­­­Министерство образования и науки РФ

Федеральное государственное автономное

образовательное учреждение высшего образования

«Санкт-Петербургский национальный исследовательский университет

ИТМО»

**факультет программной инженерии и компьютерной техники**

**ЛАБОРАТОРНАЯ РАБОТА № 1**

по дисциплине

‘Базы данных’

Вариант №31188

*Выполнил:*

Студент группы P3118

Левченко Ярослав Алексеевич

*Преподаватель:*

Инячина Диана



Санкт-Петербург, 2023

**Задание:**

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 полдень он шел в камбуз готовить обед, передавая на это время управление кораблем ЭАЛу. Впрочем, даже и в камбузе он оставался в курсе всех событий на корабле, потому что в этой крохотной кабинке, - служившей одновременно и столовой, был установлен параллельный ситуационный экран-индикатор; кроме того, ЭАЛ мог в любой момент вызвать его к пульту. Фрэнк завтракал вместе с ним, а потом отправлялся на шесть часов спать в свою каюту. Обычно за обедом они смотрели какую-нибудь телевизионную программу с Земли, которую им передавали по специальному каналу.”

**Список сущностей:**

Стержневые:

* *Корабль* – id, название, тип\_корабля, связь с землей
* *Человек* – id, имя, фамилия, возраст, национальность, происхождение
* *Планета* – id, название, удаленность от солнца, масса, количество спутников

Ассоциации:

* *Специальный канал Связи* – корабль-планета (количество поддерживаемых каналов, время гарантии)
* *Работники* – корабль-человек (должность, дата начала работы)

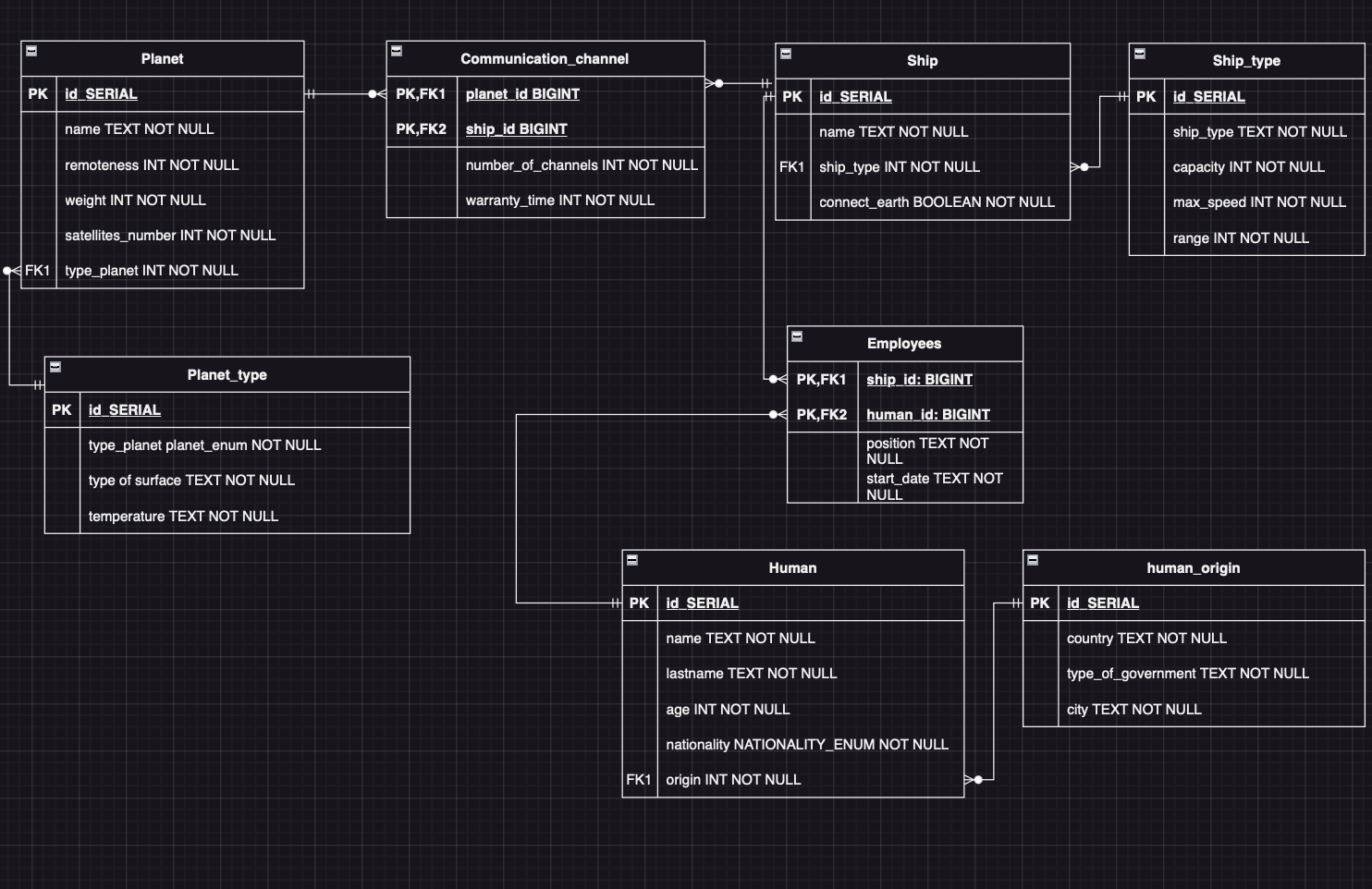
Характеристики:

