МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

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

Факультет Прикладная Информатика

ЛАБОРАТОРНАЯ РАБОТА № 3

по дисциплине

“ Проектирование и реализация баз данных”

Выполнил:
Мохаджер Алиреза
Джафари Хоссаин

Студент группы K3240

Преподаватель:
Белов Александр Олегович
Говорова Марина Михайловна

Санкт-Петербург, 2025

Цель работы

овладеть практическими навыками создания таблиц базы данных PostgreSQL 1X, заполнения их рабочими данными, резервного копирования и восстановления БД.

Практическое задание:

1. Создать базу данных с использованием pgAdmin 4 (согласно индивидуальному заданию).
2. Создать схему в составе базы данных.
3. Создать таблицы базы данных.
4. Установить ограничения на данные: Primary Key, Unique, Check, Foreign Key.
5. Заполнить таблицы БД рабочими данными.
6. Создать резервную копию БД.

Указание:

Создать две резервные копии:

- с расширением CUSTOM для восстановления БД;
- с расширением PLAIN для листинга (в отчете);
- при создании резервных копий БД настроить параметры Dump options для Type of objects и Queries .

7. Восстановить БД.

Список сущностей

Стержневые:

device - информация устройств

Data storage - Различные типы хранилищ информации

Burst - хранит информацию о вспышке в космосе

Planet - Планета, вокруг которой вращаются космические аппараты.

Planet

Ассоциативные:

Record information of explosion - Запись информации о вспышке космическим устройством

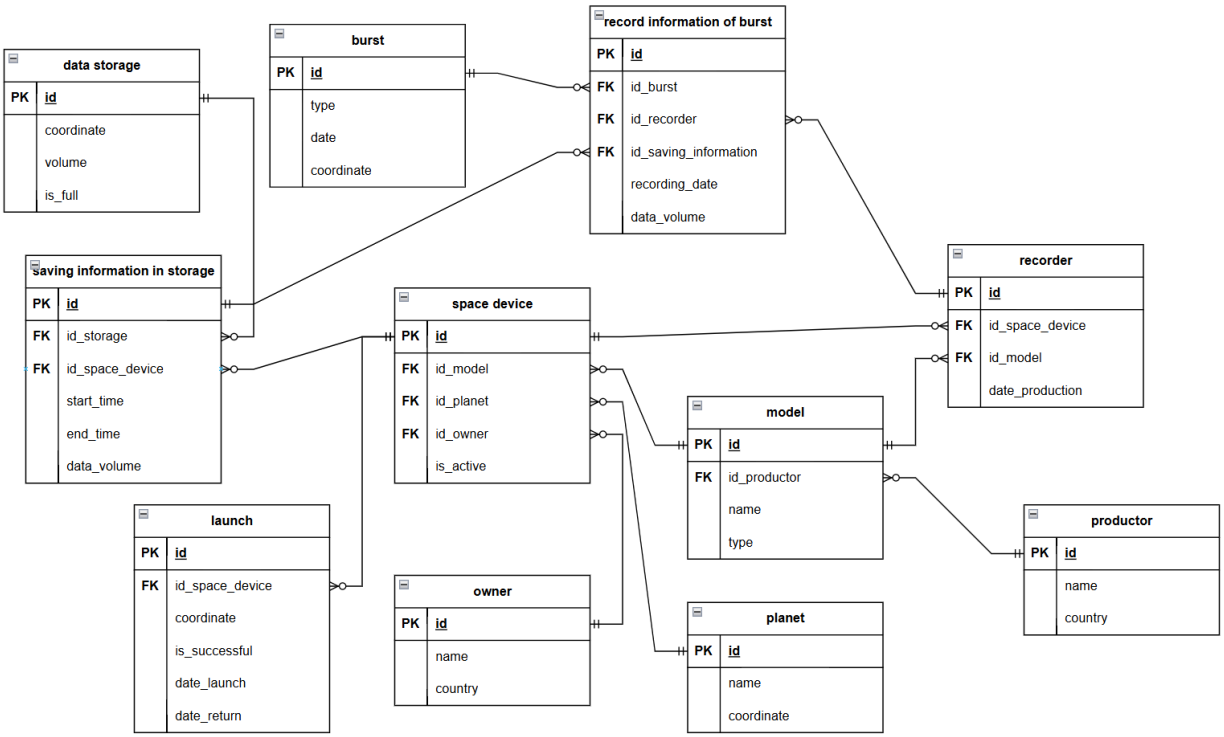
Save information in storage - Сохранить в хранилище информацию,отправленную из космоса

Характеристические:

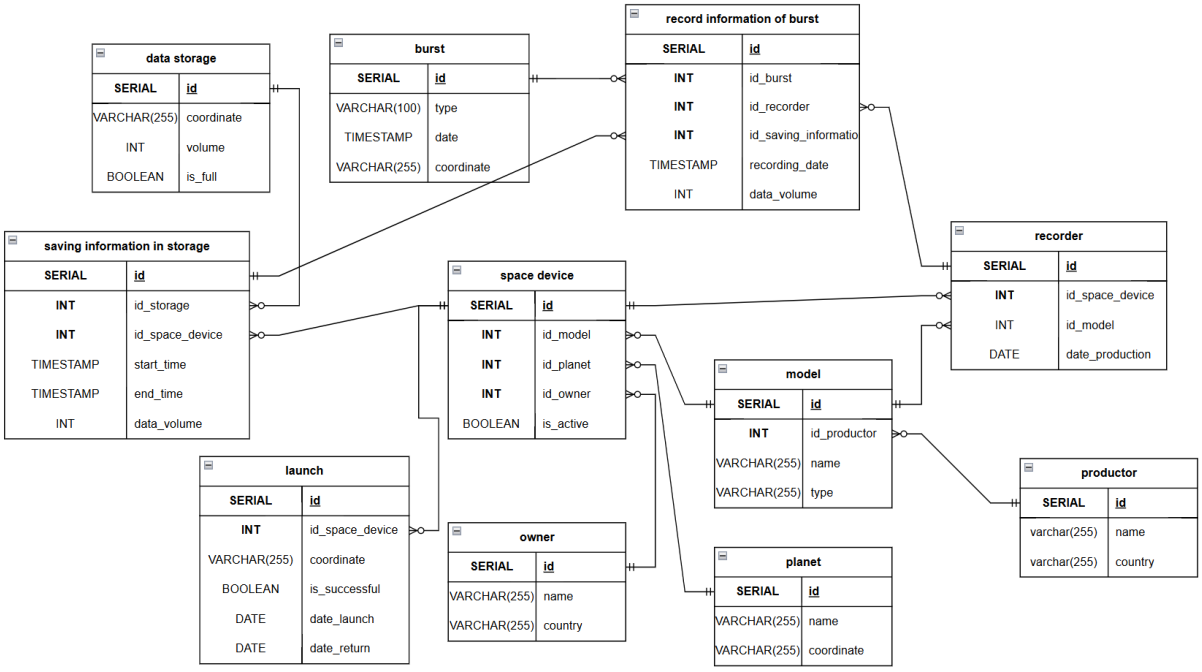
Space devices

Recorder

Инфологическая модель



Даталогическая модель



Создание модели в PostgreSQL

```
DROP TABLE IF EXISTS record_information_of_burst CASCADE;

DROP TABLE IF EXISTS recorder CASCADE;

DROP TABLE IF EXISTS saving_information_in_storage CASCADE;

DROP TABLE IF EXISTS burst CASCADE;

DROP TABLE IF EXISTS launch CASCADE;

DROP TABLE IF EXISTS space_device CASCADE;

DROP TABLE IF EXISTS data_storage CASCADE;

DROP TABLE IF EXISTS model CASCADE;

DROP TABLE IF EXISTS planet CASCADE;

DROP TABLE IF EXISTS owner CASCADE;

DROP TABLE IF EXISTS productor CASCADE;


CREATE TABLE productor (
    id SERIAL PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    country VARCHAR(255) NOT NULL
);


CREATE TABLE model (
    id SERIAL PRIMARY KEY,
    id_productor INT NOT NULL REFERENCES productor(id) ON DELETE CASCADE,
    name VARCHAR(255) NOT NULL,
    type VARCHAR(100) NOT NULL CHECK (type IN ('satellite', 'probe', 'telescope', 'rover', 'station'))
);


CREATE TABLE owner (
    id SERIAL PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    country VARCHAR(255) NOT NULL
```

```
);

CREATE TABLE planet (
    id SERIAL PRIMARY KEY,
    name VARCHAR(255) NOT NULL UNIQUE,
    coordinate VARCHAR(255) NOT NULL CHECK (coordinate ~ '^[A-Za-z0-9\-]+$')
);

CREATE TABLE space_device (
    id SERIAL PRIMARY KEY,
    id_model INT NOT NULL REFERENCES model(id) ON DELETE CASCADE,
    id_planet INT REFERENCES planet(id) ON DELETE SET NULL,
    id_owner INT NOT NULL REFERENCES owner(id) ON DELETE CASCADE,
    is_active BOOLEAN NOT NULL DEFAULT FALSE
);

CREATE TABLE launch (
    id SERIAL PRIMARY KEY,
    id_space_device INT NOT NULL REFERENCES space_device(id) ON DELETE CASCADE,
    coordinate VARCHAR(255) NOT NULL,
    is_successful BOOLEAN,
    date_launch DATE NOT NULL CHECK (date_launch <= CURRENT_DATE),
    date_return DATE CHECK (date_return IS NULL OR date_return >= date_launch)
);

CREATE TABLE data_storage (
    id SERIAL PRIMARY KEY,
    coordinate VARCHAR(255) NOT NULL CHECK (coordinate ~ '^[A-Za-z0-9\-]+$'),
    volume INT NOT NULL CHECK (volume > 0),
    is_full BOOLEAN NOT NULL DEFAULT FALSE
```

```
);

CREATE TABLE burst (
    id SERIAL PRIMARY KEY,
    type VARCHAR(100) NOT NULL CHECK (type IN ('gamma', 'x-ray', 'radio', 'optical', 'particle')),
    date TIMESTAMP NOT NULL CHECK (date <= CURRENT_TIMESTAMP),
    coordinate VARCHAR(255) NOT NULL
);

CREATE TABLE recorder (
    id SERIAL PRIMARY KEY,
    id_space_device INT NOT NULL REFERENCES space_device(id) ON DELETE CASCADE,
    id_model INT NOT NULL REFERENCES model(id) ON DELETE CASCADE,
    date_production DATE NOT NULL CHECK (date_production <= CURRENT_DATE)
);

CREATE TABLE saving_information_in_storage (
    id SERIAL PRIMARY KEY,
    id_storage INT NOT NULL REFERENCES data_storage(id) ON DELETE CASCADE,
    id_space_device INT NOT NULL REFERENCES space_device(id) ON DELETE CASCADE,
