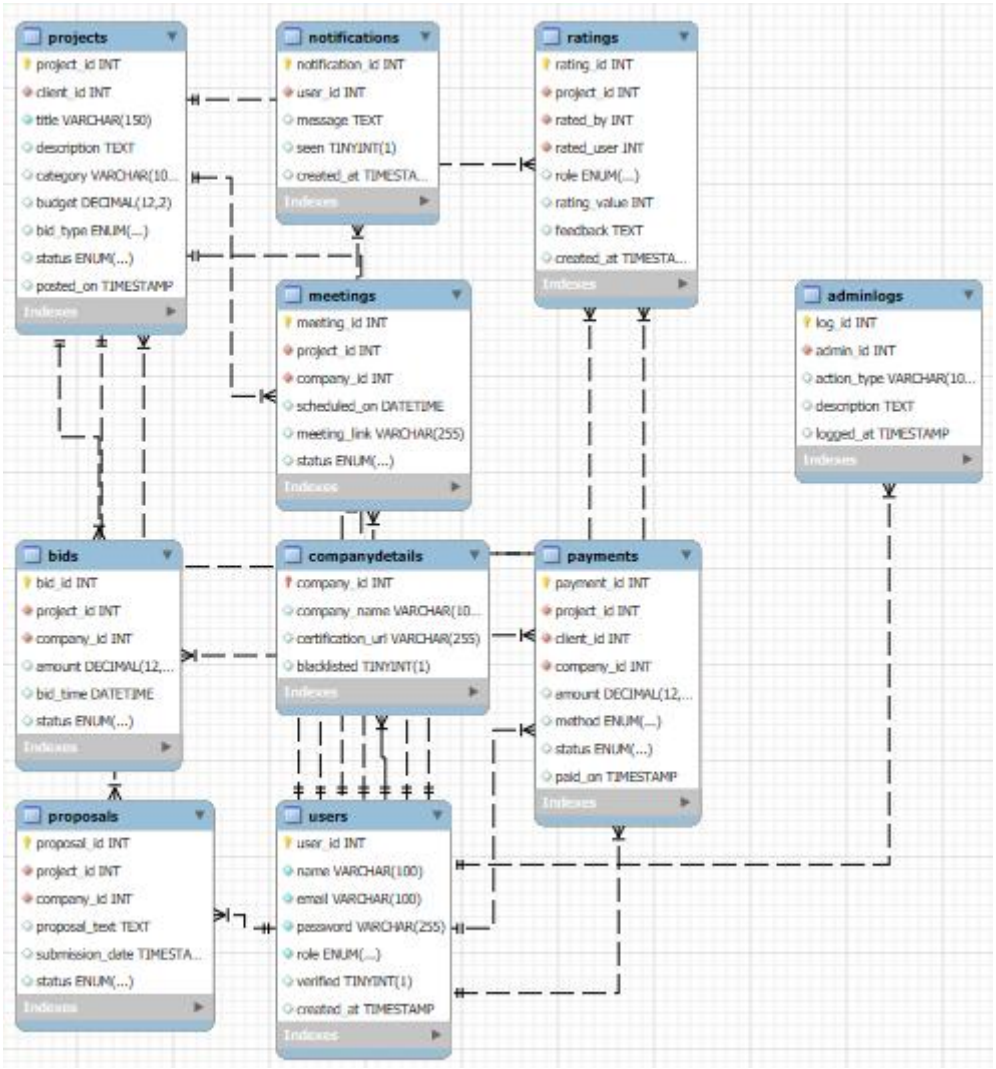
```
SELECT project_id, DATE_FORMAT(posted_on, '%D %M %Y') FROM Projects;

SELECT bid_id, TIMESTAMPDIFF(DAY, bid_time, NOW()) AS days_since_bid FROM Bids;
```

