МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

## 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

«ИНФОРМАЦИОННЫЕ СИСТЕМЫ И БАЗЫ ДАННЫХ»

Вариант № 336805

***Выполнил:***

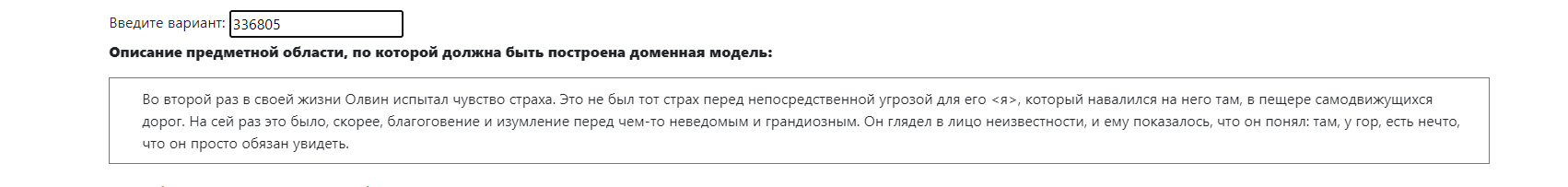
Студент группы P33312

Лысенко А.К.

***Преподаватель:***  Николаев В.В.

Санкт-Петербург, 2023

**Текст задания**



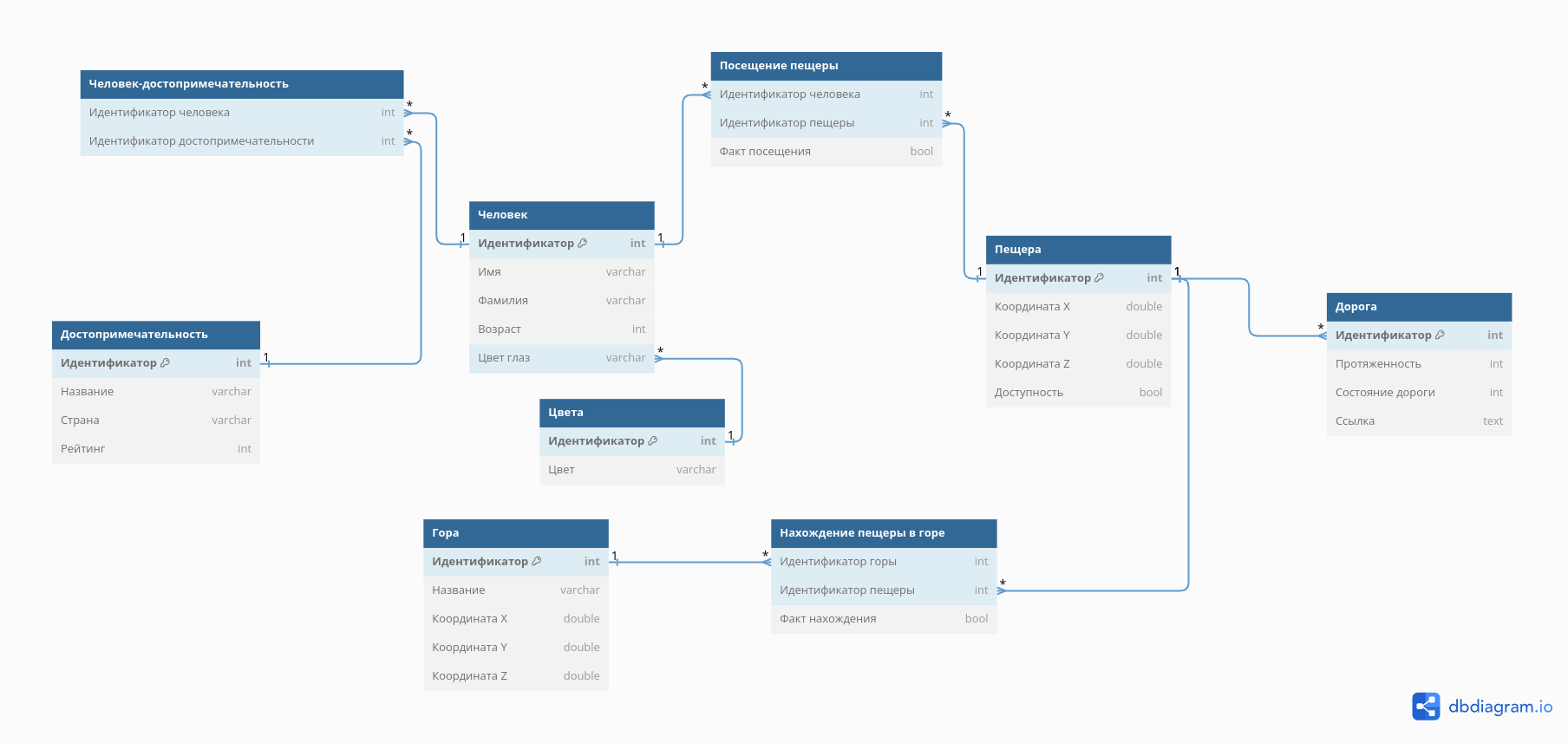
**Описание предметной области**

Прочитав данный текст, можно понять, что здесь есть такая сущность, как **Человек** – главный герой рассказа. Далее в тексте затрагивается такое место, как **пещера**. Это тоже выделяем в отдельную сущность. В конце упоминается **гора**, которую мы тоже выделяем в сущность. Помимо этого, можно сделать вывод о том, что **пещеру можно посещать**, а также мы можем заметить тот факт, что **пещера может находиться внутри горы**. Ну и чтобы понимать о какой горе или пещере идет речь, мы имеем их **координаты**. Также можно отметить такую сущность, как **дорога**, которая может находиться внутри пещеры. В тексте можно заметить выражение, “должен увидеть”, я интерпретировал это, как **достопримечательность.**

