Архитектор базы данных

1. Основные сущности

Таблица: user (Пользователи)

- Назначение: Хранение учетных данных, профилей и прав доступа пользователей системы.
- Описание полей:
  - user_id (INTEGER) — Уникальный идентификатор пользователя. PRIMARY KEY. Значение генерируется последовательностью user_id_seq.
  - email (VARCHAR(255)) — Электронная почта, используется как логин.
  - hashed_password (VARCHAR) — Хэшированный пароль для безопасной аутентификации.
  - full_name (VARCHAR(255)) — Полное имя пользователя.
  - phone_number (VARCHAR(20)) — Контактный номер телефона.
  - role_id (INTEGER) — Идентификатор роли пользователя в системе. Ссылается на таблицу role.
  - is_active (BOOLEAN) — Флаг активности учетной записи (активна/заблокирована).
  - is_superuser (BOOLEAN) — Флаг, указывающий на наличие административных прав.
  - created_at, updated_at (TIMESTAMPTZ) — Временные метки создания и последнего обновления записи. created_at по умолчанию устанавливается в now().
- Ограничения (Constraints):
  - user_pk (PRIMARY KEY): Первичный ключ на поле user_id.
  - user_role_id_fkey (FOREIGN KEY): Внешний ключ, связывающий role_id с таблицей role. Гарантирует целостность ссылки на существующую роль.
  - NOT NULL: Обязательные поля: email, is_active, is_superuser, hashed_password, user_id, created_at.
- Индексы (Indexes):
  - ix_user_email (UNIQUE INDEX, BTREE): Обеспечивает уникальность значений в поле email и ускоряет поиск пользователя по email. Создан явно командой CREATE UNIQUE INDEX.

Таблица: role (Роли)

- Назначение: Справочник возможных ролей пользователей в системе.
- Описание полей:
  - role_id (INTEGER) — Уникальный идентификатор роли. PRIMARY KEY. Значение генерируется последовательностью role_id_seq.
  - name (VARCHAR) — Наименование роли (например, "Администратор", "Модератор", "Пользователь").
- Ограничения:
  - role_pk (PRIMARY KEY): Первичный ключ на поле role_id.
  - NOT NULL: name, role_id.
- Индексы:

    - role_name_key (UNIQUE INDEX, BTREE): Гарантирует уникальность названий ролей и ускоряет поиск роли по имени.

Таблица: entity (Основная сущность: Место или Мероприятие)

- Назначение: Хранение полной информации об объектах каталога (мероприятие, место).
- Описание полей:
    - entity_id (INTEGER) — Уникальный идентификатор объекта. PRIMARY KEY. Значение генерируется последовательностью entity_id_seq.
    - name, contributors, address, description, links, contacts, photo, metro (VARCHAR) — Основные атрибуты объекта: название, организаторы, адрес, описание, ссылки, контакты, фото, ближайшее метро.
    - cost, average_cost, age_gap, date (VARCHAR) — Дополнительные атрибуты: информация о стоимости, возрастные ограничения, дата проведения.
    - created_at, updated_at (TIMESTAMPTZ) — Временные метки. created_at по умолчанию now().
- Ограничения:
    - entity_pk (PRIMARY KEY): Первичный ключ на поле entity_id.
    - NOT NULL: Поля name, contributors, address, description, links, contacts, photo, metro, entity_id, created_at.
- Индексы: Явные индексы не созданы. Для ускорения поиска по name или metro рекомендуется добавить не уникальные индексы.

