1. **Title Slide**:
   * Include a title with the name of your project or database and add your name and any relevant affiliations. In this case now say that this database in for a xxxxx website. Xxxxx being the name of the website.
2. **Introduction**:
   * Briefly introduce the purpose of the database. Mention its importance and how it addresses specific needs or problems.

The purpose of this database is to facilitate the management and organization of properties, users, locations, property types, and professionals within a web-based application focused on real estate or property management. Here's a breakdown of its main purposes:

1. Property Management:

Store detailed information about individual properties, including their addresses, prices, descriptions, images, and statuses.

Categorize properties based on types (e.g., house, apartment) to facilitate search and categorization functionalities.

Track the availability and status of each property (e.g., available, sold) for efficient property management.

1. User Management:

Maintain user accounts with essential details such as names, email addresses, and passwords.

Establish relationships between users and their respective professions or roles within the real estate domain.

Enable user authentication and authorization to access and interact with the application's features and functionalities.

1. Location Management:

Store geographic data about various locations, including cities, states, countries, and their associated zip codes.

Provide location-related information to help users find properties based on their preferred locations or search criteria.

1. Professional Management:

Define and categorize professionals involved in real estate transactions, such as real estate agents, brokers, or property managers.

Facilitate collaboration between users and professionals by providing information about their respective roles and expertise.

Overall, the database serves as a central repository for managing and organizing data related to properties, users, locations, property types, and professionals, thereby supporting the functionalities and operations of the web-based real estate or property management application. It enables efficient data storage, retrieval, and manipulation, contributing to enhanced user experience and streamlined property transactions.

1. **Database Overview**:
   * Provide an overview of the tables in your database. Mention the primary keys of each table and their significance.
   * In this case the tables are;
2. User Table:

Keys:

Primary Key: UserId

Functions:

Stores information about users such as their first name, last name, password, email, and profession.

Establishes a relationship with the Profession table through the ProfId foreign key.

1. PropertyType Table:

Keys:

Primary Key: TypeId

Functions:

Defines various types of properties.

Enables categorization of properties based on their type.

1. Property Table:

Keys:

Primary Key: PropertyId

Functions:

Represents individual properties with attributes like address, price, description, image URL, and status.

Links each property to a specific PropertyType through the TypeId foreign key.

Associates properties with their respective owners (users) via the UserId foreign key.

Establishes a connection with the Location table through the ZipCode foreign key.

1. Location Table:

Keys:

Primary Key: ZipCode

Functions:

Stores location-related information such as city, state, and country.

Linked to properties through the ZipCode foreign key.

1. Professional Table:

Keys:

Primary Key: ProfId

Functions:

Represents professionals associated with the properties.

Defines different types of professionals (e.g., real estate agents, brokers) through the ProfType attribute.

Each table serves a specific purpose in the database schema, facilitating the organization and management of data related to users, properties, property types, locations, and professionals. ForeignKey constraints establish relationships between these tables to maintain data integrity and enforce referential integrity.

1. **Entity-Relationship Diagram (ERD)**:
   * Display an ERD to visually represent the relationships between tables. Clearly illustrate how each table is connected through foreign key relationships which I will mention in another point below. Explain the cardinality (one-to-one, one-to-many, many-to-many) of these relationships. The ERD design in this case is a **chen’s notation** design.
2. **Table Definitions**:
   * For each table, present its structure, including column names, data types, and any constraints. Explain the purpose of each table and its role within the database.
3. User Table:

Structure:

UserId (Primary Key, INT): Unique identifier for each user.

FirstName (VARCHAR(50)): First name of the user.

LastName (VARCHAR(50)): Last name of the user.

Password (VARCHAR(100)): Password associated with the user's account.

Email (VARCHAR(100)): Email address of the user.

ProfId (INT): Foreign key referencing the Professional table.

Purpose:

The User table stores information about users who interact with the system.

Role:

It facilitates user management by storing user details such as name, email, and password.

Establishes a relationship with the Professional table to link users with their respective professions.

1. PropertyType Table:

Structure:

TypeId (Primary Key, INT): Unique identifier for each property type.

TypeName (VARCHAR(50)): Name of the property type (e.g., house, apartment).

Purpose:

Defines various types of properties available in the system.

Role:

Helps categorize properties based on their type, aiding in property classification and search functionalities.

1. Property Table:

Structure:

PropertyId (Primary Key, INT): Unique identifier for each property.

Address (VARCHAR(255)): Address of the property.

Price (DECIMAL(10, 2)): Price of the property.

Description (TEXT): Description of the property.

ImageURL (VARCHAR(255)): URL of the property image.