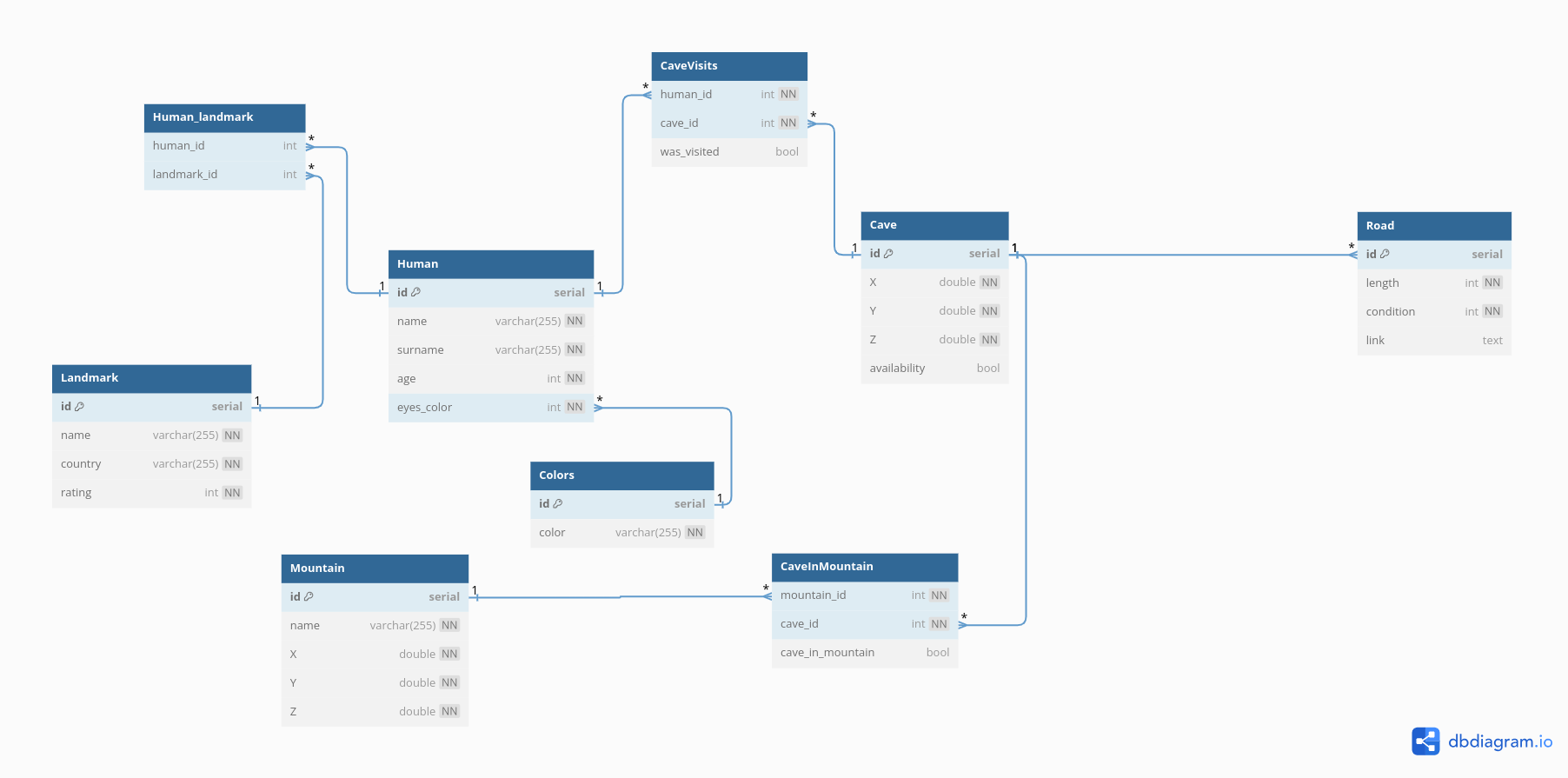
**Список сущностей и их классификация**

* **Стержневые**1) Человек – идентификатор, имя, фамилия, возраст.  
  2) Пещера – идентификатор, координата X, Y, Z, доступность пещеры.   
  3) Гора – идентификатор, название, координата X, Y, Z.  
  4) Достопримечательность – название, страна, рейтинг, ссылка на фото.  
  5) Дорога – протяженность, состояние дороги.
* **Ассоциативные**1) Посещение пещеры – идентификатор человека, идентификатор пещеры, факт посещения  
  2) Нахождение пещеры в горе – идентификатор горы, идентификатор пещеры, факт нахождения.   
  3) Человек-достопримечательность – таблица, которая показывает какие достопримечательности должны быть увидены людьми, и какие достопримечательности должны увидеть люди.
* **Характеристические**1) Цвет глаз – идентификатор цвета, название цвета.

**Инфологическая модель**



**Даталогическая модель**

****

**Реализация даталогической модели на SQL.**

**CREATE TABLE IF NOT EXISTS** landmark (  
 id **serial primary key not null**,  
 **name varchar**(255) **not null check** ( *length*(**name**) > 0 ),  
 country **varchar**(255) **not null check** ( *length*(country) > 0 ),  
 rating **int not null check** ( rating >= 0 **and** rating <= 5 ),  
 link **text**  
);  
  
**CREATE TABLE IF NOT EXISTS** colors (  
 id **serial primary key not null**,  
 color **varchar**(255) **not null**  
);  
  
**CREATE TABLE IF NOT EXISTS** human (  
 id **serial primary key not null**,  
 **name varchar**(255) **not null check** ( *length*(**name**) > 0 ),  
 surname **varchar**(255) **not null check** ( *length*(surname) > 0 ),  
