CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

NAAC GRADE A+
Accredited University

UNIVERSITY INSTITUTE OF COMPUTING

CASE STUDY REPORT
ON
COURT CASE MANAGEMENT SYSTEM

Program Name: BCA

Subject Name/Code: Database Management System (23CAT-251)

Submitted By:

Name: Jaismeen Kaur

UID: 23BCA10312

Section: 4-‘B’

Submitted To:

Name: Mr.Arvinder Singh

Designation: Assistant Professor

INTRODUCTION

In today’s fast-paced and increasingly digital world, the judicial system requires efficient tools to manage the growing volume of legal cases and associated data. This project focuses on the development of a **Court Case Management System (CCMS)** using a **Database Management System (DBMS)** to facilitate the organized handling of court proceedings, case information, and administrative processes.

The primary goal of this project is to design a system that can store, retrieve, and manage detailed information related to court cases—such as case IDs, types, parties involved, hearing dates, judges assigned, and case status. The system leverages the capabilities of a relational DBMS to ensure **data accuracy, consistency, security, and easy accessibility** for authorized users.

The project involves designing an efficient **database schema**, developing user-friendly interfaces, and implementing key functionalities such as **case entry, updates, scheduling, and report generation**. Through this system, we demonstrate how DBMS technology can be effectively applied to enhance the operational efficiency of court management and support the goal of delivering timely justice.

TECHNIQUES

The primary technology used in this project is MySQL, an open-source relational database management system. The following techniques have been implemented:

- **Entity-Relationship Modeling** for data structure visualisation.
- **Normalisation** to organise data efficiently and remove redundancy.
- **SQL Queries** for data manipulation and retrieval.
- **Use of Constraints** like PRIMARY KEY, FOREIGN KEY to enforce relationships.
- **Join operations** to combine data from multiple tables.
- **Aggregate Functions** to summarize and analyze data.
- **Filtering and Sorting** to extract meaningful insights from the dataset.
- **Stored Procedures and Views** (optional enhancements) for automation.

The goal is to simulate a real-time cinema database with multiple users accessing the system concurrently. Though our current system is simplified, it lays the foundation for large-scale enterprise software.

SYSTEM CONFIGURATION

- **Operating System:**
- Windows 10 or higher / Linux / macOS
- **Database Software:**
- MySQL or PostgreSQL
- **RAM:**
- Minimum 4GB
- **Processor:**
- Intel i3 or equivalent and above
- **Other Tools:**
- MySQL Workbench, DBeaver, or phpMyAdmin

INPUT

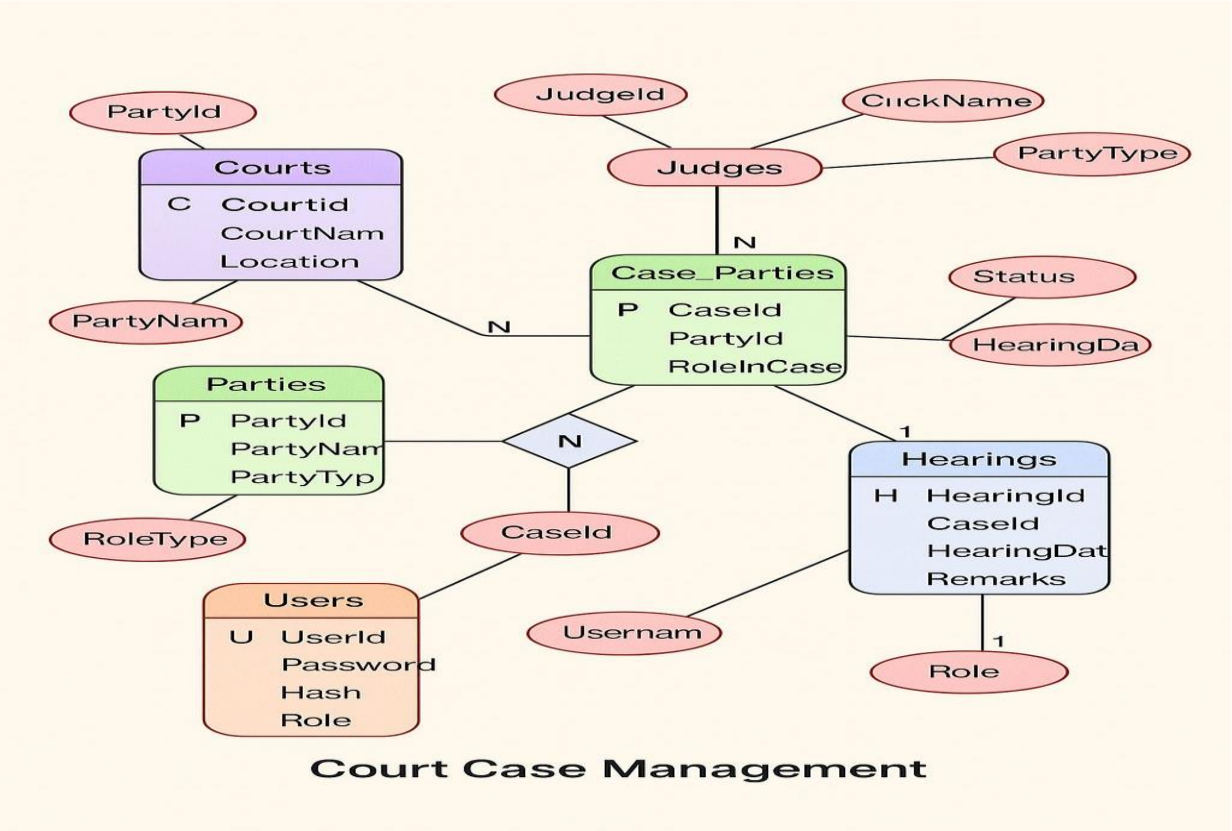
Here's a list of the **table names** and their **details** in the **Court Case Management System** database:

---

1. **Courts Table**
   - **Attributes**: court_id, court_name, location
2. **Judges Table**
   - **Attributes**: judge_id, judge_name, court_id
3. **Parties Table**
   - **Attributes**: party_id, party_name, party_type
4. **Cases Table**
   - **Attributes**: case_id, case_number, case_type, filing_date, status, judge_id
5. **Case_Parties Table**
   - **Attributes**: case_id, party_id, role_in_case
6. **Hearings Table**
   - **Attributes**: hearing_id, case_id, hearing_date, remarks
7. **Users Table**
   - **Attributes**: user_id, username, password_hash, role

This is the summary of the **tables** and their associated **attributes** in the database

ENTITY-RELATIONSHIP DIAGRAM



The Entity-Relationship (ER) diagram outlines the structure and relationships among different entities of the court. It forms the blueprint for the actual database schema.

Each entity has clearly defined attributes and is connected using appropriate relationships like one-to-many and many-to-one, ensuring normalization and avoiding data redundancy.

RELATIONSHIP BETWEEN TABLES

These relationships ensure that the relational database mirrors real-world interactions within a court.

No.	Relationship Type	Parent Table	Child Table	Foreign Key in Child Table	Description
1	One-to-Many	Courts	Judges	`court_id`	Each court can have multiple judges.
2	One-to-Many	Judges	Cases	`judge_id`	A judge can be assigned to multiple cases.
3	One-to-Many	Cases	Hearings	`case_id`	Each case can have multiple hearings.
4	One-to-Many	Cases	Case_Partis	`case_id`	A case can have multiple parties involved.
5	One-to-Many	Parties	Case_Partis	`party_id`	A party can be involved in multiple cases.
6	Standalone	*(None)*	Users	*(N/A)*	Users table is standalone for system login roles (Admin, Clerk, Judge).

