Database Mini Project Report

Project Title: Freelancer Payment & Project Tracker

Name: Kammari Vijaya Tejaswini

Date: 06/07/2025

Objective:

To design and implement a database system that helps **freelancers** manage clients, projects, tasks, invoices, and payments. The system enables tracking of deadlines, payment statuses, pending dues, and overdue invoices.

Tables Used:

Table Name	Purpose
`Clients`	Stores client information (name, email, phone)
`Projects`	Stores project details linked to clients
`Project_Tasks`	Stores tasks/milestones for each project
`Invoices`	Stores invoice details and payment status
`Payments`	Stores payment records linked to invoices

Relationships:

- One **Client** → Many **Projects**
- One **Project** → Many **Tasks**, **Invoices**
- One **Invoice** → Many **Payments**

Database Schema (SQL):

```
CREATE TABLE Clients (
    client_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(100),
    email VARCHAR(100),
    phone VARCHAR(15)
);
```

```sql
CREATE TABLE Projects (
    project_id INT PRIMARY KEY AUTO_INCREMENT,
    client_id INT,
    project_name VARCHAR(100),
    start_date DATE,
    end_date DATE,
    status VARCHAR(20),
    FOREIGN KEY (client_id) REFERENCES Clients(client_id)
);

CREATE TABLE Project_Tasks (
    task_id INT PRIMARY KEY AUTO_INCREMENT,
    project_id INT,
    task_name VARCHAR(100),
    due_date DATE,
    is_completed BOOLEAN,
    FOREIGN KEY (project_id) REFERENCES Projects(project_id)
);

CREATE TABLE Invoices (
    invoice_id INT PRIMARY KEY AUTO_INCREMENT,
    project_id INT,
    amount DECIMAL(10,2),
    due_date DATE,
    is_paid BOOLEAN,
    FOREIGN KEY (project_id) REFERENCES Projects(project_id)
);

CREATE TABLE Payments (
    payment_id INT PRIMARY KEY AUTO_INCREMENT,
    invoice_id INT,
    paid_amount DECIMAL(10,2),
    payment_date DATE,
    FOREIGN KEY (invoice_id) REFERENCES Invoices(invoice_id)
);
```

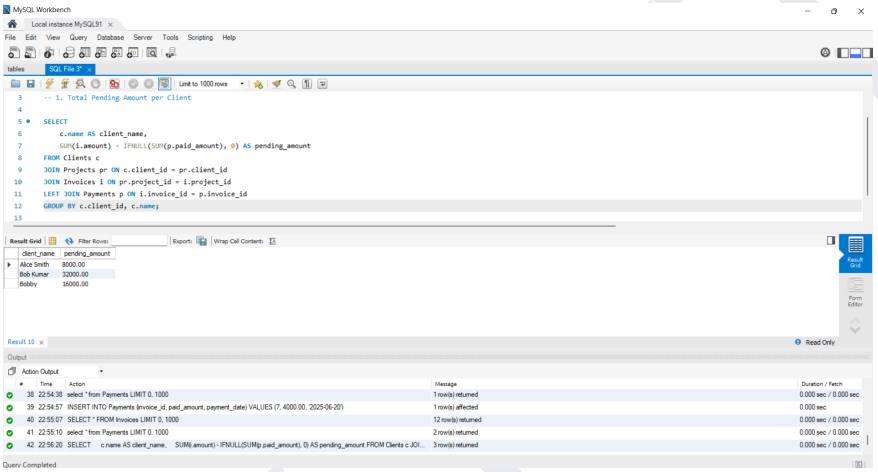
Sample Data Inserted:

```
-- Clients
INSERT INTO Clients (name, email, phone) VALUES
('Alice Smith', 'alice@example.com', '1234567890'),
('Bob Kumar', 'bob@example.com', '9876543210');

-- Projects
INSERT INTO Projects (client_id, project_name, start_date, end_date,
status) VALUES
(1, 'Website Redesign', '2025-07-01', '2025-08-01', 'Ongoing'),
(2, 'Mobile App Development', '2025-06-15', '2025-08-15',
'Ongoing');

-- Tasks
INSERT INTO Project_Tasks (project_id, task_name, due_date,
is_completed) VALUES
(1, 'Design UI Mockups', '2025-07-10', TRUE),
(1, 'Develop Frontend', '2025-07-20', FALSE),
(2, 'Setup Backend', '2025-07-25', FALSE);

-- Invoices
INSERT INTO Invoices (project_id, amount, due_date, is_paid) VALUES
(1, 5000.00, '2025-07-15', FALSE),
(2, 8000.00, '2025-07-25', TRUE);

-- Payments
INSERT INTO Payments (invoice_id, paid_amount, payment_date) VALUES
(2, 8000.00, '2025-07-20');
```