 age **int not null check** ( age >= 0 **and** age <= 120),  
 eyes\_color **int not null**,  
 **foreign key** (eyes\_color) **references** colors(id)  
);  
  
**CREATE TABLE IF NOT EXISTS** human\_landmark (  
 human\_id **int not null**,  
 landmark\_id **int not null**,  
 **foreign key** (human\_id) **references** human(id),  
 **foreign key** (landmark\_id) **references** landmark(id),  
 **unique** (human\_id, landmark\_id)  
);  
  
**CREATE TABLE IF NOT EXISTS** mountain (  
 id **serial primary key not null**,  
 **name varchar**(255) **not null check** ( *length*(**name**) > 0 ),  
 X **double precision not null**,  
 **Y double precision not null**,  
 Z **double precision not null**  
  
);  
  
**CREATE TABLE IF NOT EXISTS** cave (  
 id **serial primary key not null**,  
 X **double precision not null**,  
 **Y double precision not null**,  
 Z **double precision not null**,  
 availability **bool**  
);  
  
**CREATE TABLE IF NOT EXISTS** road (  
 id **serial primary key not null**,  
 **length int not null**,  
 condition **int not null check** ( condition >= 0 **and** condition <= 5),  
 cave\_id **int**,  
 **foreign key** (cave\_id) **references** cave(id)  
);  
  
**CREATE TABLE IF NOT EXISTS** cave\_visits (  
 human\_id **int not null**,  
 cave\_id **int not null**,  
 was\_visited **bool**,  
 **foreign key** (human\_id) **references** human(id),  
 **foreign key** (cave\_id) **references** cave(id),  
 **unique** (human\_id, cave\_id)  
);  
  