TABULAR FORMAT (SCHEMA)

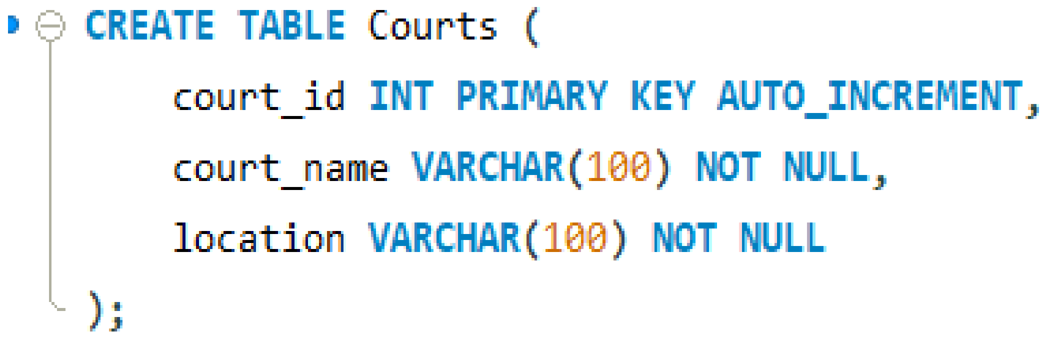
Table Name	Primary Key	Foreign Key	Description
Courts	course_id	—	Stores information about the courts
Judges	judge_id	Courts(court_id)	Stores judges detail
Parties	party_id	—	Parties involved
Cases	case_id	Judges(judge_id)	About the cases

Case_parties	(case_id, party_id)	Case(case_id)
		Parties(party_id)
			Connects cases to parties

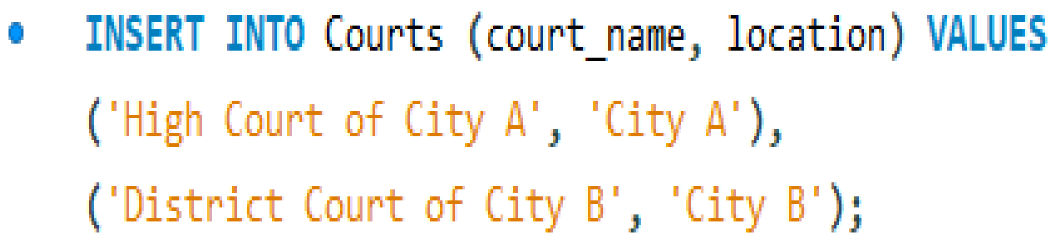
Hearings	hearing_id	Cases(case_id)	About hearings
Users	user_id	—	System users

TABLE CREATION

1. Courts Table:

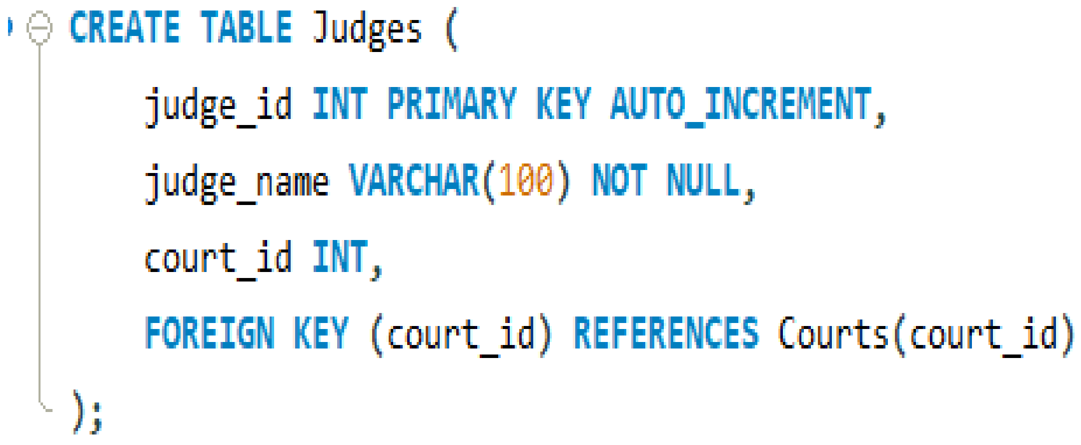


```
CREATE TABLE Courts (
    court_id INT PRIMARY KEY AUTO_INCREMENT,
    court_name VARCHAR(100) NOT NULL,
    location VARCHAR(100) NOT NULL
);
```

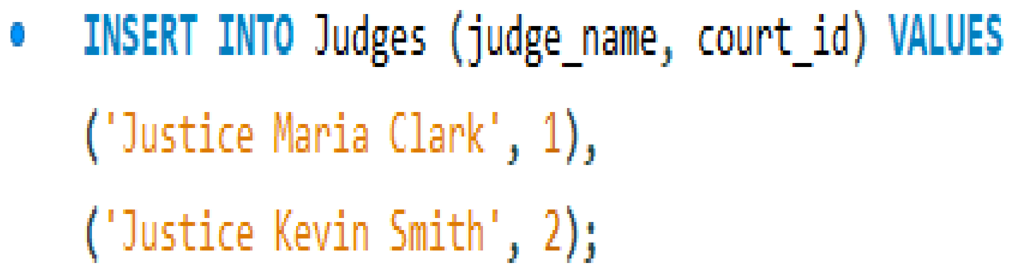


```
• INSERT INTO Courts (court_name, location) VALUES
  ('High Court of City A', 'City A'),
  ('District Court of City B', 'City B');
```

2. Judges Table:

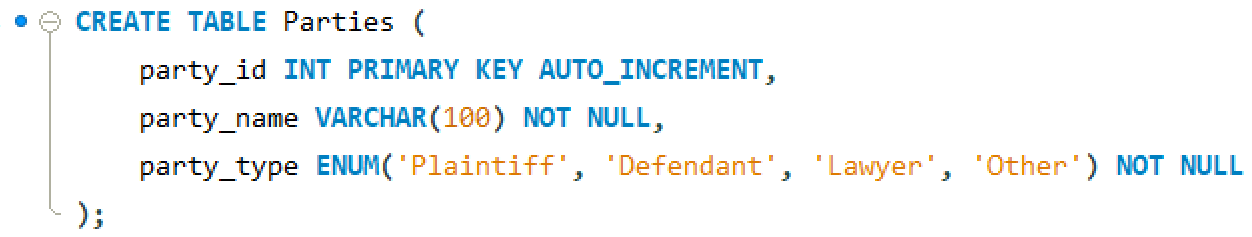


```
CREATE TABLE Judges (
    judge_id INT PRIMARY KEY AUTO_INCREMENT,
    judge_name VARCHAR(100) NOT NULL,
    court_id INT,
    FOREIGN KEY (court_id) REFERENCES Courts(court_id)
);
```

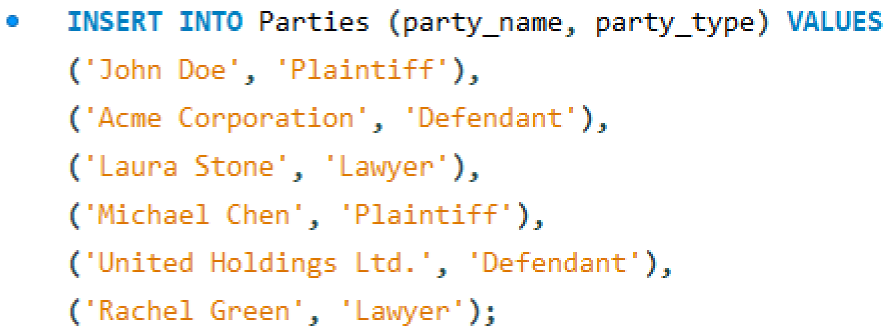


```
• INSERT INTO Judges (judge_name, court_id) VALUES
  ('Justice Maria Clark', 1),
  ('Justice Kevin Smith', 2);
```