Sample Queries & Outputs:

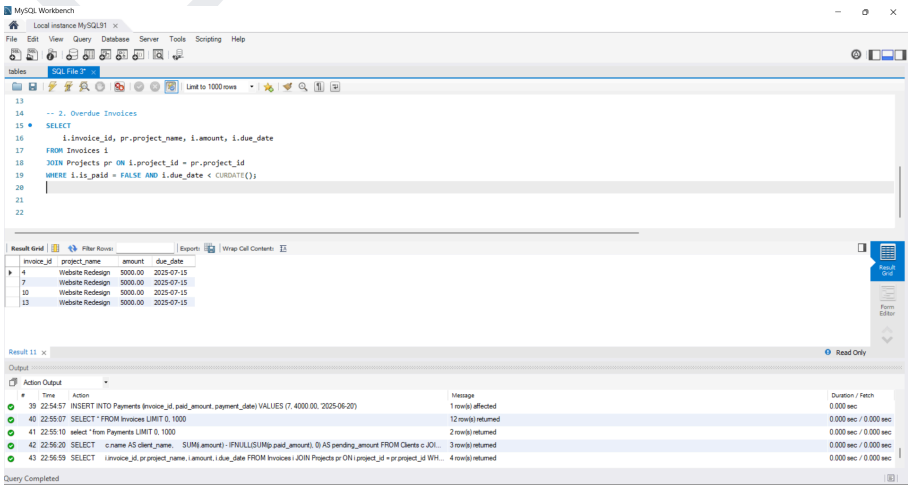
1. Total Pending Amount per Client:

```
SELECT
    c.name AS client_name,
    SUM(i.amount) - IFNULL(SUM(p.paid_amount), 0) AS pending_amount
FROM Clients c
JOIN Projects pr ON c.client_id = pr.client_id
JOIN Invoices i ON pr.project_id = i.project_id
LEFT JOIN Payments p ON i.invoice_id = p.invoice_id
GROUP BY c.client_id, c.name;
```



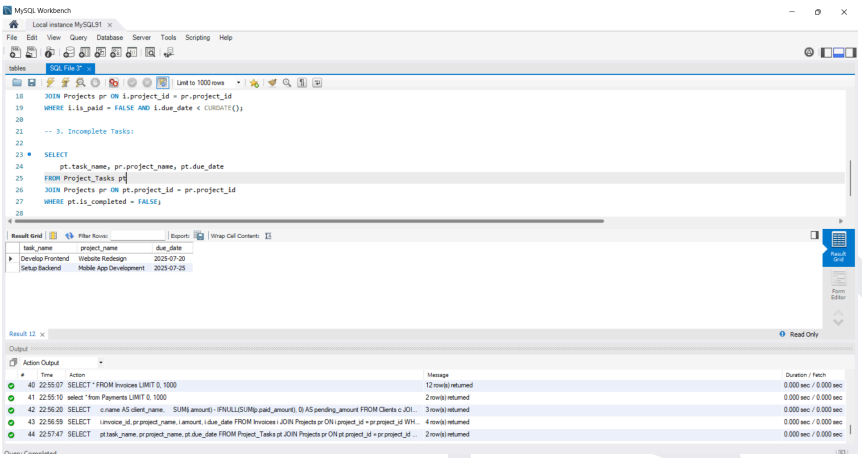
2. Overdue Invoices:

```
SELECT
    i.invoice_id, pr.project_name, i.amount, i.due_date
FROM Invoices i
JOIN Projects pr ON i.project_id = pr.project_id
WHERE i.is_paid = FALSE AND i.due_date < CURDATE();
```