**CREATE TABLE IF NOT EXISTS** cave\_in\_mountain (  
 mountain\_id **int not null**,  
 cave\_id **int not null**,  
 cave\_in\_mountain **bool**,  
 **foreign key** (mountain\_id) **references** mountain(id),  
 **foreign key** (cave\_id) **references** cave(id),  
 **unique** (mountain\_id, cave\_id)  
);

**insert into** landmark (**name**, country, rating, link) **values** ('Basilica of the Holy Family', 'Spain', 4, **null**);  
**insert into** landmark (**name**, country, rating, link) **values** ('Coliseum', 'Italy', 5, **null**);  
**insert into** landmark (**name**, country, rating, link) **values** ('Dubai Fountain', 'UAE', 3, **null**);  
**insert into** landmark (**name**, country, rating, link) **values** ('Empire State Building', 'USA', 5, **null**);  
**insert into** landmark (**name**, country, rating, link) **values** ('Cat memorial', 'Russia', 5, 'http://www.evpatori.ru/wp-content/uploads/2021/11/%D0%BA%D0%BE%D1%82-%D1%81-%D1%80%D1%8B%D0%B1%D0%BE%D0%B9.jpg');  
  
**insert into** colors (color) **values** ('black');  
**insert into** colors (color) **values** ('green');  
**insert into** colors (color) **values** ('blue');  
**insert into** colors (color) **values** ('brown');  
  
**insert into** human (**name**, surname, age, eyes\_color) **values** ('Artem', 'Lysenko', 19, 2);  
**insert into** human (**name**, surname, age, eyes\_color) **values** ('Nadezhda', 'Naumova', 22, 3);  
**insert into** human (**name**, surname, age, eyes\_color) **values** ('Ivan', 'Ivanov', 41, 1);  
  
**insert into** human\_landmark (human\_id, landmark\_id) **values** (1, 1);  
**insert into** human\_landmark (human\_id, landmark\_id) **values** (1, 4);  
**insert into** human\_landmark (human\_id, landmark\_id) **values** (1, 5);  
**insert into** human\_landmark (human\_id, landmark\_id) **values** (2, 5);  
  
**insert into** cave (x, **y**, z, availability) **values** (13.11341, 85.314, 94.1341, **true**);  
**insert into** cave (x, **y**, z, availability) **values** (13.11341, 85.314, 94.1341, **true**);  
**insert into** cave (x, **y**, z, availability) **values** (13.11341, 85.314, 94.1341, **true**);  
  
**insert into** mountain (**name**, x, **y**, z) **values** ('Jomolungma', 53.21414, 75.134, 3.40194);  
**insert into** mountain (**name**, x, **y**, z) **values** ('Chogori', 33.3144, 35.1134, 4.518);  
**insert into** mountain (**name**, x, **y**, z) **values** ('Kanchenjunga', 17.48481, 42.2131, 7.1384);  
**insert into** mountain (**name**, x, **y**, z) **values** ('Lhotse', 51.6422, 48.3123, 3.45141);  
  
**insert into** cave\_in\_mountain(mountain\_id, cave\_id, cave\_in\_mountain) **values** (1, 1, **true**);  
**insert into** cave\_in\_mountain(mountain\_id, cave\_id, cave\_in\_mountain) **values** (1, 2, **false**);  
**insert into** cave\_in\_mountain(mountain\_id, cave\_id, cave\_in\_mountain) **values** (2, 1, **true**);  
  
**insert into** cave\_visits(human\_id, cave\_id, was\_visited) **VALUES** (1, 1, **true**);  
**insert into** cave\_visits(human\_id, cave\_id, was\_visited) **VALUES** (1, 2, **true**);  
**insert into** cave\_visits(human\_id, cave\_id, was\_visited) **VALUES** (2, 1, **true**);  
  
  
  
  
**insert into** road ( **length**, condition, cave\_id) **values** (1000, 1, 1);  
**insert into** road ( **length**, condition, cave\_id ) **values** (500, 2, 1);  
**insert into** road ( **length**, condition, cave\_id ) **values** (783, 2, 2);  
**insert into** road ( **length**, condition, cave\_id ) **values** (1435, 3, 3);  
**insert into** road ( **length**, condition, cave\_id ) **values** (783, 1, **null**);

**Вывод**

В ходе выполнения лабораторной работы я научился различать инфологическую и даталогическую модель данных, научился создавать таблицы в базе данных, добавлять данные в таблице и получать данные из таблиц посредством запросов. Я познакомился с DDL, DML и СУБД PostgreSQL.