3. Parties Table

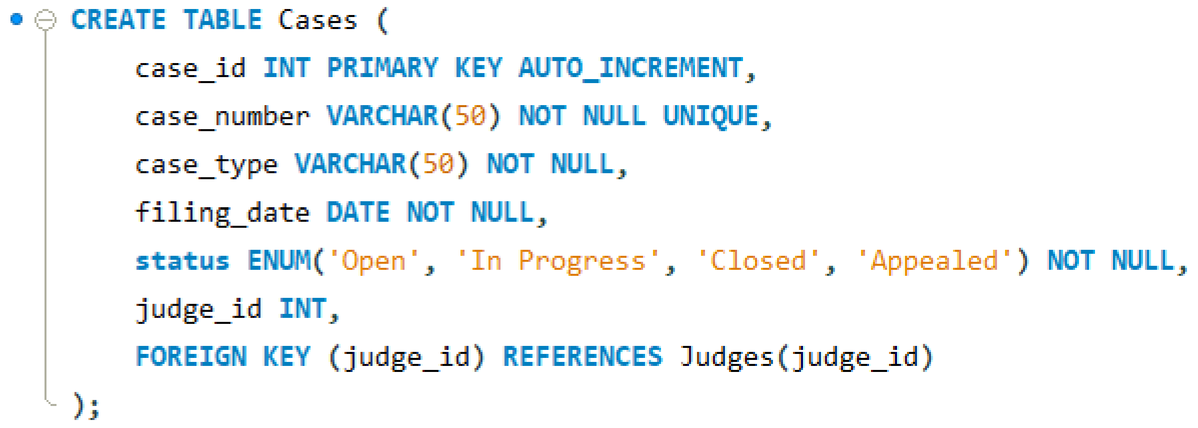


```sql
CREATE TABLE Parties (
    party_id INT PRIMARY KEY AUTO_INCREMENT,
    party_name VARCHAR(100) NOT NULL,
    party_type ENUM('Plaintiff', 'Defendant', 'Lawyer', 'Other') NOT NULL
);
```

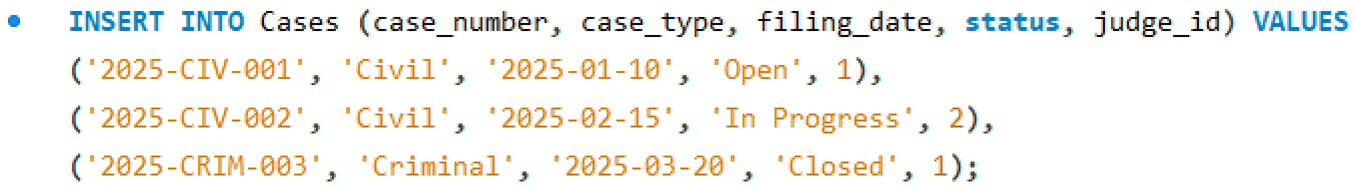


```sql
INSERT INTO Parties (party_name, party_type) VALUES
('John Doe', 'Plaintiff'),
('Acme Corporation', 'Defendant'),
('Laura Stone', 'Lawyer'),
('Michael Chen', 'Plaintiff'),
('United Holdings Ltd.', 'Defendant'),
('Rachel Green', 'Lawyer');
```

4.Cases Table

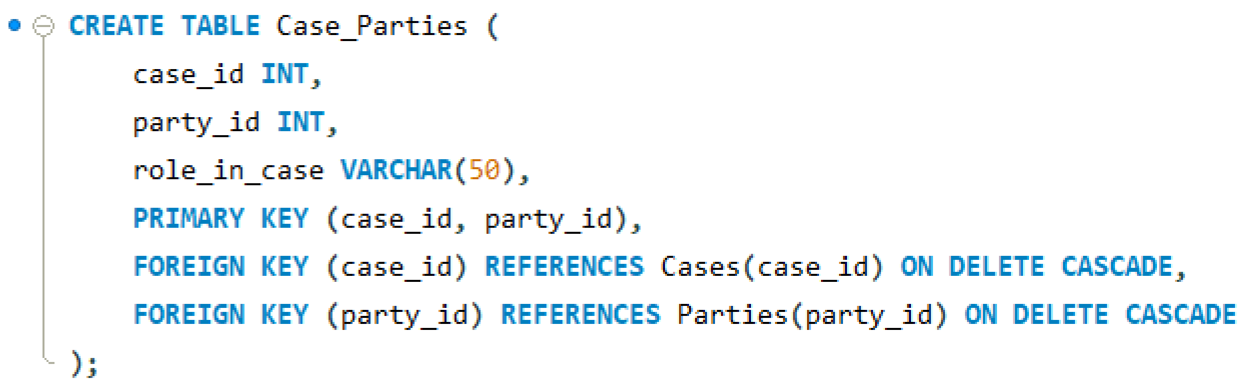


```sql
CREATE TABLE Cases (
    case_id INT PRIMARY KEY AUTO_INCREMENT,
    case_number VARCHAR(50) NOT NULL UNIQUE,
    case_type VARCHAR(50) NOT NULL,
    filing_date DATE NOT NULL,
    status ENUM('Open', 'In Progress', 'Closed', 'Appealed') NOT NULL,
    judge_id INT,
    FOREIGN KEY (judge_id) REFERENCES Judges(judge_id)
);
```

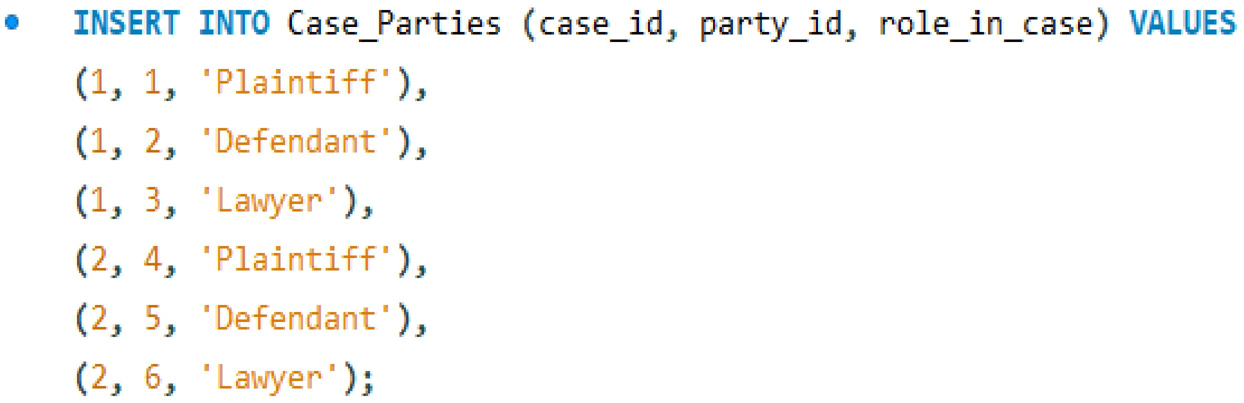


```sql
INSERT INTO Cases (case_number, case_type, filing_date, status, judge_id) VALUES
('2025-CIV-001', 'Civil', '2025-01-10', 'Open', 1),
('2025-CIV-002', 'Civil', '2025-02-15', 'In Progress', 2),
('2025-CRIM-003', 'Criminal', '2025-03-20', 'Closed', 1);
```

