**UNIVERSITY OF DAYTON**

**COURT CASE MANAGEMENT SYSTEM**

**(Final Report)**

DATA ADDED, SQL QUERIES CREATED

AND

TABLES CREATED

By

SANDEEP SRIVATSAVA SINGARAJU (1017432390)

SAI ARAVIND NAKARIKANTI (1017453510)

SOURAB DEB (1017442430)

Under the Guidance of

CEMIL KIRBAS

**DATABASE MANAGEMENT SYSTEMS**

**CPS 542**

DEPARTMENT OF COMPUTER SCIENCE

Table of Contents

Introduction 3

Entities 3

Attributes 4

Relationships 5

ER Diagram 7

Relational Schema 9

Normalization 11

Logical Model 13

Physical Model 14

Table Views 17

Data Manipulation 21

Conclusion 26

**Court Case Database Management Design**

**Introduction:**

A court database is a collection of data about court cases and legal proceedings that is usually kept up to date by a government organization. It might contain information about the parties involved, the accusations or charges, the verdict, and any supporting records or transcripts. Attorneys, academics, journalists, and the general public can access information regarding court cases and the legal system through court databases. They might be accessible online, where you can search for them using the case number, the parties’ names, or other terms. Access to court schedules, dockets, and other details regarding forthcoming court appearances is also available through some court databases.

**Entities:**

In DBMS (Database Management System), an entity is a real-world object or concept that has attributes that describe its characteristics. Entities can be anything from physical objects, such as people or products, to abstract concepts, such as accounts or transactions.

In a database, an entity is typically represented as a table, with each row representing an instance of that entity and each column representing an attribute of that entity. For example, a client entity in a database might have attributes such as client name, address, phone number, and email.

Entities are important in database design because they help to organize and structure the data in a way that makes sense for the business or application. By identifying the entities and their attributes, designers can create a database schema that is optimized for storing and retrieving data efficiently.

In addition to attributes, entities can also have relationships with other entities. For example, a client entity might have a relationship with an attorney entity, indicating that a client can place multiple orders. These relationships are also important in database design because they help to define the overall structure of the database and ensure that the data is consistent and accurate. In this database we have 7 entities and they are:

* Client
* Case
* Attorney
* Court
* Judge
* Jury
* Bar Council

**Attributes:**

In database management systems, an attribute refers to a characteristic or property of an entity. Attributes are used to describe the entities that make up a database, and they provide the details and information that can be stored, retrieved, and analysed within the database.

Attributes can be of different types, depending on the kind of data they represent. Some common types of attributes include:

* Numeric attributes: These attributes represent numeric values such as integers or floating-point numbers. Examples include price, quantity, and age.
* Text attributes: These attributes represent textual data, such as names, addresses, and descriptions.
* Boolean attributes: These attributes represent a binary value, such as true/false or yes/no.
* Date/time attributes: These attributes represent dates and times, such as birthdates, order dates, or delivery times.

Attributes can be further classified as primary or foreign key attributes, which help define relationships between different entities in a database. A primary key is a unique identifier for an entity, while a foreign key is a reference to the primary key of another entity.

Attributes play a crucial role in the design and development of a database. They provide the structure for storing data and enable efficient searching, sorting, and querying of data within the database. By carefully selecting and organizing attributes, database designers can create a database that is optimized for the specific needs of the organization or application. In this database the attributes are as follows:

* Client – C\_ID, C\_Lname, C\_Fname, C\_Address, C\_Sex
* Case – Case\_ID, Case\_Desp, Attorney\_ID
* Attorney – A\_ID, A\_Name, A\_Email, A\_Address
* Court – Co\_ID, Co\_Name, Co\_Address, Co\_City, Co\_Zipcode
* Judge – J\_ID, J\_Name, J\_PhoneNo, J\_Address
* Jury – No\_of\_Jurors, Juries\_Verdicts
* Bar Council – A\_General, A\_ID, Bar\_council\_ID

**Relationships:**

In database management systems, a relationship is an association between two or more entities that can be used to represent the real-world connections and interactions between those entities. Relationships allow data to be organized in a meaningful way and enable queries to be performed that can retrieve related data from multiple tables.

There are three main types of relationships that can exist between entities in a database:

One-to-one relationship: In this type of relationship, one instance of an entity is associated with only one instance of another entity. For example, in a database of employees and their office locations, each employee can have only one office location, and each office location can be associated with only one employee.

One-to-many relationship: In this type of relationship, one instance of an entity can be associated with multiple instances of another entity, but each instance of the other entity can be associated with only one instance of the first entity. For example, in a database of customers and their orders, each customer can place multiple orders, but each order is associated with only one customer.