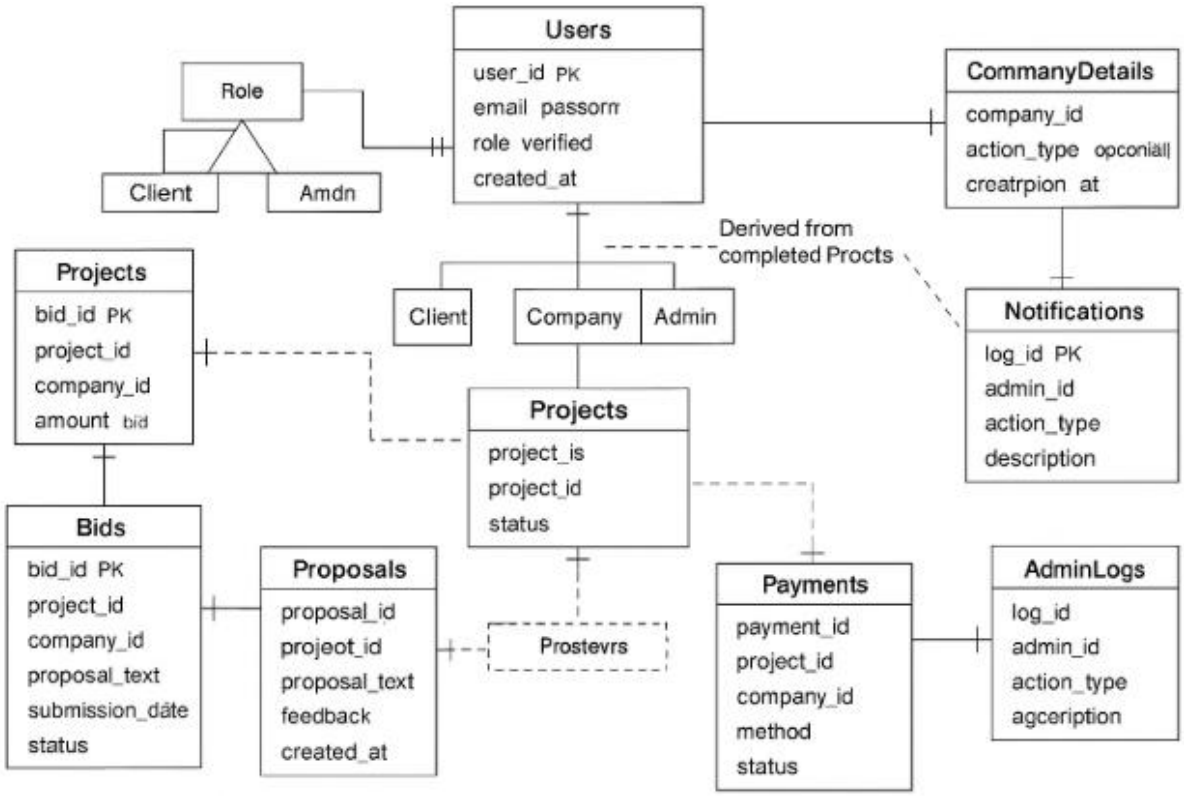
Output:

bid_id	days_since_bid
1	0
2	0

14. Lab 08: ERD Creation



15. Lab 10: Enhanced ERD



16. Lab 11: Aggregate Functions

Aggregation used to summarize data

Code:

SELECT project_id, AVG(amount) AS avg_bid FROM Bids GROUP BY project_id;

SELECT company_id, COUNT(*) FROM Proposals GROUP BY company_id;

Output:

company_id	COUNT(*)
2	1
3	1

17. Lab 12: Joins

Joins to combine data across tables

Code:

SELECT U.name, P.title FROM Users U

JOIN Projects P ON U.user_id = P.client_id;

SELECT B.bid_id, P.title, U.name

FROM Bids B

JOIN Projects P ON B.project_id = P.project_id

JOIN Users U ON B.company_id = U.user_id;

Output:

bid_id	title	name
1	Mobile App Prototype	TechCorp Ltd.
2	Mobile App Prototype	BuildX Solutions

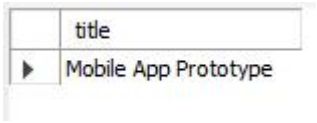
18. Lab 13: Subqueries

Subqueries used for comparison and filtering

Code:

```
SELECT name FROM Users WHERE user_id IN (
  SELECT company_id FROM Bids
  WHERE amount > (SELECT AVG(amount) FROM Bids)
);
SELECT title FROM Projects WHERE project_id IN (
  SELECT project_id FROM Bids GROUP BY project_id
  HAVING COUNT(*) < 3
);
```

Output:



	title
▶	Mobile App Prototype

Class Diagram