5.Case_Parties Table

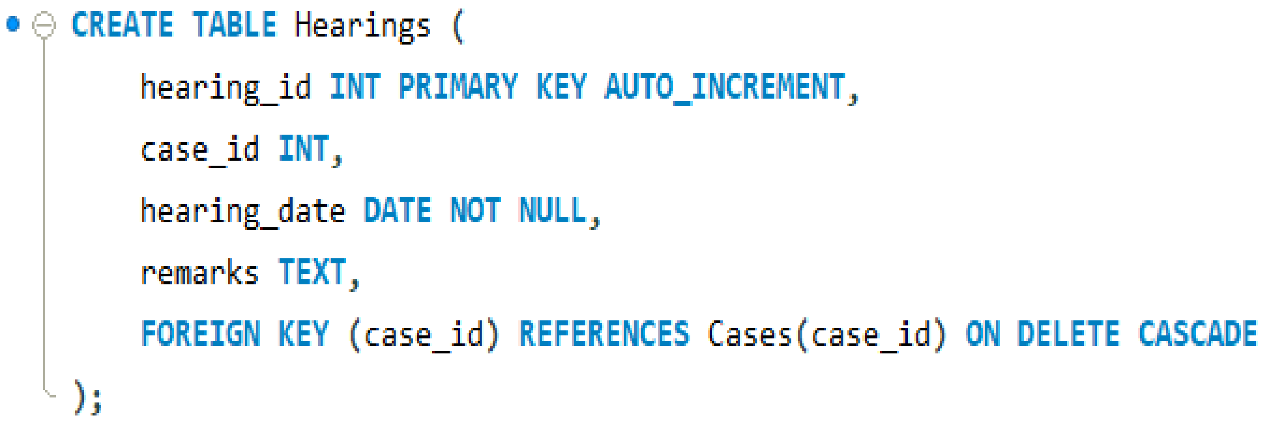


```sql
CREATE TABLE Case_Parties (
    case_id INT,
    party_id INT,
    role_in_case VARCHAR(50),
    PRIMARY KEY (case_id, party_id),
    FOREIGN KEY (case_id) REFERENCES Cases(case_id) ON DELETE CASCADE,
    FOREIGN KEY (party_id) REFERENCES Parties(party_id) ON DELETE CASCADE
);
```

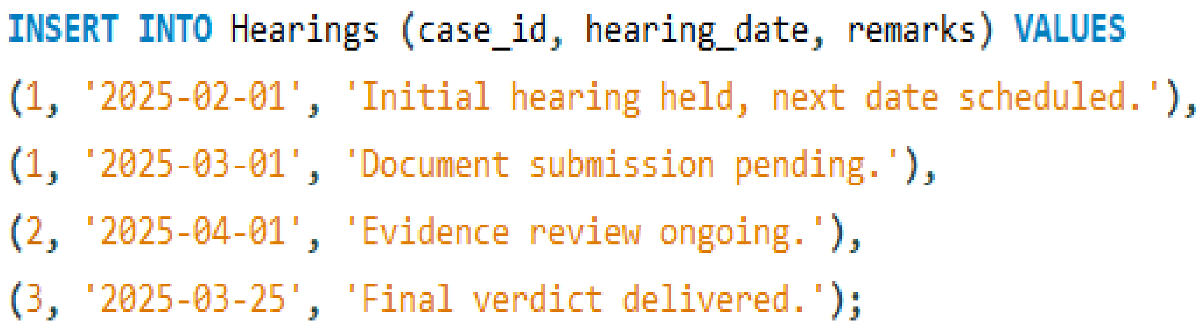


```sql
INSERT INTO Case_Parties (case_id, party_id, role_in_case) VALUES
(1, 1, 'Plaintiff'),
(1, 2, 'Defendant'),
(1, 3, 'Lawyer'),
(2, 4, 'Plaintiff'),
(2, 5, 'Defendant'),
(2, 6, 'Lawyer');
```

6.Hearings Table

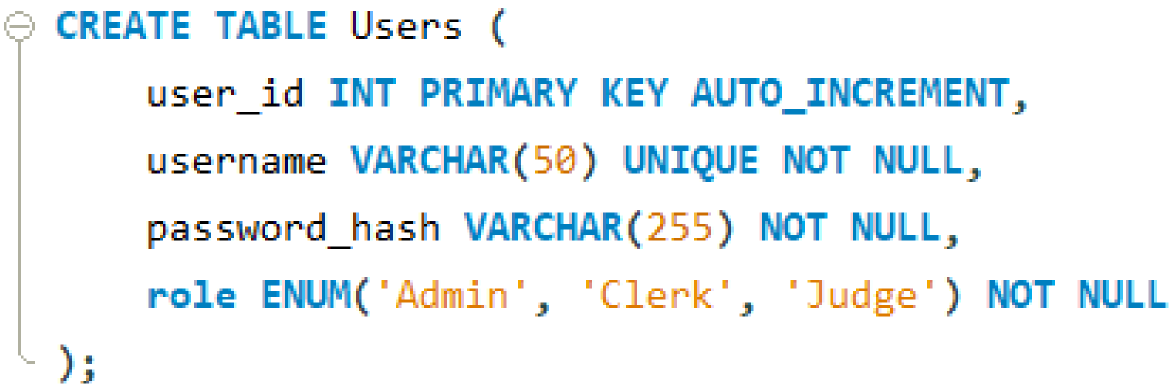


```sql
CREATE TABLE Hearings (
    hearing_id INT PRIMARY KEY AUTO_INCREMENT,
    case_id INT,
    hearing_date DATE NOT NULL,
    remarks TEXT,
    FOREIGN KEY (case_id) REFERENCES Cases(case_id) ON DELETE CASCADE
);
```

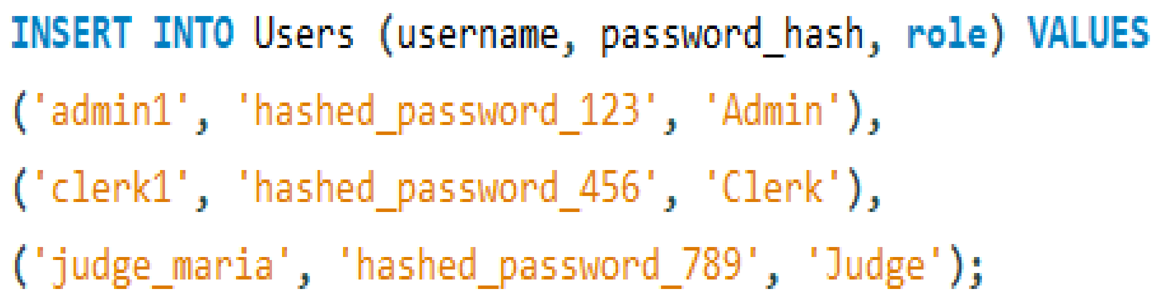


```sql
INSERT INTO Hearings (case_id, hearing_date, remarks) VALUES
(1, '2025-02-01', 'Initial hearing held, next date scheduled.'),
(1, '2025-03-01', 'Document submission pending.'),
(2, '2025-04-01', 'Evidence review ongoing.'),
(3, '2025-03-25', 'Final verdict delivered.');
```

