Database Table Relationships ❓

Based on the columns that appear in multiple tables and typical database design principles, the following relationships can be inferred:

---

1. `Societies` and `Users` Tables: 🏛️👤

- **Relationship:** One-to-One / Many-to-One
- **Foreign Key:** `Societies.president_id` refers to `Users.user_id`.
- **Explanation:** Each society has one president, and a president is a user. The `president_id` in the `Societies` table links directly to the `user_id` of the user who is the president. While a user can be president of only one society at a time (implied by `Societies.president_id` being a direct reference), a user may or may not be a president.

---

2. `Societies` and `Societies_Members` Tables: 🏛️👥

- **Relationship:** One-to-Many
- **Foreign Key:** `Societies_Members.society_id` refers to `Societies.society_id`.
- **Explanation:** A society can have multiple members. The `society_id` in the `Societies_Members` table indicates which society a particular member belongs to.

---

3. `Users` and `Societies_Members` Tables: 👤👥

- **Relationship:** One-to-Many
- **Foreign Key:** `Societies_Members.user_id` refers to `Users.user_id`.
- **Explanation:** A user can be a member of one or more societies (though in the provided data, `user_id` 19 is a member of society 8 and `user_id` 18 is also a member of society 8, indicating a user can be a member). The `user_id` in the `Societies_Members` table identifies which user is a member.

---

4. `Users` and `Role_Request` Tables: 👤✋

- **Relationship:** One-to-Many
- **Foreign Key:** `Role_Request.user_id` refers to `Users.user_id`.
- **Explanation:** A user can make multiple role requests. The `user_id` in the `Role_Request` table indicates which user made the request for a specific role.

---

5. `Societies` and `Events` Tables: 🎉

- **Relationship:** One-to-Many
- **Foreign Key:** `Events.society_id` refers to `Societies.society_id`.
- **Explanation:** A society can organize multiple events. The `society_id` in the `Events` table indicates which society created or is responsible for a particular event.

---

6. `Users` and `Events` Tables: ✍

- **Relationship:** One-to-Many
- **Foreign Key:** `Events.created_by` refers to `Users.user_id`.
- **Explanation:** A user can create multiple events. The `created_by` in the `Events` table identifies which user initiated the event.

---

7. `Events` and `Event_Request` Tables: ✅

- **Relationship:** One-to-One (or One-to-Many if an event can have multiple approval requests, though the sample only shows one)
- **Foreign Key:** `Event_Request.event_id` refers to `Events.event_id`.
- **Explanation:** An event can have one or more approval requests associated with it. The `event_id` in the `Event_Request` table links to the specific event being requested for approval. Given the current data, it seems to be a one-to-one relationship, where each event request corresponds to a single event for approval.