МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное автономное
образовательное учреждение высшего образования
«Санкт-Петербургский национальный исследовательский университет
информационных технологий, механики и оптики»

ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ И КОМПЬЮТЕРНОЙ ТЕХНИКИ

ЛАБОРАТОРНАЯ РАБОТА № 1

по дисциплине

‘Базы данных’

Вариант №408078

Выполнил:
Студент группы P3132
Юксель Хамза

Преподаватель:
Егошин Алексей
Васильевич

УНИВЕРСИТЕТ ИТМО

Санкт-Петербург, 2025

Описание задания

Для выполнения лабораторной работы №1 необходимо:

1. На основе предложенной предметной области (текста) составить ее описание. Из полученного описания выделить сущности, их атрибуты и связи.
2. Составить инфологическую модель.
3. Составить даталогическую модель. При описании типов данных для атрибутов должны использоваться типы из СУБД PostgreSQL.
4. Реализовать даталогическую модель в PostgreSQL. При описании и реализации даталогической модели должны учитываться ограничения целостности, которые характерны для полученной предметной области.
5. Заполнить созданные таблицы тестовыми данными.

Описание предметной области

В конце концов они остановились перед закрытой дверью, которая тотчас же медленно скользнула вбок, а затем снова задвинулась за ними, отрезав им путь к отступлению. Впереди была еще одна дверь которая, однако, при их приближении не отворилась. Хедрон не сделал ни малейшей попытки хотя бы коснуться ее, он просто остановился. Через короткое время прозвучал тихий голос: <Будьте добры, назовите ваши имена>.

In the city of Diaspar, which can host multiple zones every zone is protected by a single security system that cannot exist without its zone, and that system in turn manages multiple doors or barriers while always including exactly one voice interface. Characters in the city live independently and may or may not interact with any door or barrier; when they do, their attempts are logged through access records, each of which can spawn multiple events describing what happened. A character can also try to escape from a zone, creating escape attempts linked to that zone, though a zone itself can have zero or many such attempts. Lastly, missions in Diaspar represent grand objectives that characters can optionally join via mission participation, meaning a mission can involve many characters or none, and each character can choose to take part in many missions or none at all.

В городе Диаспар, которому могут принадлежать несколько зон, каждая зона охраняется единой системой безопасности, которая не может существовать без своей зоны. Эта система, в свою очередь, управляет несколькими дверями или барьерами и всегда включает в себя один голосовой интерфейс. Персонажи в городе живут независимо и могут как взаимодействовать, так и не взаимодействовать с дверями или барьерами; когда это происходит, их попытки фиксируются в записях доступа, каждая из которых может порождать несколько событий, описывающих произошедшее. Персонаж также может попытаться сбежать из зоны, создавая попытки побега, которые связываются с этой зоной, хотя у самой зоны таких попыток может быть как ноль, так и много. Наконец, в Диаспаре существуют миссии — это важные задачи, в которых персонажи могут участвовать по желанию, что создает связи через участие в миссиях: миссия может включать много персонажей или ни одного, а каждый персонаж может участвовать в нескольких миссиях или не участвовать вовсе

Список сущностей

Стержневые (Core):