7.Users Table

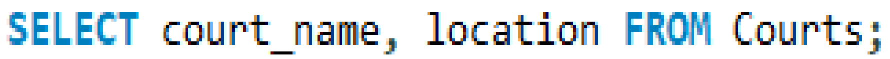


```sql
CREATE TABLE Users (
    user_id INT PRIMARY KEY AUTO_INCREMENT,
    username VARCHAR(50) UNIQUE NOT NULL,
    password_hash VARCHAR(255) NOT NULL,
    role ENUM('Admin', 'Clerk', 'Judge') NOT NULL
);
```

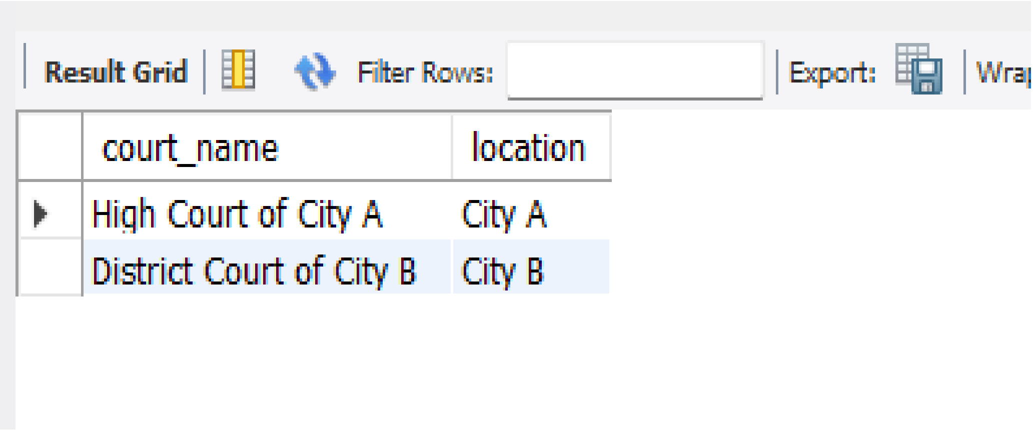


```sql
INSERT INTO Users (username, password_hash, role) VALUES
('admin1', 'hashed_password_123', 'Admin'),
('clerk1', 'hashed_password_456', 'Clerk'),
('judge_maria', 'hashed_password_789', 'Judge');
```

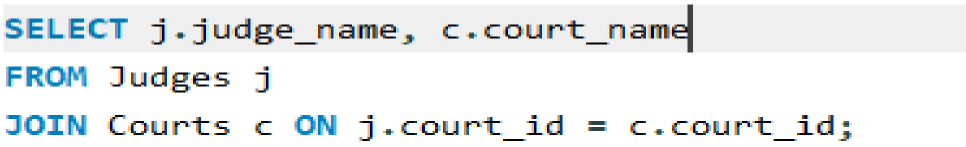
SQL QUERIES (14 Queries)

- 

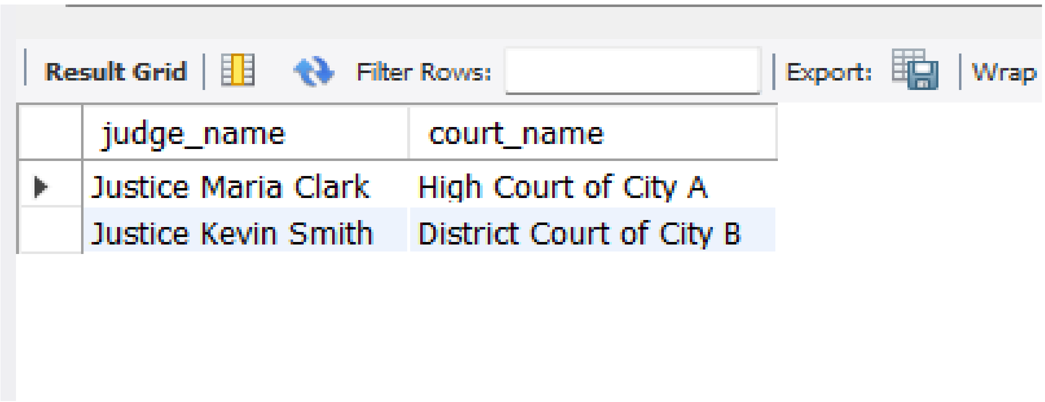
```sql
SELECT court_name, location FROM Courts;
```



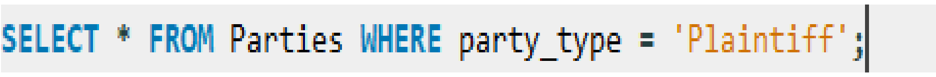
	court_name	location
▶	High Court of City A	City A
	District Court of City B	City B

- 

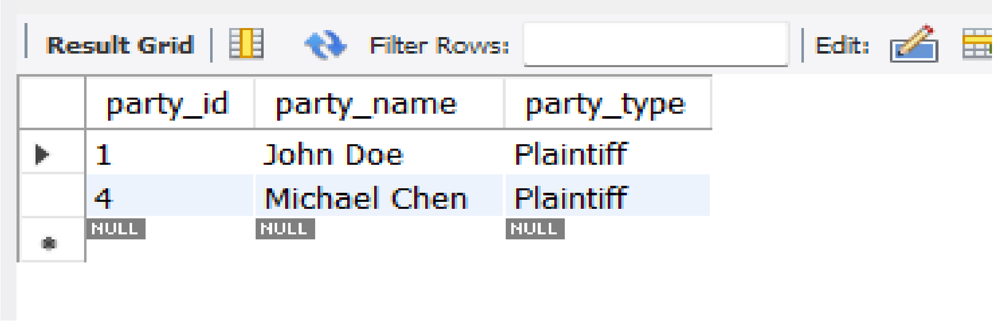
```sql
SELECT j.judge_name, c.court_name
FROM Judges j
JOIN Courts c ON j.court_id = c.court_id;
```



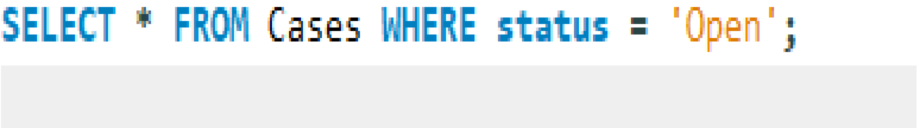
	judge_name	court_name
▶	Justice Maria Clark	High Court of City A
	Justice Kevin Smith	District Court of City B

- 

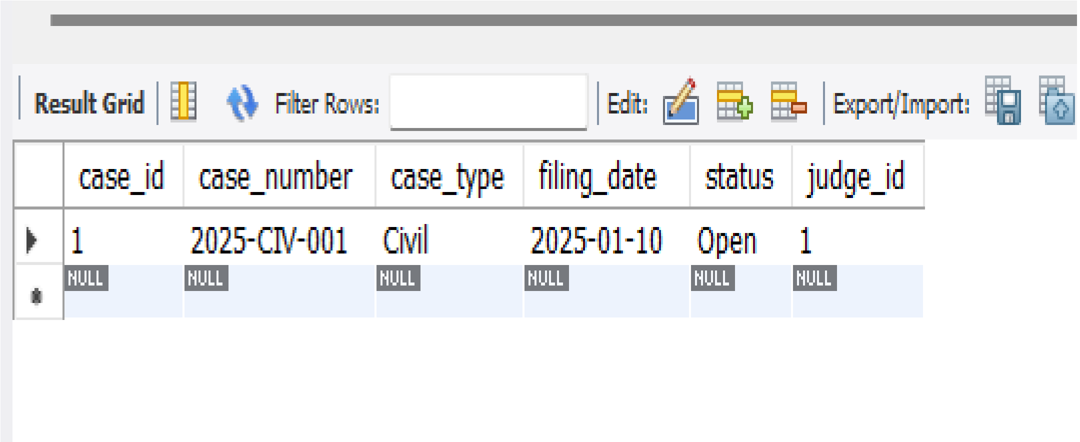
```sql
SELECT * FROM Parties WHERE party_type = 'Plaintiff';
```



party_id	party_name	party_type
1	John Doe	Plaintiff
4	Michael Chen	Plaintiff
NULL	NULL	NULL

