AcademiX Database Schema (Structured Format with Relationships & Explanations)

The database schema for **AcademiX** consists of multiple interconnected tables to manage users, collaborations, projects, mentorships, queries, and authentication efficiently. Below is a structured representation of the schema along with explanations for each table, relationships, and attributes.

---

1. Users Table (`users`)

Purpose: Stores user-related data, including authentication details.
Primary Key: `user_id`
Relationships:

- Connects to **Projects** (one-to-many) → A user can create multiple projects.
- Connects to **Mentorships** (one-to-many) → A user can be a mentor or mentee.
- Connects to **Study Groups** (many-to-many) → A user can be part of multiple study groups.
- Connects to **Queries** (one-to-many) → A user can ask multiple queries.

Column Name	Data Type	Constraints	Description
`user_id`	INT	PRIMARY KEY, AUTO_INCREMENT	Unique identifier for each user.
`name`	VARCHAR(100)	NOT NULL	Full name of the user.
`email`	VARCHAR(255)	UNIQUE, NOT NULL	Email used for authentication.
`password_hash`	VARCHAR(255)	NOT NULL	Encrypted password for security.
`role`	ENUM('student', 'mentor')	NOT NULL	Defines user type: student or mentor.
`created_at`	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Account creation timestamp.

2. Projects Table (`projects`)

Purpose: Stores projects created by students for collaboration.
Primary Key: `project_id`
Foreign Key: `owner_id` (references `users.user_id`)
Relationships:

- Each **Project** belongs to one **User** (creator).
- Projects can have **multiple contributors** (many-to-many).

Column Name	Data Type	Constraints	Description
`project_id`	INT	PRIMARY KEY, AUTO_INCREMENT	Unique identifier for each project.
`owner_id`	INT	FOREIGN KEY REFERENCES `users(user_id)`	User who created the project.
`title`	VARCHAR(255)	NOT NULL	Title of the project.
`description`	TEXT	NOT NULL	Detailed project description.
`status`	ENUM('open', 'in progress', 'completed')	DEFAULT 'open'	Current state of the project.
`created_at`	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Project creation timestamp.

3. Collaborations Table (`collaborations`)

Purpose: Manages the users collaborating on projects.
Primary Key: `collab_id`
Foreign Keys:

- `project_id` (references `projects.project_id`)
- `user_id` (references `users.user_id`)

Column Name	Data Type	Constraints	Description
collab_id	INT	PRIMARY KEY, AUTO_INCREMENT	Unique identifier for collaboration.
project_id	INT	FOREIGN KEY REFERENCES projects(project_id)	Associated project.
user_id	INT	FOREIGN KEY REFERENCES users(user_id)	User participating in collaboration.
role	VARCHAR(50)	NOT NULL	Role in the project (e.g., developer, designer).
joined_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Timestamp when the user joined.

---

4. Mentorships Table (mentorships)

Purpose: Stores mentorship relationships between students and mentors.
Primary Key: mentorship_id
Foreign Keys:

- mentor_id (references users.user_id)
- mentee_id (references users.user_id)

Column Name	Data Type	Constraints	Description
mentorship_id	INT	PRIMARY KEY, AUTO_INCREMENT	Unique mentorship record.
mentor_id	INT	FOREIGN KEY REFERENCES users(user_id)	The mentor in the relationship.

`mentee_id`	INT	FOREIGN KEY REFERENCES `users(user_id)`	The student being mentored.
`status`	ENUM('active', 'completed')	DEFAULT 'active'	Status of mentorship.
`started_at`	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Mentorship start date.

---

5. Study Groups Table (`study_groups`)

Purpose: Manages study groups where students collaborate.
Primary Key: `group_id`
Foreign Key: `owner_id` (references `users.user_id`)

Column Name	Data Type	Constraints	Description
`group_id`	INT	PRIMARY KEY, AUTO_INCREMENT	Unique study group ID.
`owner_id`	INT	FOREIGN KEY REFERENCES `users(user_id)`	The creator of the study group.
`group_name`	VARCHAR(100)	NOT NULL	Name of the study group.
`description`	TEXT	NOT NULL	Study group details.
`created_at`	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Creation timestamp.

---

6. Study Group Members (`study_group_members`)

Purpose: Stores which users belong to which study groups (many-to-many relationship).
Primary Key: Composite key (`group_id`, `user_id`)
Foreign Keys:

- `group_id` (references `study_groups.group_id`)
- `user_id` (references `users.user_id`)

Column Name	Data Type	Constraints	Description
`group_id`	INT	FOREIGN KEY REFERENCES `study_groups(group_id)`	The study group.
`user_id`	INT	FOREIGN KEY REFERENCES `users(user_id)`	The user in the study group.
`joined_at`	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	When the user joined.

---

7. Queries Table (`queries`)

Purpose: Stores queries posted by users for academic discussions.
Primary Key: `query_id`
Foreign Key: `user_id` (references `users.user_id`)

Column Name	Data Type	Constraints	Description
`query_id`	INT	PRIMARY KEY, AUTO_INCREMENT	Unique query ID.
`user_id`	INT	FOREIGN KEY REFERENCES `users(user_id)`	User who posted the query.
`title`	VARCHAR(255)	NOT NULL	Query title.
`content`	TEXT	NOT NULL	Full query description.

`created_at`	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Timestamp when query was posted.

---

8. OTP Verification Table (`otp_verifications`)

Purpose: Stores OTPs for authentication.
Primary Key: `otp_id`
Foreign Key: `user_id` (references `users.user_id`)

Column Name	Data Type	Constraints	Description
`otp_id`	INT	PRIMARY KEY, AUTO_INCREMENT	Unique OTP ID.
`user_id`	INT	FOREIGN KEY REFERENCES `users(user_id)`	User requesting OTP.
`otp_code`	VARCHAR(6)	NOT NULL	Generated OTP code.
`expires_at`	TIMESTAMP	NOT NULL	Expiration time of OTP.

Relationship Summary

- **One-to-Many:**
  - A **User** can have multiple **Projects**.
  - A **User** can create multiple **Study Groups**.
  - A **User** can post multiple **Queries**.
- **Many-to-Many:**
  - A **Study Group** can have multiple **Users** and vice versa.
  - A **Project** can have multiple **Collaborators** and vice versa.
- **One-to-One:**
  - Each **User** can only be in **one active mentorship** at a time.