* *Происхождение человека* – id, страна, город, тип правления
* *Тип корабля* – id, тип корабля, вместимость, скорость, расстояние

**Инфологическая модель:**

Изображение выглядит как текст, снимок экрана, диаграмма

Автоматически созданное описание

**Даталогическая модель:**

**Реализация на уровне PostgreSQL:**

-- dropping enum types

DROP TYPE IF EXISTS nationality\_enum CASCADE;

DROP TYPE IF EXISTS planet\_enum CASCADE;

-- dropping tables

DROP TABLE IF EXISTS Communication\_channel CASCADE;

DROP TABLE IF EXISTS Employees CASCADE;

DROP TABLE IF EXISTS Human CASCADE;

DROP TABLE IF EXISTS human\_origin CASCADE;

DROP TABLE IF EXISTS Planet CASCADE;

DROP TABLE IF EXISTS Planet\_type CASCADE;

DROP TABLE IF EXISTS Ship CASCADE;

DROP TABLE IF EXISTS Ship\_type CASCADE;

-- dropping domains

DROP DOMAIN IF EXISTS positive\_integer CASCADE;

DROP DOMAIN IF EXISTS positive\_decimal CASCADE;

DROP DOMAIN IF EXISTS max\_speed\_constraint CASCADE;

DROP DOMAIN IF EXISTS range\_constraint CASCADE;

-- creating enums

CREATE TYPE nationality\_enum AS ENUM ('American', 'British', 'Canadian', 'Chinese', 'French', 'German', 'Indian', 'Japanese', 'Russian', 'Spanish');

CREATE TYPE planet\_enum AS ENUM ('Rocky planets', 'Gas giants', 'Ice giants', 'Dwarf planets', 'Planets in the ocean', 'Interstellar planets', 'Brown', 'Spanish');

-- create domains

CREATE DOMAIN positive\_integer AS INTEGER CHECK (VALUE > 0);

CREATE DOMAIN positive\_decimal AS DECIMAL CHECK (VALUE > 0);

CREATE DOMAIN max\_speed\_constraint AS positive\_integer;

CREATE DOMAIN range\_constraint AS positive\_integer;

-- creating tables

CREATE TABLE IF NOT EXISTS Ship\_type (

id SERIAL PRIMARY KEY,

ship\_type TEXT NOT NULL,

capacity INTEGER NOT NULL,

max\_speed max\_speed\_constraint NOT NULL,

range range\_constraint NOT NULL

);

CCREATE TABLE IF NOT EXISTS Ship (

id SERIAL PRIMARY KEY,

name TEXT NOT NULL,

ship\_type INTEGER REFERENCES Ship\_type (id) NOT NULL,

connect\_earth BOOLEAN NOT NULL

);

CREATE TABLE IF NOT EXISTS Planet\_type (

id SERIAL PRIMARY KEY,

type\_planet planet\_enum NOT NULL,

type\_of\_surface TEXT NOT NULL,

temperature TEXT NOT NULL

);