- *City*
- *Character*
- *Mission*

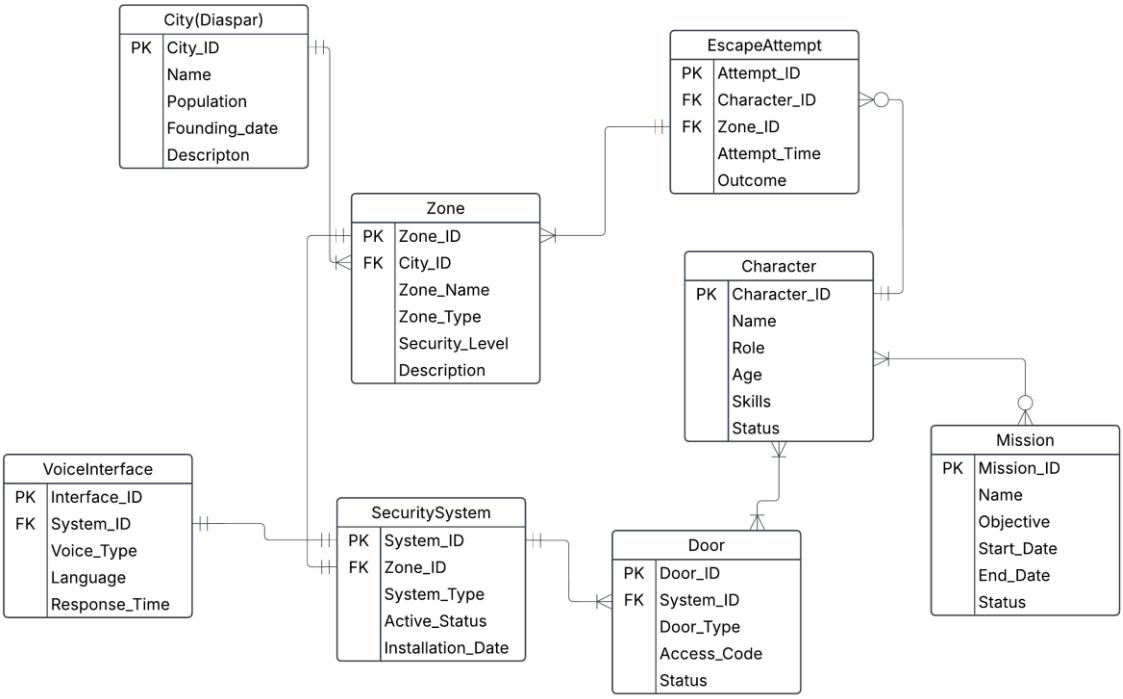
Характеристические (Characteristic):

- *Zone*
- *Security System*
- *Door/Barrier*
- *Voice Interface*
- *Event*
- *Escape Attempt*

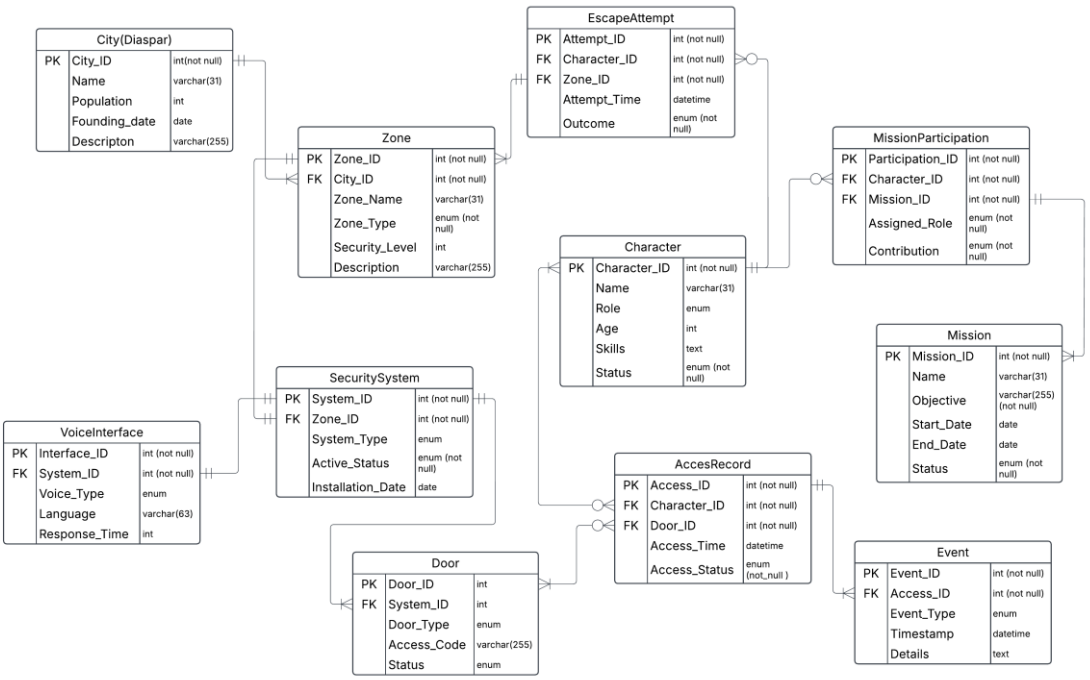
Ассоциативные (Associative):