Таблица: category (Категории)

- Назначение: Иерархический справочник категорий для классификации объектов. Поддерживает вложенность (дерево категорий).
- Описание полей:
    - category_id (INTEGER) — Уникальный идентификатор категории. PRIMARY KEY. Значение генерируется последовательностью category_id_seq.
    - name (VARCHAR) — Название категории.
    - parent_id (INTEGER) — Идентификатор родительской категории. Если NULL, категория является корневой.
    - created_at, updated_at (TIMESTAMPTZ) — Временные метки.
- Ограничения:
    - category_pk (PRIMARY KEY): Первичный ключ на поле category_id.
    - category_parent_id_fkey (FOREIGN KEY): Внешний ключ, связывающий parent_id с category_id в этой же таблице. Реализует иерархию и гарантирует ссылку на существующую категорию.
    - NOT NULL: name, category_id, created_at.
- Индексы:
    - category_name_key (UNIQUE INDEX, BTREE): Гарантирует уникальность названий категорий и ускоряет поиск по имени.

Таблица: tag (Теги)

- Назначение: Ключевые характеристики сущностей.
- Описание полей:

    - tag_id (INTEGER) — Уникальный идентификатор тега. PRIMARY KEY. Значение генерируется последовательностью tag_id_seq.
    - name (VARCHAR) — Название тега.
    - created_at, updated_at (TIMESTAMPTZ) — Временные метки.
- Ограничения:
    - tag_pk (PRIMARY KEY): Первичный ключ на поле tag_id.
    - NOT NULL: name, tag_id, created_at.
- Индексы:
    - tag_name_key (UNIQUE INDEX, BTREE): Гарантирует уникальность названий тегов и ускоряет поиск по имени.

2. Связующие таблицы

Таблица: entitycategory (Связь объекта с категориями)

- Назначение: Реализация связи «многие-ко-многим» между объектами (entity) и категориями (category). Один объект может принадлежать к нескольким категориям, и в одной категории может быть много объектов.
- Описание полей:
    - entity_category_id (INTEGER) — Первичный ключ связи. PRIMARY KEY. Значение генерируется последовательностью entitycategory_entity_category_id_seq.
    - entity_id (INTEGER) — Идентификатор объекта. FOREIGN KEY на entity(entity_id).
    - category_id (INTEGER) — Идентификатор категории. FOREIGN KEY на category(category_id).
    - created_at, updated_at (TIMESTAMPTZ) — Временные метки.
- Ограничения:
    - entitycategory_pkey (PRIMARY KEY): Первичный ключ на поле entity_category_id.
    - entitycategory_entity_id_fkey, entitycategory_category_id_fkey (FOREIGN KEY): Внешние ключи, обеспечивающие целостность связей.
    - NOT NULL: entity_id, category_id, entity_category_id, created_at.
- Индексы:
    - pair_entity_category (UNIQUE INDEX, BTREE): Составной уникальный индекс по полям (entity_id, category_id). Гарантирует, что одна и та же категория не будет присвоена объекту дважды. Также ускоряет поиск всех категорий объекта или всех объектов в категории.

Таблица: entitytag (Связь объекта с тегами)

- Назначение: Реализация связи «многие-ко-многим» между объектами (entity) и тегами (tag).
- Структура, ограничения и индексы аналогичны таблице entitycategory. Использует уникальный индекс pair_entity_tag.

Таблица: relationtype (Типы отношений пользователя к объекту)

- Назначение: Справочник типов связей между пользователем и объектом (например, "создал", "посетил", "в избранном", "организатор").

- Описание полей:
    - relation_type_id (INTEGER) — Уникальный идентификатор типа отношения. PRIMARY KEY. Значение генерируется последовательностью relationtype_relation_type_id_seq.
    - name (VARCHAR) — Название типа.
    - created_at, updated_at (TIMESTAMPTZ) — Временные метки.
- Ограничения:
    - relationtype_pkey (PRIMARY KEY): Первичный ключ.
    - NOT NULL: name, relation_type_id, created_at.
- Индексы: Явные индексы не созданы. Для ускорения поиска по имени можно добавить индекс.

