Introduction

This database design creates a clear framework for handling users, roles, ideas, votes, and comments in our **University Idea Management System**. Each table is specifically designed to fit the project's needs, and the connections between them ensure data accuracy while making it easy to expand and search for information.

Please note that the proposed DBMS for this is PostgreSQL version 14.

1. Key Entities and Relationships

The primary entities based on the project requirements include:

1. **User**: Represents staff members who can submit ideas, comment, vote, and assume roles (e.g., QA Coordinator, QA Manager).
2. **Role**: Defines user roles in the system (e.g., QA Coordinator, QA Manager, Staff).
3. **Department**: Represents different departments in the university.
4. **Idea**: The main table for submitted ideas.
5. **Comment**: Stores comments on ideas.
6. **Vote**: Tracks votes on ideas (Thumbs Up or Thumbs Down).
7. **Category**: Allows tagging of ideas by categories.
8. **Notification**: For handling email notifications.
9. **SystemData**: For administrative configurations like closure dates.

2. Database Tables

Here's a suggested schema with each table, key attributes, and relationships:

1. User Table

- user_id (PK): Unique identifier.
- name: Full name of the user.
- email: Unique email, used for notifications.
- department_id (FK): Foreign key linking to Department.
- role_id (FK): Foreign key linking to Role.
- agreed_terms: Boolean indicating if they agreed to the terms and conditions.
- **Relationships**:
    - **One-to-Many** with **Department**.
    - **One-to-Many** with **Role**.
    - **One-to-Many** with **Idea**.

2. Role Table

- role_id (PK): Unique identifier.
- role_name: Name of the role (e.g., QA Manager, QA Coordinator, Staff).
- **Relationships**:
    - **One-to-Many** with **User**.

3. Department Table

- department_id (PK): Unique identifier.
- department_name: Name of the department.
- **Relationships**:
    - **One-to-Many** with **User**.

4. Idea Table

- idea_id (PK): Unique identifier.
- title: Title of the idea.
- description: Detailed description of the idea.
- user_id (FK): Foreign key linking to the **User** who submitted the idea.
- category_id (FK): Foreign key linking to **Category**.
- submit_date: Date when the idea was submitted.
- is_anonymous: Boolean indicating if the idea is posted anonymously.
- is_active: Boolean indicating if the idea is still open for votes and comments.
- **Relationships**:
    - **Many-to-One** with **User**.
    - **Many-to-One** with **Category**.
    - **One-to-Many** with **Comment**.
    - **One-to-Many** with **Vote**.

5. Comment Table

- comment_id (PK): Unique identifier.
- idea_id (FK): Foreign key linking to **Idea**.
- user_id (FK): Foreign key linking to **User** who made the comment.
- content: Content of the comment.
- is_anonymous: Boolean indicating if the comment is anonymous.
- comment_date: Date and time the comment was posted.
- **Relationships**:

    - **Many-to-One** with **Idea**.
    - **Many-to-One** with **User**.

6. Vote Table

- vote_id (PK): Unique identifier.
- idea_id (FK): Foreign key linking to **Idea**.
- user_id (FK): Foreign key linking to **User** who voted.
- vote_type: Boolean or integer indicating Thumbs Up (1) or Thumbs Down (-1).
- **Relationships**:
    - **Many-to-One** with **Idea**.
    - **Many-to-One** with **User**.

7. Category Table

- category_id (PK): Unique identifier.
- category_name: Name of the category.
- **Relationships**:
    - **One-to-Many** with **Idea**.

8. Notification Table

- notification_id (PK): Unique identifier.
- idea_id (FK): Foreign key linking to **Idea**.
- user_id (FK): Foreign key linking to **User** who is notified.
- notification_type: Type of notification (e.g., New Comment, Idea Submission).
- is_read: Boolean indicating if the notification has been read.
- **Relationships**:
    - **Many-to-One** with **User**.
    - **Many-to-One** with **Idea**.

9. SystemData Table

- setting_id (PK): Unique identifier.
- closure_date: Date when new idea submissions close.
- comment_closure_date: Date when comments close.
- **Purpose**:
    - Used by the administrator to set system-wide settings, including the idea submission closure date.

3. Data Integrity and Constraints

- **Primary and Foreign Keys**: Use primary keys for unique identification and foreign keys to enforce relationships between tables.
- **Referential Integrity**: Enforce relationships to prevent orphaned records (e.g., an idea without a user or department).
- **Data Validation**:
    - Use constraints like NOT NULL where required (e.g., email, name).
    - Validate vote_type to only accept 1 or -1.
- **Unique Constraints**: Set unique constraints on fields like email and possibly user_id + idea_id in the Vote table to prevent multiple votes from the same user on the same idea.

4. Indexes and Optimization

- **Indexes**: Create indexes on fields frequently queried, such as submit_date in the Idea table, vote_type in the Vote table, and user_id in the User table.
- **Pagination**: Use LIMIT and OFFSET in SQL queries to paginate results (e.g., for lists of ideas or comments).

5. Additional Suggestions

- **Stored Procedures or Triggers**: Consider using triggers to send email notifications after a new idea or comment is posted.
- **Role-Based Access Control**: Implement role-based access in the application layer, but enforce it in the database using role tables and SQL grants if possible.