CSCI 43500 DBMS

Project Instructions: Part 1

Project Overview

For this project, each group will work on designing a database system for one of the following domains.

Social media platform

To create a comprehensive social media platform, we need a database system that can manage user profiles, content creation, social interactions, and community engagement. Users will create profiles that include essential information such as username, location, and privacy settings. This data will allow users to personalize their profiles while controlling the visibility of their content. Posts will be a primary content type, allowing users to share various media types (text, images, videos) with timestamps and location tags. Each post should include visibility settings to define if it's public, friends-only, or private. This will allow users to curate their online presence and engage selectively within the platform.

Social interactions will be at the core of the platform, enhancing user engagement through likes, comments, and friend connections. The database should store details for each interaction, including user likes and comments on posts, allowing us to track engagement levels for each post. The friendship functionality will allow users to send, accept, or decline friend requests, with a status field to indicate pending, accepted, or blocked requests. In addition, the platform should support private messaging between users, storing message content, timestamps, and read/unread status to facilitate real-time communication. Notifications will play a critical role in alerting users about new friend requests, messages, likes, and comments, enhancing engagement by keeping users informed.

To promote a sense of community, the platform will include groups that users can join to discuss specific topics or interests. Each group will have its privacy settings, allowing it to be public, private, or invitation-only. Group memberships should record each member's role, such as admin or member, to manage permissions for posting and moderation within the group. The platform will also allow users to follow other users or pages, creating a more flexible network structure where users can choose to follow content without establishing a friend connection. These features will allow the database to handle diverse types of user interactions and ensure a scalable, efficient platform.

Hospital management system

The hospital management system should serve as a centralized database to manage and streamline patient records, doctor information, appointments, treatments, and billing processes. This system will track every patient's personal details, medical history, and visit records to ensure accurate treatment and continuity of care. Each patient entry will include their name, contact information, date of birth, and unique patient ID. Furthermore, the system should accommodate detailed medical records, including diagnoses, prescribed treatments, medications, and allergies, enabling doctors to make informed decisions based on comprehensive patient histories.

For managing healthcare staff, especially doctors and specialists, the database should store information about each medical professional's credentials, specialties, contact details, and schedule. This information will support the appointment scheduling system by allowing patients to book consultations with doctors who have availability and expertise relevant to their conditions. The appointment functionality should handle booking details, including patient and doctor IDs, appointment dates, times, and reasons for the visit. To accommodate a real-time scheduling system, the database should also track cancellations, reschedules, and follow-up appointments.

Financial tracking is another critical component of this system. The database should support billing for various services, such as consultations, treatments, and hospital stays. Each patient's billing history, including charges, payment status, and insurance claims, should be accessible to both the finance team and patients. To streamline insurance claim processes, the database should also track insurance providers, policy details, and coverage. Overall, the hospital management system will serve as a backbone for efficient hospital operations, helping maintain patient information, managing staff schedules, facilitating timely care, and keeping accurate billing records.

Food Delivery Service

For our food delivery service, we require a comprehensive database that can manage customer information, restaurant menus, order processing, and delivery tracking. Each customer should have a unique profile that includes their name, contact information, and saved addresses to facilitate a smooth ordering process. We need to keep a record of each customer's order history, allowing users to view previous orders and reorder with ease. Additionally, payment methods associated with customer profiles should be stored securely to enable seamless transactions.

Each participating restaurant must have its own profile in the system, containing essential details such as name, location, contact information, and available cuisine categories. Restaurants should be able to list their menu items, each with associated attributes like price, description, and availability. This will allow our system to display accurate, up-to-date menu information to customers. Additionally, restaurants should have the capability to update their menus and pricing as needed. Orders placed by customers will be linked to the corresponding restaurant, tracking items selected, quantities, and special instructions.

Once an order is placed, the system must handle order fulfillment, tracking each stage from preparation to delivery. This includes assigning delivery personnel, estimating delivery times, and tracking the real-time location of each order. The delivery personnel should have profiles with basic contact information and availability status to streamline the assignment process. Orders will include status updates (e.g., preparing, out for delivery, delivered) to keep customers informed. Additionally, the system should support multiple payment statuses (paid, pending, refunded) and provide a summary of charges, including delivery fees and taxes.