Status (VARCHAR(50)): Status of the property (e.g., available, sold).

TypeId (INT): Foreign key referencing the PropertyType table.

UserId (INT): Foreign key referencing the User table.

ZipCode (INT): Foreign key referencing the Location table.

Purpose:

Represents individual properties available for sale or rent.

Role:

Stores detailed information about each property, including its address, price, description, and status.

Establishes relationships with the PropertyType, User, and Location tables to link properties with their respective types, owners, and locations.

1. Location Table:

Structure:

ZipCode (Primary Key, INT): Unique identifier for each location.

City (VARCHAR(100)): City name.

State (VARCHAR(100)): State name.

Country (VARCHAR(100)): Country name.

Purpose:

Stores location-related information such as city, state, and country.

Role:

Provides geographic data necessary for property listings, enabling users to search for properties based on location.

1. Professional Table:

Structure:

ProfId (Primary Key, INT): Unique identifier for each professional.

ProfType (VARCHAR(100)): Type of profession (e.g., real estate agent, broker).

Purpose:

Represents professionals associated with properties (e.g., real estate agents, brokers).

Role:

Defines different types of professionals involved in property transactions, facilitating collaboration between users and professionals within the system.

Each table plays a specific role within the database, storing relevant data and establishing relationships to ensure efficient data management and retrieval for the web database application.

1. **Normalization**:

Normalization is a database design technique used to organize tables and minimize data redundancy while ensuring data integrity. The normalization process involves breaking down a large table into smaller, more manageable tables and establishing relationships between them. Here's how the tables in the provided database design are normalized:

1. First Normal Form (1NF):

Each table has a primary key that uniquely identifies each record (UserId for User table, TypeId for PropertyType table, PropertyId for Property table, ZipCode for Location table, and ProfId for Professional table).

Each column contains atomic values, meaning each cell in the table holds a single value.

1. Second Normal Form (2NF):

In the User table, there's a partial dependency between ProfId and other attributes (FirstName, LastName, Password, Email). To address this, ProfId is removed from the User table, and a separate Professional table is created. This ensures that each attribute is fully dependent on the primary key (UserId).

The Property table contains attributes (Address, Price, Description, ImageURL, Status) that are fully dependent on the primary key (PropertyId), satisfying 2NF.

1. Third Normal Form (3NF):

In the Property table, the UserId foreign key establishes a relationship between properties and users. However, the User table already contains information about users. To avoid transitive dependencies, the User table is not further decomposed. This ensures that each attribute in the Property table is dependent only on the primary key (PropertyId).

The same applies to the Location table, where attributes (City, State, Country) are fully dependent on the primary key (ZipCode), satisfying 3NF.

Normalization helps minimize data redundancy by breaking down tables into smaller, more manageable entities and establishing relationships between them. By ensuring that each table represents a single, coherent entity and that data dependencies are properly handled, normalization helps maintain data integrity and facilitates efficient data storage and retrieval.

1. **Constraints and Relationships**:
   * Highlight the foreign key constraints that enforce relationships between tables.
2. Property Table:

Foreign Key: TypeId

References: PropertyType(TypeId)

Function: Associates each property with its corresponding property type, defining the category or classification of the property.

Foreign Key: UserId

References: User(UserId)

Function: Establishes a relationship between properties and their owners (users), indicating who owns or is associated with each property.

Foreign Key: ZipCode

References: Location(ZipCode)

Function: Links each property to its geographic location, providing information about the city, state, and country where the property is situated.

1. User Table:

Foreign Key: ProfId

References: Professional(ProfId)

Function: Connects users to their respective professions, indicating the type of professional or occupation associated with each user.

1. Professional Table:

No foreign keys in the Professional table, but it provides the ProfId key referenced by the User table.

These foreign keys establish relationships between different tables in the database, enabling data normalization and ensuring referential integrity. By linking related entities, foreign keys facilitate data retrieval, analysis, and management within the database schema.

1. **Conclusion**:
   * Summarize the key points of your presentation.

In conclusion, the database we've designed serves as the backbone of our web-based application, facilitating efficient property management, user interaction, and collaboration within the real estate domain. Through meticulous normalization, we've organized the data into distinct tables, each with a specific purpose and role within the system.

From storing detailed property information to managing user accounts and facilitating location-based searches, our database enables seamless navigation and operation of the application. By establishing relationships between entities such as properties, users, locations, property types, and professionals, we ensure data integrity and enable robust functionalities.

As we move forward with the development of our web application, this database will continue to evolve and adapt to meet the needs of our users and stakeholders. With its solid foundation and carefully designed structure, we are confident that our database will serve as a reliable and efficient tool for property management.