Таблица: userentity (Связь пользователя с объектом)

- Назначение: Фиксация конкретных отношений пользователей к объектам (например, "Петя посетил Музей" или "Анна добавила Парк в избранное").
- Описание полей:
    - user_entity_id (INTEGER) — Первичный ключ связи. PRIMARY KEY. Значение генерируется последовательностью userentity_user_entity_id_seq.
    - user_id (INTEGER) — Идентификатор пользователя. FOREIGN KEY на user(user_id).
    - entity_id (INTEGER) — Идентификатор объекта. FOREIGN KEY на entity(entity_id).
    - relation_type_id (INTEGER) — Идентификатор типа отношения. FOREIGN KEY на relationtype(relation_type_id).
    - created_at, updated_at (TIMESTAMPTZ) — Временные метки.
- Ограничения:
    - userentity_pkey (PRIMARY KEY): Первичный ключ.
    - userentity_user_id_fkey, userentity_entity_id_fkey, userentity_relation_type_id_fkey (FOREIGN KEY): Внешние ключи, обеспечивающие целостность данных.
    - NOT NULL: user_id, entity_id, relation_type_id, user_entity_id, created_at.
- Индексы:
    - pair_user_entity (UNIQUE INDEX, BTREE): Составной уникальный индекс по полям (user_id, entity_id). Предотвращает дублирование связи между одной и той же парой пользователь-объект (например, нельзя дважды добавить одно место в избранное). Это ключевое бизнес-ограничение.

Таблица: usertag (Связь пользователя с тегами)

- Назначение: Сохранение персональных предпочтений пользователя в виде тегов. Позволяет пользователю отмечать интересные ему теги (например, "кофейни", "искусство", "парки") для персонализации ленты или получения рекомендаций. Реализует связь «многие-ко-многим» между user и tag.
- Описание полей:
    1. user_tag_id (INTEGER) — Первичный ключ связи. PRIMARY KEY.
    2. user_id (INTEGER) — Идентификатор пользователя. FOREIGN KEY на user(user_id).
    3. tag_id (INTEGER) — Идентификатор тега. FOREIGN KEY на tag(tag_id).

4. created_at, updated_at (TIMESTAMPTZ) — Временные метки создания и обновления связи. Могут быть NULL, что нежелательно.

- Ограничения (Constraints):
    1. PRIMARY KEY (явно в коде): Первичный ключ на поле user_tag_id.
    2. pair_user_tag_user (FOREIGN KEY): Внешний ключ, связывающий user_id с таблицей user. Помечен как NOT VALID, что означает, что ограничение создано, но не проверяется для уже существующих в таблице строк.
    3. pair_user_tag_tag (FOREIGN KEY): Внешний ключ, связывающий tag_id с таблицей tag. Помечен как NOT VALID, что означает, что ограничение создано, но не проверяется для уже существующих в таблице строк.
    4. NOT NULL: user_tag_id, user_id, tag_id.

Таблица: usercategory (Связь пользователя с категориями)

- Назначение: Сохранение категорий, которые пользователь выбрал как интересные для себя (например, "Музеи", "Театры", "Спорт"). Используется для настройки персональной ленты событий или фильтрации контента. Реализует связь «многие-ко-многим» между user и category.
- Описание полей:
    1. user_category_id (INTEGER) — Первичный ключ связи. PRIMARY KEY.
    2. user_id (INTEGER) — Идентификатор пользователя. FOREIGN KEY на user(user_id).
    3. category_id (INTEGER) — Идентификатор категории. FOREIGN KEY на category(category_id).
    4. created_at, updated_at (TIMESTAMPTZ) — Временные метки. Могут быть NULL, что нежелательно.
- Ограничения:
    1. usercategory_pkey (PRIMARY KEY): Первичный ключ на поле user_category_id.
    2. user_category_user (FOREIGN KEY): Внешний ключ, связывающий user_id с таблицей user.
    3. user_category_category (FOREIGN KEY): Внешний ключ, связывающий category_id с таблицей category.
    4. NOT NULL: user_category_id, user_id, category_id.

Порядок заполнения таблиц:

1. Роли
2. Категории
3. Теги
4. Сущности

Затем заполняется автоматически таблица Пользователи (user) по мере добавления новых зарегистрированных пользователей. Также заполняются все связующие таблицы по мере заполнения основных.