MMORPG (Massively Multiplayer Online Role-Playing Game)

To create a robust MMORPG (Massively Multiplayer Online Role-Playing Game) database system, we must manage diverse game elements, including player profiles, character details, in-game assets, quests, and multiplayer interactions. Each player will have an account that includes a username, email, password, and profile information. Players can create multiple characters within their account, each with unique attributes like character class, level, experience points, health, and mana. As characters progress in the game, they will earn experience, level up, and acquire skills and items that improve their gameplay, all of which should be stored and updated in real-time.

The game world will include various items and resources that players can acquire, trade, or equip to their characters. Items will have attributes like item type, rarity, and associated effects or boosts (e.g., increased health or attack power). To facilitate in-game economy and trade, the database will also track transactions between players and NPCs (non-playable characters), storing details of each transaction, such as item exchanged, currency involved, and time of transaction. Quests are a core component of the MMORPG experience, providing players with objectives and rewards upon completion. Each quest will have attributes like quest name, level requirement, objectives, rewards, and completion status, enabling players to track their progress.

Guilds and multiplayer parties enhance the social experience within MMORPGs by allowing players to collaborate. The database should support guild management, with attributes for guild names, members, and roles (e.g., leader, officer, member). Additionally, real-time multiplayer interactions, such as forming parties for dungeons or PvP (player vs. player) arenas, will require tracking party members, session details, and outcomes. The database will need to handle a high volume of updates to ensure the smooth functioning of dynamic interactions, such as combat, trading, and progression tracking.

Each group must choose one of these topics and create a comprehensive database design to manage the essential functions of that system. The database should handle various interactions and transactions required for the system's operation, providing efficient data management and retrieval.

Project Requirements

Your group project must include the following tasks:

1. **Identify Entities and Attributes**: Each group should identify the primary entities within the selected domain. For each entity, list the relevant attributes that capture the necessary data.
2. **Identify Primary and Foreign Keys**: Determine the primary key for each entity and identify any foreign keys that establish relationships between entities.
3. **Draw the ER Diagram**: Create an Entity-Relationship (ER) diagram that visually represents the database design. Ensure all entities, attributes, primary keys, foreign keys, and relationships are accurately depicted. Identify any relationships between each entity.
4. **Divide the Group Tasks**: Each group consists of three members. Once the design phase is complete, all three students will collaborate on finalizing the database design. Afterward, the students will proceed with coding the database creation in SQL.

Project Guidelines

1. Entity Identification

Examine the selected domain and identify all key entities. Entities represent major components within your database, such as "User" or "Post" in a social media platform, "Patient" or "Doctor" in a hospital management system, etc. Focus on defining the core entities that are central to the database functionality.

2. Attribute Definition

For each entity, list attributes that describe the necessary information. For instance, a "User" entity might have attributes such as 'username', 'email', 'password', and 'date_joined'. Ensure that each attribute is essential to the entity and provides meaningful data.

3. Primary and Foreign Key Identification

Identify a primary key for each entity. A primary key is a unique identifier for records within an entity. In cases where entities are related, define foreign keys to establish these relationships. Foreign keys should reference primary keys in related entities, creating links between tables.

4. ER Diagram Creation

Construct an ER diagram that visualizes the database structure. The ER diagram should include all entities, attributes, primary keys, foreign keys, and relationships. Use appropriate notation to indicate entity relationships (one-to-one, one-to-many, etc.) and cardinalities.

5. Task Division for Database Implementation

After finalizing the database design, all three students from each group will work on implementing the database. This will involve writing SQL code to create tables, define relationships, and enforce constraints based on the ER diagram.

Submission Requirements

Each group should submit the following:

- A written document listing identified entities and attributes, including primary and foreign keys for each entity.
- An ER diagram of the database design.
- SQL code for creating the database tables and relationships.
- Documentation that outlines the design process, any assumptions made, and the responsibilities of each group member.

Evaluation Criteria

Projects will be evaluated based on:

- Accuracy and completeness of entities, attributes, and relationships.
- Correct identification of primary and foreign keys.
- Clarity and correctness of the ER diagram.
- Proper implementation of SQL code based on the database design.
- Quality of documentation and teamwork coordination.

Deadline

The final project submission is due on 04/20/25 (end of day). Late submissions will be penalized according to the class policy.