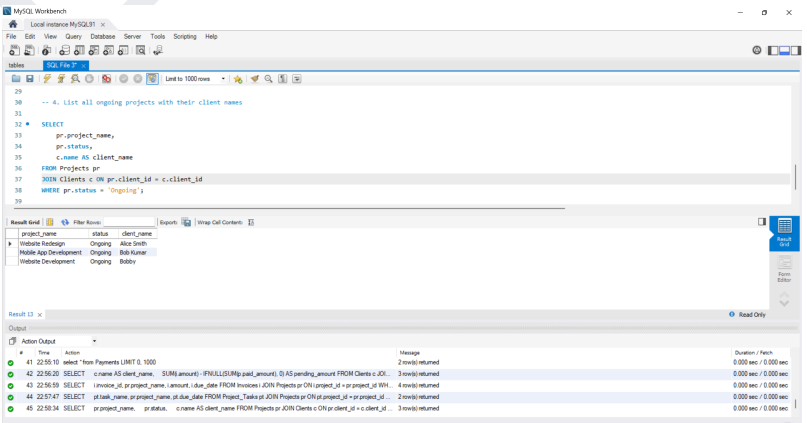
3. Incomplete Tasks:

```
SELECT
    pt.task_name, pr.project_name, pt.due_date
FROM Project_Tasks pt
JOIN Projects pr ON pt.project_id = pr.project_id
WHERE pt.is_completed = FALSE;
```



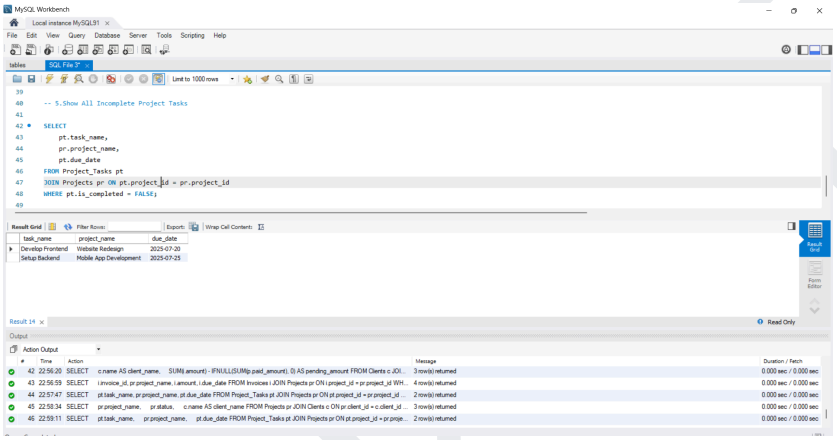
4. List all ongoing projects with their client names

```
SELECT
    pr.project_name,
    pr.status,
    c.name AS client_name
FROM Projects pr
JOIN Clients c ON pr.client_id = c.client_id
WHERE pr.status = 'Ongoing';
```



5.Show All Incomplete Project Tasks

```
SELECT
    pt.task_name,
    pr.project_name,
    pt.due_date
FROM Project_Tasks pt
JOIN Projects pr ON pt.project_id = pr.project_id
WHERE pt.is_completed = FALSE;
```



Conclusion:

This project simulates a real-world scenario where freelancers manage multiple projects and clients. It enables financial tracking, task management, and automated pending amount calculations. The design mimics platforms like Upwork and can serve as a backend for future web applications.