- Access Record
- Mission Participation

Инфологическая модель



Даталогическая модель



Реализация модели на SQL

```
CREATE TYPE outcome AS ENUM ('Success', 'Failure');
CREATE TYPE zone_type AS ENUM ('Forbidden', 'Residental', 'Industrial');
CREATE TYPE system_type AS ENUM ('Surveillance', 'Access', 'Control');
CREATE TYPE active_status AS ENUM ('Active', 'Disable');
CREATE TYPE door_type AS ENUM ('Sliding', 'Locking');
CREATE TYPE door_status AS ENUM ('Open', 'Closed', 'Locked', 'Blocked');
CREATE TYPE voice_type AS ENUM ('Male', 'Female','Robotic');
CREATE TYPE role AS ENUM ('Scout', 'Leader');
CREATE TYPE character_status AS ENUM ('Dead', 'Captured','Alive');
CREATE TYPE access_status AS ENUM ('Garanted', 'Denied');
CREATE TYPE event_Type AS ENUM ('Door Opened', 'Security Alert');
CREATE TYPE mission_status AS ENUM ('In Progress', 'Completed');
CREATE TYPE assigned_role AS ENUM ('Recon', 'Support');
CREATE TYPE contribution AS ENUM ('Strategy', 'Combat');



CREATE TABLE City (
  city_id SERIAL PRIMARY KEY,
  name VARCHAR(31) NOT NULL DEFAULT 'Diaspar',
  population INT CHECK (population >=0),
  founding_date DATE CHECK (founding_date <= CURRENT_DATE)
);

CREATE TABLE Character (
  character_id SERIAL PRIMARY KEY,
  name VARCHAR(31) NOT NULL,
  role role NOT NULL,
  age INT CHECK (age BETWEEN 0  AND 100),
  skills TEXT,
  status character_status NOT NULL
);

CREATE TABLE Zone (
  zone_id SERIAL PRIMARY KEY,
  city_id INT REFERENCES city(city_id) ON DELETE CASCADE NOT NULL,
  name VARCHAR(31) NOT NULL,
  zone_type zone_type NOT NULL,
  security_level INT,
  description TEXT
);

CREATE TABLE SecuritySystem (
    system_id SERIAL PRIMARY KEY,
    zone_id INT REFERENCES zone(zone_id) ON DELETE CASCADE NOT NULL,
    city_id INT REFERENCES city(city_id) ON DELETE CASCADE NOT NULL,
    security_type system_type NOT NULL,
    status active_status NOT NULL DEFAULT 'Active',
    installation_date DATE NOT NULL
);
```

```
CREATE OR REPLACE FUNCTION check_installation_date()
RETURNS TRIGGER AS
$$
DECLARE
   city_founding_date DATE;
BEGIN
   -- Pull the founding date for this city.
   SELECT founding_date INTO city_founding_date
   FROM city
   WHERE city_id = NEW.city_id;

   -- If the city was not found, or if the date check fails, throw an error.
   IF city_founding_date IS NULL THEN
      RAISE EXCEPTION 'No such city_id: %', NEW.city_id;
   END IF;

   IF NEW.installation_date < city_founding_date THEN
      RAISE EXCEPTION 'installation_date (%) must be on or after city founding_date (%)',
                  NEW.installation_date, city_founding_date;
   END IF;

   RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trg_check_installation_date
BEFORE INSERT OR UPDATE ON SecuritySystem
FOR EACH ROW
EXECUTE PROCEDURE check_installation_date();

CREATE TABLE Door (
  Door_ID SERIAL PRIMARY KEY,
  System_ID INT REFERENCES SecuritySystem(System_ID) ON DELETE CASCADE NOT NULL,
  Door_Type Door_Type,
  Access_Code VARCHAR(255),
  Status Door_Status
);

CREATE TABLE AccessRecord (
  Access_ID SERIAL PRIMARY KEY,
  Character_ID INT REFERENCES Character(Character_ID) ON DELETE CASCADE,
  Door_ID INT REFERENCES Door(Door_ID) ON DELETE CASCADE,
  Access_Time TIMESTAMP NOT NULL DEFAULT now(),