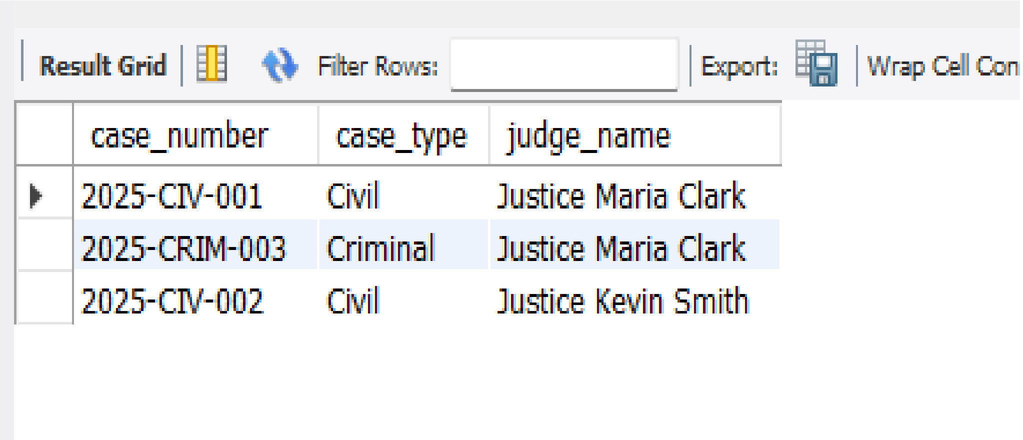
- 

```sql
SELECT * FROM Cases WHERE status = 'Open';
```



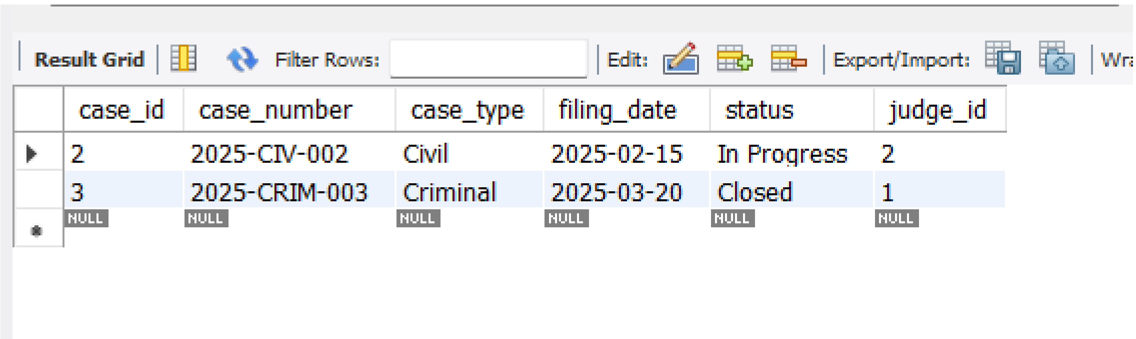
case_id	case_number	case_type	filing_date	status	judge_id
1	2025-CIV-001	Civil	2025-01-10	Open	1
NULL	NULL	NULL	NULL	NULL	NULL

-
  ```
  SELECT c.case_number, c.case_type, j.judge_name
  FROM Cases c
  JOIN Judges j ON c.judge_id = j.judge_id;
  ```

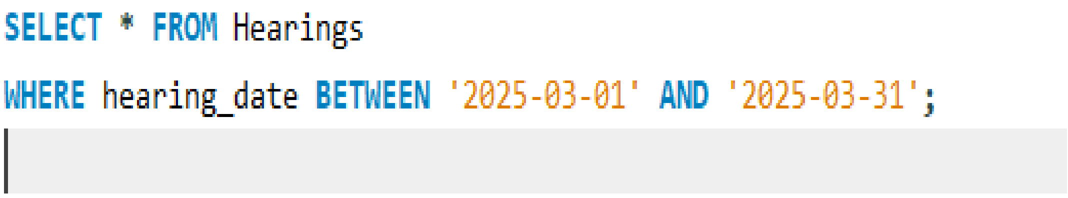


case_number	case_type	judge_name
2025-CIV-001	Civil	Justice Maria Clark
2025-CRIM-003	Criminal	Justice Maria Clark
2025-CIV-002	Civil	Justice Kevin Smith

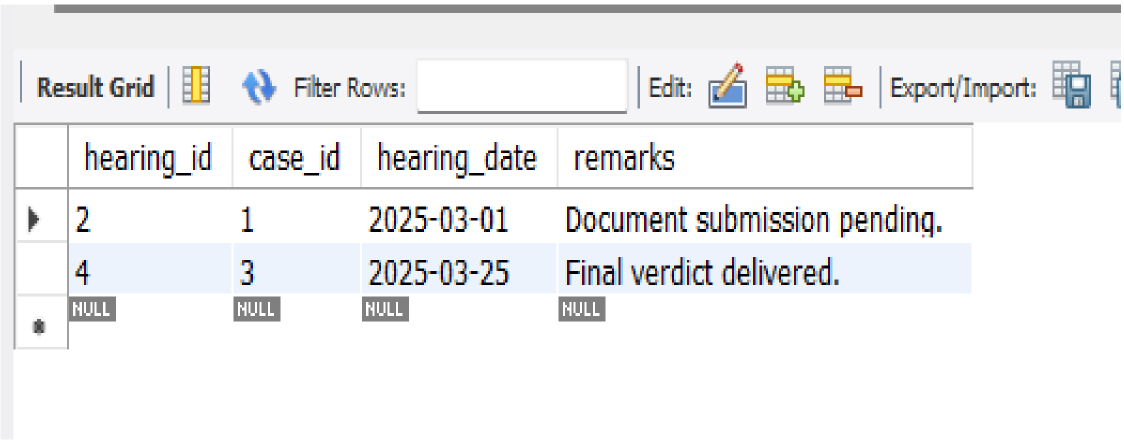
-
  ```
  SELECT * FROM Cases WHERE filing_date > '2025-02-01';
  ```



case_id	case_number	case_type	filing_date	status	judge_id
2	2025-CIV-002	Civil	2025-02-15	In Progress	2
3	2025-CRIM-003	Criminal	2025-03-20	Closed	1
NULL	NULL	NULL	NULL	NULL	NULL