CREATE TABLE IF NOT EXISTS Planet (

id SERIAL PRIMARY KEY,

name TEXT NOT NULL,

remoteness INTEGER NOT NULL,

weight INTEGER NOT NULL,

satelittes\_number INTEGER NOT NULL,

type\_planet\_id INTEGER REFERENCES Planet\_type (id) NOT NULL

);

CREATE TABLE IF NOT EXISTS Communication\_channel (

Ship\_id INTEGER REFERENCES Ship (id) NOT NULL,

Planet\_id INTEGER REFERENCES Planet (id) NOT NULL,

number\_of\_channels INTEGER NOT NULL,

warranty\_time INTEGER NOT NULL,

PRIMARY KEY (Ship\_id, Planet\_id)

);

CREATE TABLE IF NOT EXISTS human\_origin (

id SERIAL PRIMARY KEY,

country TEXT NOT NULL,

type\_of\_government TEXT NOT NULL,

city TEXT NOT NULL

);

CREATE TABLE IF NOT EXISTS Human (

id SERIAL PRIMARY KEY,

name TEXT NOT NULL,

lastname TEXT NOT NULL,

age INTEGER NOT NULL,

nationality nationality\_enum NOT NULL,

human\_origin\_id INTEGER REFERENCES human\_origin (id) NOT NULL

);

CREATE TABLE IF NOT EXISTS Employees (

Ship\_id INTEGER REFERENCES Ship (id) NOT NULL,

Human\_id INTEGER REFERENCES Human (id) NOT NULL,

position TEXT NOT NULL,

start\_date TEXT NOT NULL,

PRIMARY KEY (Ship\_id, Human\_id)

);

-- insert data into tables

INSERT INTO Ship\_type (ship\_type, capacity, max\_speed, range)

VALUES

('Cargo Ship', 500, 50, 1000),

('Passenger Ship', 200, 40, 800),

('Exploration Ship', 100, 60, 1500);

INSERT INTO Ship (name, ship\_type, connect\_earth)

VALUES

('Ship 1', 1, true),

('Ship 2', 2, false),

('Ship 3', 3, true);

INSERT INTO Planet\_type (type\_planet, type\_of\_surface, temperature)

VALUES

('Rocky planets', 'Solid', 'Varies'),

('Gas giants', 'Gas', 'High'),

('Ice giants', 'Ice and Gas', 'Very Cold'),

('Dwarf planets', 'Solid and Ice', 'Cold'),

('Planets in the ocean', 'Water', 'Moderate'),

('Interstellar planets', 'Varies', 'Varies'),

('Brown', 'Varies', 'Varies'),

('Spanish', 'Varies', 'Varies');

INSERT INTO Planet (name, remoteness, weight, satelittes\_number, type\_planet\_id)

VALUES

('Earth', 1, 100, 1, 1),

('Mars', 2, 90, 2, 1),

('Jupiter', 5, 1000, 79, 2);

INSERT INTO Communication\_channel (Ship\_id, Planet\_id, number\_of\_channels, warranty\_time)

VALUES

(1, 1, 10, 2),

(2, 1, 5, 1),

(3, 2, 8, 3);

INSERT INTO human\_origin (country, type\_of\_government, city)

VALUES

('USA', 'Democratic', 'New York'),

('Russia', 'Federal Republic', 'Moscow'),

('China', 'Communist', 'Beijing');

INSERT INTO Human (name, lastname, age, nationality, human\_origin\_id)

VALUES

('John', 'Doe', 30, 'American', 1),

('Ivan', 'Ivanov', 35, 'Russian', 2),

('Li', 'Chen', 28, 'Chinese', 3);

INSERT INTO Employees (Ship\_id, Human\_id, position, start\_date)

VALUES

(1, 1, 'Captain', '2022-01-01'),

(1, 2, 'Engineer', '2022-02-01'),

(2, 3, 'Pilot', '2022-03-01');

**Вывод:** во время выполнения лабораторной работы я ознакомился с архитектурой построения ANSI-SPARC и базовым синтаксисом PostgreSQL, научился создавать инфологические и даталогические диаграммы, enum’ы, ssh’а также создавать серверную базу данных и с ней взаимодействовать.