  Access_Status Access_Status
);

CREATE TABLE Event (
  Event_ID SERIAL PRIMARY KEY,
  Access_ID INT REFERENCES AccessRecord(Access_ID) ON DELETE CASCADE,
  Event_Type Event_Type NOT NULL,
  Timestamp TIMESTAMP NOT NULL DEFAULT now(),
  Details TEXT
);

CREATE TABLE EscapeAttempt (
  Attempt_ID SERIAL PRIMARY KEY,
  Character_ID INT REFERENCES Character(Character_ID) ON DELETE CASCADE NOT NULL,
```

```
  Zone_ID INT REFERENCES Zone(Zone_ID) ON DELETE CASCADE NOT NULL,
  Attempt_Time TIMESTAMP NOT NULL DEFAULT now(),
  Outcome Outcome
);

CREATE TABLE Mission (
  mission_id SERIAL PRIMARY KEY,
  name VARCHAR(31) NOT NULL,
  objective TEXT,
  start_date DATE,
  end_date DATE
  CHECK (end_date > start_date),
  status mission_status
);

CREATE TABLE MissionParticipation (
  Participation_ID SERIAL PRIMARY KEY,
  Character_ID INT REFERENCES Character(Character_ID) ON DELETE CASCADE NOT NULL,
  Mission_ID INT REFERENCES Mission(Mission_ID) ON DELETE CASCADE NOT NULL,
  Assigned_Role Assigned_Role,
  Contribution Contribution
);

CREATE TABLE VoiceInterface (
  Interface_ID SERIAL PRIMARY KEY,
  System_ID INT REFERENCES SecuritySystem(System_ID) ON DELETE CASCADE NOT NULL,
  Voice_Type Voice_Type,
  Language VARCHAR(63),
  Response_Time INT
);




INSERT INTO City (City_ID, Name, Population, Founding_Date)
VALUES (1, 'Diaspar', 100000, '2000-01-01');


INSERT INTO Character (Character_ID,Name, Role, Age, Skills, Status)
VALUES
  ('1', Hedron', 'Leader', 40, 'Leadership, Strategy', 'Alive'),
  ('2', 'Astra', 'Scout', 28, 'Stealth, Navigation', 'Alive'),
  ('3', 'Finn', 'Scout', 32, 'Agility, Observation', 'Alive');

INSERT INTO Zone (Zone_ID, City_ID, Name, Zone_Type, Security_Level, Description)
VALUES
  (1, 1, 'Forbidden Zone', 'Forbidden', 10, 'A dangerous and restricted area beyond Diaspar"s boundaries.');

INSERT INTO SecuritySystem (System_ID, Zone_ID, Type, Status, Installation_Date)
VALUES
  (1, 1, 'Control', 'Active', '1990-06-15');

INSERT INTO Door (Door_ID, System_ID, Door_Type, Access_Code, Status)
VALUES
  (1, 1, 'Locking', 'XYZ123', 'Closed');
```

```
INSERT INTO VoiceInterface (Interface_ID, System_ID, Voice_Type, Language, Response_Time)
VALUES
  (1, 'Robotic', 'English', 3);

INSERT INTO AccessRecord (Access_ID, Character_ID, Door_ID, Access_Time, Access_Status)
VALUES
  (1, 1, now(), 'Garanted');

INSERT INTO Event (Event_ID, Access_ID, Event_Type, Timestamp, Details)
VALUES
 (1, 1, 'Security Alert', now(), 'Alert! Security Breach' );

INSERT INTO EscapeAttempt (Attempt_ID , Character_ID, Zone_ID, Attempt_Time, Outcome)
VALUES
  (1, 1, 1, now(), 'Success');

INSERT INTO Mission (Mission_ID, Name, Objective, Start_Date, End_Date, Status)
VALUES
  (1, 'Escape', 'Uncover the truth beyond Diaspar and flee the sealed city.', CURRENT_DATE, NULL, 'In
Progress');

INSERT INTO MissionParticipation (Character_ID, Mission_ID, Assigned_Role, Contribution)
VALUES
  (1, 1, 'Recon', 'Strategy'),
  (2, 1, 'Support', 'Combat'),
  (3, 1, 'Support', 'Combat');
```

Выводы

При выполнении лабораторной работы я познакомился с разными моделями представления данных: составил инфологическую и даталогическую модель сущностей. Научился реализовывать даталогические модели произвольной предметной области с помощью SQL.