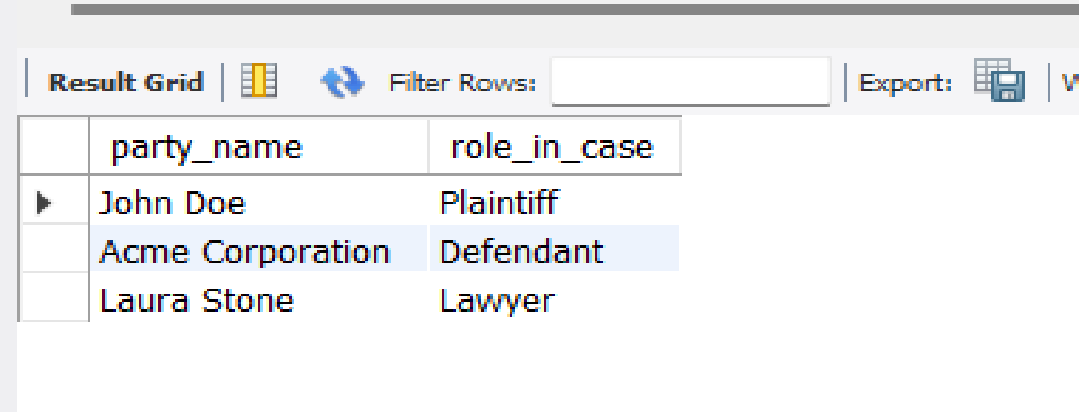


-
  ```
  SELECT * FROM Hearings
  WHERE hearing_date BETWEEN '2025-03-01' AND '2025-03-31';
  ```



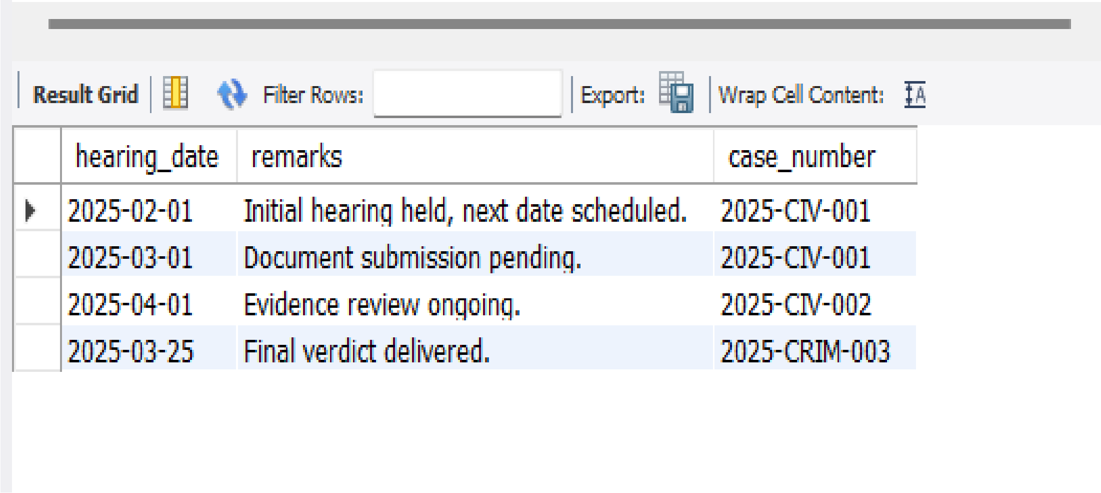
	hearing_id	case_id	hearing_date	remarks
▶	2	1	2025-03-01	Document submission pending.
	4	3	2025-03-25	Final verdict delivered.
*	NULL	NULL	NULL	NULL

-
  ```
  SELECT p.party_name, cp.role_in_case
  FROM Case_Parties cp
  JOIN Parties p ON cp.party_id = p.party_id
  WHERE cp.case_id = 1;
  ```



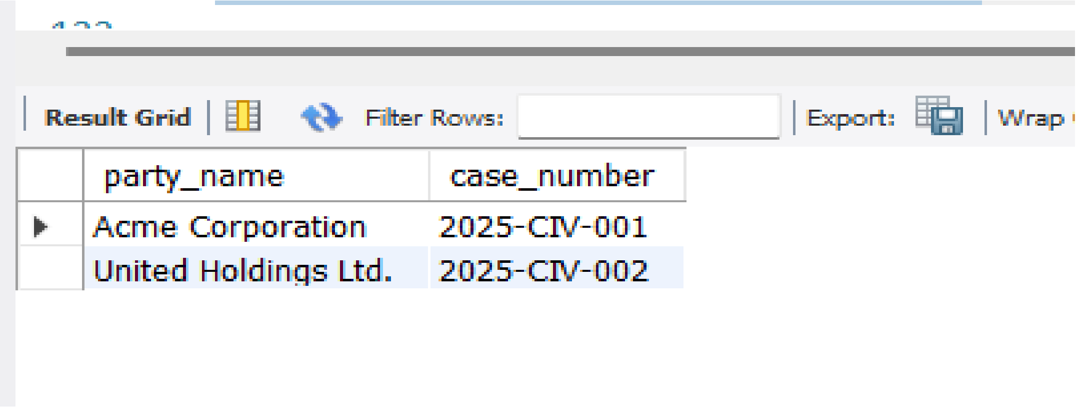
	party_name	role_in_case
▶	John Doe	Plaintiff
	Acme Corporation	Defendant
	Laura Stone	Lawyer

-
  ```
  SELECT h.hearing_date, h.remarks, c.case_number
  FROM Hearings h
  JOIN Cases c ON h.case_id = c.case_id;
  ```



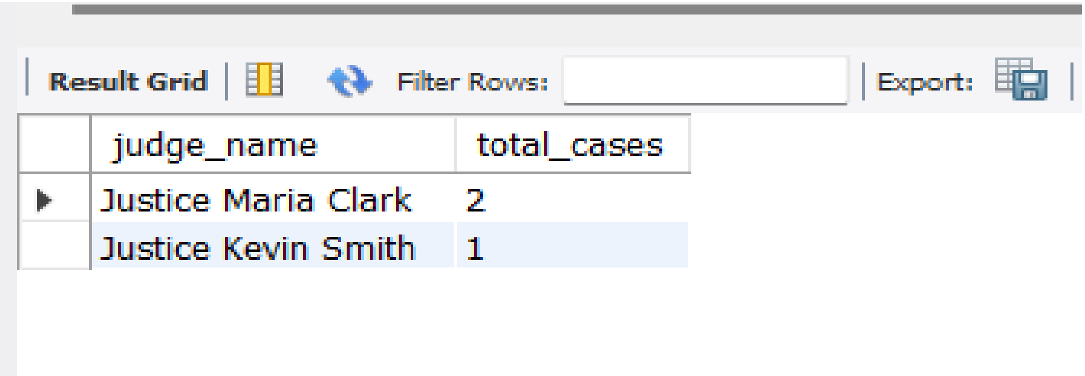
	hearing_date	remarks	case_number
▶	2025-02-01	Initial hearing held, next date scheduled.	2025-CIV-001
	2025-03-01	Document submission pending.	2025-CIV-001
	2025-04-01	Evidence review ongoing.	2025-CIV-002
	2025-03-25	Final verdict delivered.	2025-CRIM-003

-
  ```
  SELECT p.party_name, c.case_number
  FROM Case_Parties cp
  JOIN Parties p ON cp.party_id = p.party_id
  JOIN Cases c ON cp.case_id = c.case_id
  WHERE cp.role_in_case = 'Defendant';
  ```