Many-to-many relationship: In this type of relationship, multiple instances of one entity can be associated with multiple instances of another entity. For example, in a database of students and classes, each student can enroll in multiple classes, and each class can have multiple students enrolled.

To represent relationships between entities in a database, foreign keys are often used. A foreign key is a field in one table that refers to the primary key of another table, creating a link between the two tables. The foreign key allows related data to be linked and retrieved from multiple tables when performing queries.

Overall, relationships are an essential concept in database design as they help to organize and structure data in a meaningful way, enabling efficient retrieval and analysis of data.

* registers: Client registers case
* choose: Client chooses Attorney
* retains: Attorney retains case
* members\_of: Attorneys are members of bar council
* assigned\_to: Case is assigned to court
* works\_for: Judge works for court
* passes\_to: Judge passes the case to Jury

**ER DIAGRAM**

An Entity-Relationship (ER) diagram is a graphical representation of entities and their relationships to each other in a database. It is used to model and design the database schema and can help to visualize the relationships between entities.

An ER diagram consists of three main components:

Entities: Entities are objects or concepts that exist independently and have distinct attributes or characteristics. In a database, an entity is typically represented as a table.

Relationships: Relationships define the associations or connections between entities. There are three types of relationships: one-to-one, one-to-many, and many-to-many.

Attributes: Attributes describe the properties or characteristics of an entity. Each entity has one or more attributes.

ER diagrams are typically created using symbols and notation to represent entities, relationships, and attributes. The symbols used may vary depending on the specific notation system being used, but some common symbols include:

Entity: represented by a rectangle with the entity name inside

Relationship: represented by a diamond shape connecting two entities

Attribute: represented by an oval shape connected to the entity rectangle

By creating an ER diagram, designers can visualize the relationships between entities and design a database schema that is efficient, flexible, and scalable.  
The ER Diagram for Court case management database is as follows

A picture containing text, control

Description automatically generated

**Relational Schema:**

A relational schema is a blueprint that describes the structure and organization of a relational database. It defines the tables, fields, and relationships between tables that make up the database, and it provides a formal description of the data that is stored in the database.

A relational schema consists of several components, including:

Tables: Tables are the main data containers in a relational database. They are used to store data in rows and columns, with each column representing a field and each row representing a record.

Fields: Fields are individual data elements that make up a table. They are also called columns or attributes, and they represent specific characteristics or properties of the data being stored.

Primary keys: A primary key is a field or combination of fields that uniquely identify each record in a table. It is used to ensure that each record is unique and to create relationships between tables.

Foreign keys: A foreign key is a field in one table that refers to the primary key of another table. It is used to create relationships between tables and ensure data consistency.

Relationships: Relationships define the associations or connections between tables in a database. There are three main types of relationships: one-to-one, one-to-many, and many-to-many.

A relational schema can be represented graphically using an Entity-Relationship (ER) diagram, which uses symbols to represent tables, fields, and relationships between tables. The schema can also be expressed in a formal language, such as SQL (Structured Query Language), which is used to create, modify, and retrieve data from relational databases.

Overall, a relational schema is a key component of database design and helps to ensure that data is organized, structured, and stored in a way that is efficient and effective for the specific needs of the organization or application.

**Relational Schema for the Database:**

Graphical user interface, application

Description automatically generated

**Normalization:**

To normalize the given database, we need to apply normalization rules to eliminate data redundancy and improve data consistency.

1. First Normal Form (1NF)

To achieve 1NF, we need to make sure that each attribute in a table contains only atomic values. In the given database, all attributes appear to be atomic, so the database is already in 1NF.

2. Second Normal Form (2NF)

To achieve 2NF, we need to make sure that every non-key attribute is fully dependent on the primary key. In the given database, we can see that the Case table has a partial dependency on the primary key, Attorney\_ID. To remove this dependency, we can create a new table for Attorney and move the Attorney-related attributes to that table. This will give us the following tables:

Client (C\_ID, C\_Lname, C\_Fname, C\_Address, C\_Sex)

Attorney (A\_ID, A\_Name, A\_Email, A\_Address)

Case (Case\_ID, Case\_Desp, Attorney\_ID)

Court (Co\_ID, Co\_Name, Co\_Address, Co\_City, Co\_Zipcode)

Judge (J\_ID, J\_Name, J\_PhoneNo, J\_Address)

Jury (No\_of\_Jurors, Juries\_Verdicts)

Bar\_Council (A\_General, A\_ID, Bar\_council\_ID)