    start_time TIMESTAMP NOT NULL CHECK (start_time <= CURRENT_TIMESTAMP),
    end_time TIMESTAMP CHECK (end_time IS NULL OR end_time >= start_time),
    data_volume INT NOT NULL CHECK (data_volume > 0)
);

CREATE TABLE record_information_of_burst (
    id SERIAL PRIMARY KEY,
    id_burst INT NOT NULL REFERENCES burst(id) ON DELETE CASCADE,
    id_recorder INT NOT NULL REFERENCES recorder(id) ON DELETE CASCADE,
```

```
    id_saving_information INT NOT NULL REFERENCES saving_information_in_storage(id) ON DELETE
CASCADE,

    recording_date TIMESTAMP NOT NULL CHECK (recording_date <= CURRENT_TIMESTAMP),

    data_volume INT NOT NULL CHECK (data_volume > 0)

);


INSERT INTO productor (name, country) VALUES

('SpaceX', 'USA'),

('NASA', 'USA'),

('ESA', 'Europe'),

('Roscosmos', 'Russia');


INSERT INTO model (id_productor, name, type) VALUES

(1, 'Starlink', 'satellite'),

(2, 'Voyager', 'probe'),

(3, 'Hubble', 'telescope'),

(4, 'Lunokhod', 'rover');


INSERT INTO owner (name, country) VALUES

('US Gov', 'USA'),

('EU Space', 'Europe'),

('Russian Fed', 'Russia'),

('Private Corp', 'USA');


INSERT INTO planet (name, coordinate) VALUES

('Earth', 'SOL-3'),

('Mars', 'SOL-4'),

('Moon', 'SOL-3-1'),

('Venus', 'SOL-2');
```

```
INSERT INTO space_device (id_model, id_planet, id_owner, is_active) VALUES
(1, 1, 1, TRUE),
(2, NULL, 1, TRUE),
(3, NULL, 2, TRUE),
(4, 3, 3, FALSE);

INSERT INTO launch (id_space_device, coordinate, is_successful, date_launch, date_return) VALUES
(1, '28.5618N-80.5774W', TRUE, '2020-01-01', NULL),
(2, '28.5618N-80.5774W', TRUE, '1977-09-05', NULL),
(3, '5.2397N-52.7688W', TRUE, '1990-04-24', NULL),
(4, '45.9650N-63.3050E', TRUE, '1970-11-10', '1970-11-17');

INSERT INTO data_storage (coordinate, volume, is_full) VALUES
('SOL-3-001', 1000, FALSE),
('SOL-3-002', 2000, TRUE),
('SOL-4-001', 500, FALSE),
('SOL-3-003', 1500, FALSE);

INSERT INTO burst (type, date, coordinate) VALUES
('gamma', '2022-01-01 12:00:00', 'RA14h20m'),
('x-ray', '2022-02-01 12:00:00', 'RA18h45m'),
('radio', '2022-03-01 12:00:00', 'RA22h10m'),
('optical', '2022-04-01 12:00:00', 'RA5h30m');

INSERT INTO recorder (id_space_device, id_model, date_production) VALUES
(2, 2, '1976-01-01'),
(3, 3, '1989-01-01'),
(1, 1, '2019-01-01'),
(4, 4, '1970-01-01');
```

```
INSERT INTO saving_information_in_storage (id_storage, id_space_device, start_time, end_time, data_volume) VALUES

(1, 2, '2022-01-01 12:01:00', '2022-01-01 12:05:00', 100),

(2, 3, '2022-02-01 12:01:00', '2022-02-01 12:05:00', 200),

(3, 1, '2022-03-01 12:01:00', '2022-03-01 12:05:00', 150),

(4, 4, '2022-04-01 12:01:00', '2022-04-01 12:05:00', 180);


INSERT INTO record_information_of_burst (id_burst, id_recorder, id_saving_information, recording_date, data_volume) VALUES

(1, 1, 1, '2022-01-01 12:01:30', 50),

(2, 2, 2, '2022-02-01 12:01:30', 80),

(3, 3, 3, '2022-03-01 12:01:30', 60),

(4, 4, 4, '2022-04-01 12:01:30', 70);
```

Вывод

Я познакомился с базой данных PostgreSQL. Освежил свои знания языка SQL.