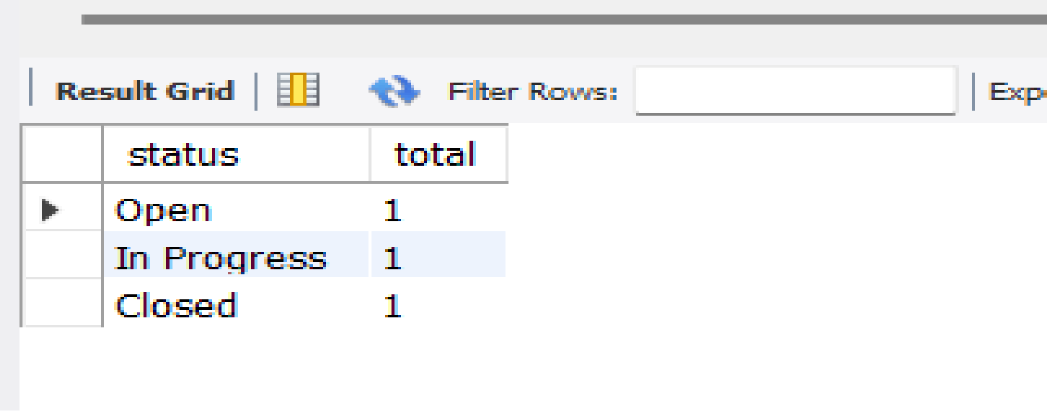
	party_name	case_number
▶	Acme Corporation	2025-CIV-001
	United Holdings Ltd.	2025-CIV-002

-
  ```
  SELECT j.judge_name, COUNT(c.case_id) AS total_cases
  FROM Judges j
  LEFT JOIN Cases c ON j.judge_id = c.judge_id
  GROUP BY j.judge_name;
  ```

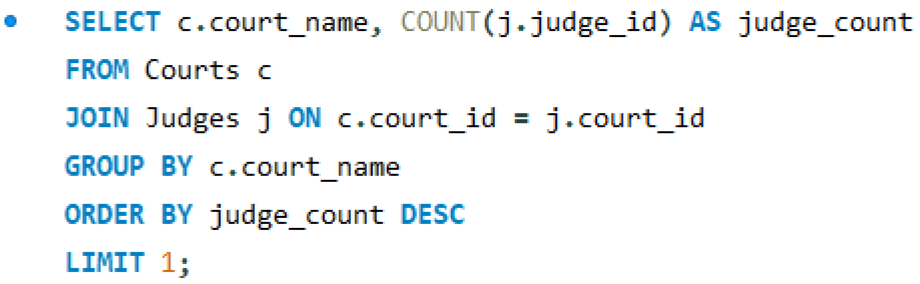


judge_name	total_cases
Justice Maria Clark	2
Justice Kevin Smith	1

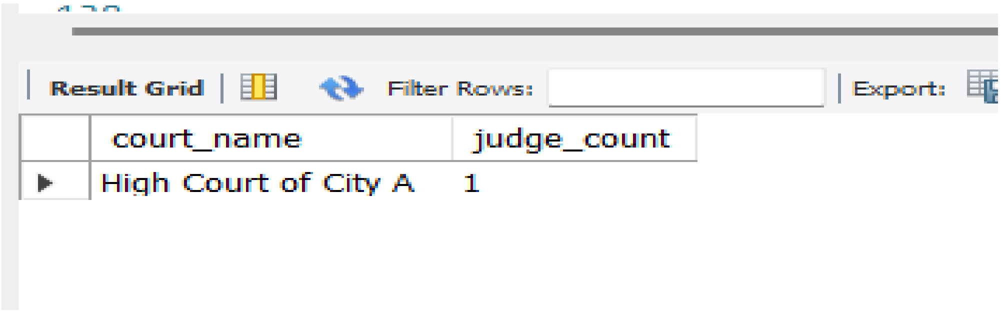
-
  ```
  SELECT status, COUNT(*) AS total
  FROM Cases
  GROUP BY status;
  ```



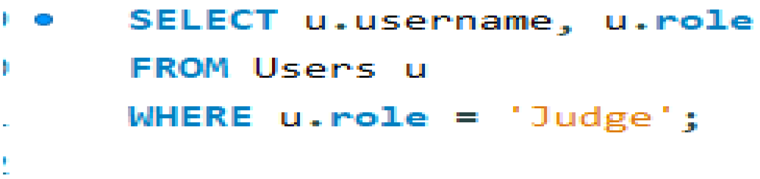
status	total
Open	1
In Progress	1
Closed	1



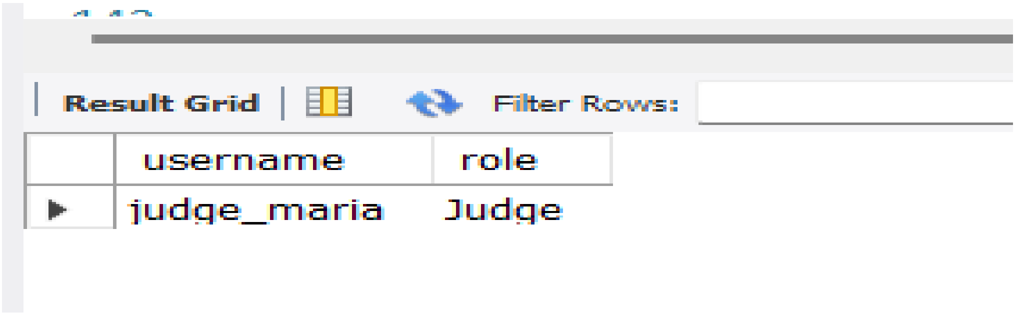
```
SELECT c.court_name, COUNT(j.judge_id) AS judge_count
FROM Courts c
JOIN Judges j ON c.court_id = j.court_id
GROUP BY c.court_name
ORDER BY judge_count DESC
LIMIT 1;
```



court_name	judge_count
High Court of City A	1



```
SELECT u.username, u.role
FROM Users u
WHERE u.role = 'Judge';
```



username	role
judge_maria	Judge

SUMMARY

The Court Case Management System is a structured relational database designed to efficiently manage and organize judicial operations, including courts, judges, legal cases, involved parties, hearings, and system users. It comprises seven interrelated tables that capture essential data such as court details, judge assignments, case information, party roles, and scheduled hearings. The system maintains data integrity through the use of primary and foreign keys, and supports role-based access for users like admins, clerks, and judges. With well-defined relationships—such as courts to judges, judges to cases, and cases to parties and hearings—the database allows seamless tracking of legal proceedings from filing to resolution. It also provides the capability to generate reports and insights, such as the number of cases per judge or hearing schedules for specific timeframes. This system enhances the efficiency, transparency, and security of court operations, making it a vital tool for digital transformation in judicial administration.

CONCLUSION

The Court Case Management System provides a reliable, efficient, and structured approach to managing judicial processes through the use of a well-designed relational database. Here is a point-wise conclusion summarizing key aspects:

🔍 Key Observations

- **The relational database structure effectively models real-world court operations.**
- **Primary and foreign key constraints ensure data consistency and integrity across tables.**
- **The Case_Parties junction table allows flexible many-to-many relationships between cases and parties.**
- **Role-based access through the Users table supports basic security and user management.**
- **SQL queries enable efficient data retrieval, reporting, and filtering of case and hearing information.**

⚠ Limitations

- **Lacks a user interface (UI) for non-technical users to interact with the system.**
- **No support for multi-judge panels or tracking of appeal hierarchies.**
- **Hearing notifications and scheduling alerts are not implemented.**
- **User authentication and password security are basic and would require enhancement.**
- **Document management (e.g., uploading legal documents or evidence) is not included.**

🚀 Future Scope

- **Develop a web-based or mobile UI for better accessibility and usability.**
- **Implement automated hearing reminders via SMS/email for parties and judges.**
- **Add support for multiple judges per case and appeals tracking.**
- **Enhance user authentication with secure login, role-based permissions, and audit trails.**
- **Integrate a document management system for uploading case files**

and legal documents.

- **Expand analytics features for monitoring case loads, judge performance, and court efficiency.**

- **Enable integration with external judicial databases or government systems for real-time updates.**