3. Third Normal Form (3NF)

To achieve 3NF, we need to make sure that there are no transitive dependencies in the tables. In the given database, we can see that the Attorney table has a transitive dependency on the Bar\_Council table through the A\_ID attribute. To remove this dependency, we can create a new table for the relationship between Attorney and Bar\_Council. This will give us the following normalized tables:

Client (C\_ID, C\_Lname, C\_Fname, C\_Address, C\_Sex)

Attorney (A\_ID, A\_Name, A\_Email, A\_Address, Bar\_council\_ID)

Attorney\_Bar\_Council (A\_ID, Bar\_council\_ID, A\_General)

Case (Case\_ID, Case\_Desp, Attorney\_ID)

Court (Co\_ID, Co\_Name, Co\_Address, Co\_City, Co\_Zipcode)

Judge (J\_ID, J\_Name, J\_PhoneNo, J\_Address, Co\_ID)

Jury (No\_of\_Jurors, Juries\_Verdicts, Case\_ID)

Bar\_Council (Bar\_council\_ID)

In the above tables, we have removed the transitive dependency between Attorney and Bar\_Council by creating a new table, Attorney\_Bar\_Council, that represents the relationship between them. Now, all tables are in 3NF, and the database is normalized.

**Logical Model**

From the Normalized Schema we can design the logical design in MySQL Work Bench:

Diagram

Description automatically generated

**Physical Model of Court case Management System done in MYSQL:**

**Graphical user interface, text, application

Description automatically generatedGraphical user interface, application, table

Description automatically generatedGraphical user interface, application

Description automatically generatedGraphical user interface

Description automatically generated with medium confidenceGraphical user interface

Description automatically generated with medium confidenceGraphical user interface

Description automatically generated with medium confidenceGraphical user interface, application, table

Description automatically generated**

**MYSQL**

In MySQL, users can create databases, tables, views, stored procedures, triggers, and functions. They can also insert, update, delete, and query data using SQL (Structured Query Language) commands. MySQL also supports transactions, which are sequences of database operations that are executed as a single unit of work to ensure data consistency.

MySQL has a number of advantages, including its ease of use, scalability, reliability, and performance. It is also widely supported by hosting providers and has a large user community, which makes it easy to find resources and support for the platform. Additionally, MySQL is open source, which means that it is free to use and can be customized to meet the specific needs of users.

**TABLE VIEWS:**

1. Client Table

Graphical user interface, text, email

Description automatically generated

1. Case Table

Graphical user interface, text

Description automatically generated

1. Court Table

Graphical user interface, text, email

Description automatically generated

1. Attorney Table

Graphical user interface, text, email

Description automatically generated

1. Bar Council Table

Graphical user interface, text, email

Description automatically generated

1. Judge Table

Graphical user interface, text, email

Description automatically generated

1. Jury Table

Graphical user interface, text, email

Description automatically generated

**Data Manipulation**

Data manipulation refers to the process of modifying, organizing, and analyzing data in a database. It involves performing various operations such as adding, deleting, updating, and retrieving data from the database.

Some common data manipulation operations are:

1. Insertion: This operation is used to add new data to the database. The INSERT statement is used for this purpose.
2. Deletion: This operation is used to remove data from the database. The DELETE statement is used for this purpose.
3. Updating: This operation is used to modify existing data in the database. The UPDATE statement is used for this purpose.
4. Retrieval: This operation is used to retrieve data from the database. The SELECT statement is used for this purpose.
5. Sorting: This operation is used to sort data in the database based on a specific criterion. The ORDER BY clause is used for this purpose.
6. Filtering: This operation is used to retrieve specific data from the database based on a condition. The WHERE clause is used for this purpose.
7. Aggregation: This operation is used to perform calculations on the data in the database. The GROUP BY and aggregate functions (e.g. SUM, AVG, MAX, MIN) are used for this purpose.
8. **INSERT**

**Graphical user interface, text, application, email

Description automatically generated**

1. **DELETE**

**Graphical user interface, text, application, email

Description automatically generated**

1. **UPDATE**

**Graphical user interface, text, email

Description automatically generated**

1. **RETRIEVAL**

**Graphical user interface, text, application, email

Description automatically generated**

1. **SORT**

**Graphical user interface, text, application, email

Description automatically generated**

1. **FILTER**

**Graphical user interface, text

Description automatically generated**

1. **AGGREGATION**

**Graphical user interface, text, application, email

Description automatically generated**

**Conclusion**

The Court Case Management System project offers details on the various judges employed by the Court, the number of clients served by various attorneys, the issues faced by clients, and a summary of their cases. The following are some benefits of this project:

* Less redundancy.
* Information retrieval is simple.
* Updates and removals are easily accomplished.