# УНИВЕРСИТЕТ ИТМО

## Факультет программной инженерии и компьютерной техники

Направление подготовки 09.03.04 Программная инженерия Дисциплина «Информационный системы и базы данных»

# Лабораторная работа №1

*Вариант 183*

Студент

*Кузнецов М. А.*

*P33131*

Преподаватель *Николаев В. В.*

Санкт-Петербург, 2022 г.

## 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.

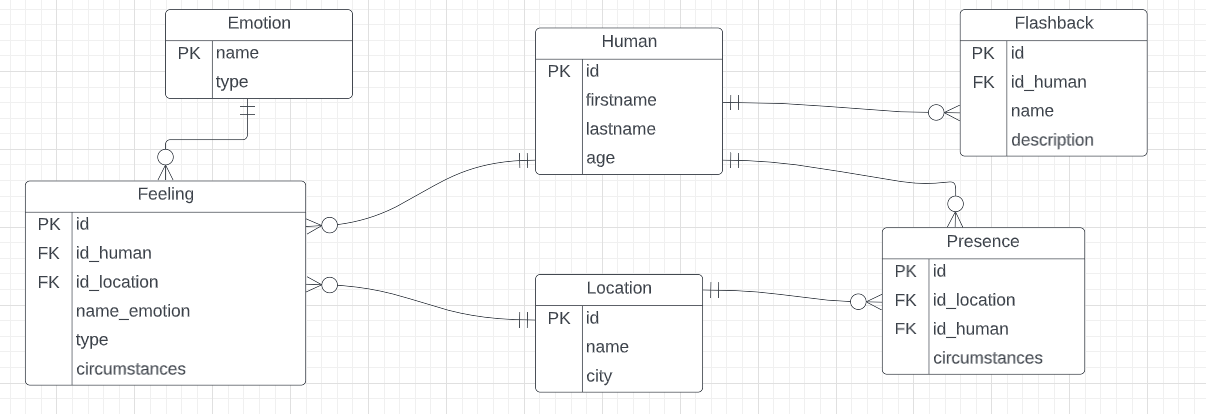
## Описание предметной области

Джизирак послушно проследовал за Ярланом Зеем в здание. Свой мозг в эти минуты он мог бы сравнить с губкой -- все впитывающей и ничего не подвергающей сомнению. Какое-то воспоминание или даже всего лишь отдаленное эхо воспоминания предупреждало его о том, что именно должно сейчас вот произойти, и он знал, что в былые времена при виде этого он сжался бы от ужаса. Теперь же он совсем не испытывал страха. Он не только сознавал себя под защитой понимания того, что все здесь происходящее -- нереально, но и присутствие Ярлана Зея казалось неким талисманом против любых опасностей, которые могли бы ему встретиться. Список сущностей

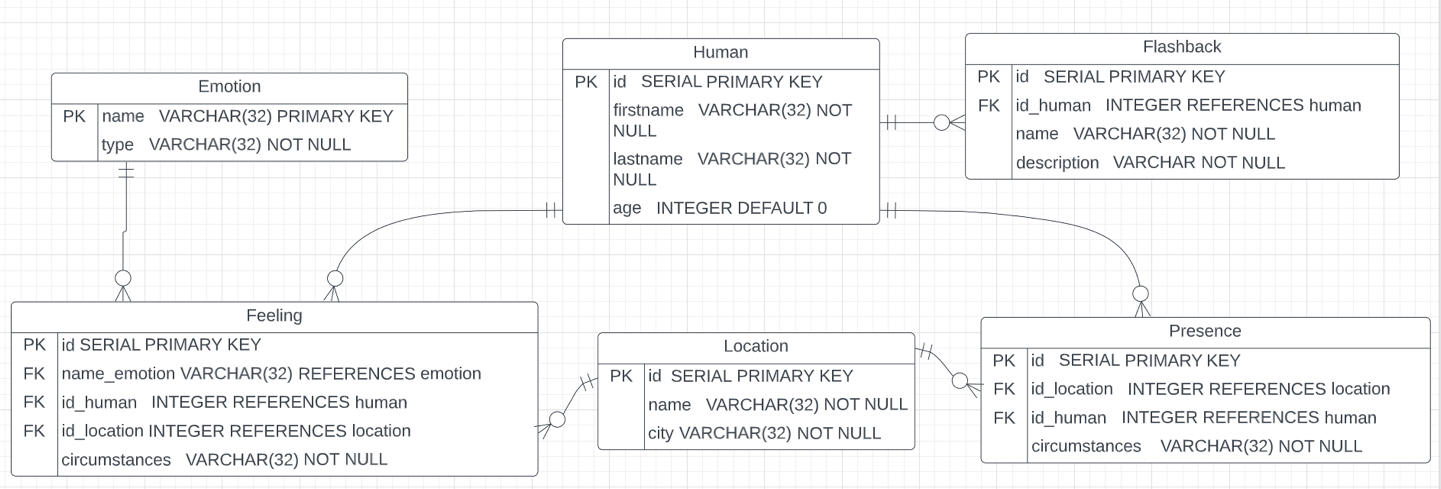
Стержневые:

* Локация – название
* Человек – *имя, фамилия, возраст*
* Эмоция – *название, тип (положительная, нейтральная, негативная)* Характеристические:
* Воспоминание *– название, кому принадлежит, о чем*
* *Нахождение – локация, кто находился, когда* Ассоциативные:
* Чувства – *кто, какие эмоции испытывает, когда*

## Инфологическая модель



## Даталогическая модель



Реализация модели на SQL

CREATE TABLE Human

(

ID SERIAL PRIMARY KEY,

Firstname VARCHAR(32) NOT NULL,

Lastname VARCHAR(32) NOT NULL,

Age INTEGER DEFAULT 0

);

CREATE TABLE Location

(

ID SERIAL PRIMARY KEY,

Name VARCHAR(32) NOT NULL,

City VARCHAR(32) NOT NULL

);

CREATE TABLE Emotion

(

Name VARCHAR(32) PRIMARY KEY,

Type VARCHAR(32) NOT NULL

);

CREATE TABLE Feeling

(

ID SERIAL PRIMARY KEY,

ID\_Emotion VARCHAR(32) REFERENCES Emotion,

ID\_Human INTEGER REFERENCES Human,

ID\_Location VARCHAR(32) NOT NULL,

Circumstances VARCHAR(32) NOT NULL

);

CREATE TABLE Flashback

(

ID SERIAL PRIMARY KEY,

Name VARCHAR(32) NOT NULL,

Description VARCHAR NOT NULL,

ID\_Human INTEGER REFERENCES Human

);

CREATE TABLE Presence

(

ID SERIAL PRIMARY KEY,

ID\_Location INTEGER REFERENCES Location,

ID\_Human INTEGER REFERENCES Human,

Circumstances VARCHAR(32) NOT NULL

);

INSERT INTO human (firstname, lastname, age)

VALUES ('Ярлан', 'Зей', 110),

('Джизирак', 'Каризижд', 40),

('Максим', 'Кузнецов', 19);

INSERT INTO location (name, city)

VALUES ('Капитолий', ‘Saint-P’),

('Университет ИТМО', ‘Saint-P’);

INSERT INTO presence (id\_location, id\_human, circumstances)

VALUES (1, 1, 'Беседа Ярлана Зея и Джизирака'),

(1, 2, 'Беседа Ярлана Зея и Джизирака');

INSERT INTO flashback (id\_human, name, description)

VALUES (2, 'Посещение Капитолия', 'Ужасные воспоминания о посещении

Джизираком Капитолия.');

INSERT INTO emotion (name, type)

VALUES ('Ужас', 'NEGATIVE'),

('Страх', 'NEGATIVE'),

('Сомнение', 'NEUTRAL');

INSERT INTO feeling (id\_human, id\_emotion, id\_location, circumstances)

VALUES (2, 1, 1, 'Во время беседы с Яраланом Зеем'),

(2, 2, 2, 'Во время беседы с Яраланом Зеем'),

(2, 3, 2, 'Во время беседы с Яраланом Зеем');

SQL-запрос: Название локации, в которой:

-- было больше всего людей старше 50 лет, испытавшие эмоцию «злости»,

-- если таких локаций несколько, то вывести локацию и имя человека, у которого наибольшее количество воспоминаний

Код запроса ниже, после идет разбор каждого важного момента:

|  |
| --- |
| CREATE OR REPLACE FUNCTION *containsTwoOrMore*() RETURNS table(res boolean)  AS $$ begin  return query select *count*(nc.name) > 1 from (select location.name, *count*(location.name) from location  join presence on location.id = presence.id\_location join human on human.id = presence.id\_human  join feeling on human.id = feeling.id\_human and location.id = feeling.id\_location  join emotion on feeling.id\_emotion = emotion.id where emotion.name = 'Злость' AND human.age > 50 group by location.name  ) as nc  where nc.count = (select *max*(n.count) from (select *count*(location.name) from location  join presence on location.id = presence.id\_location join human on human.id = presence.id\_human  join feeling on human.id = feeling.id\_human and location.id = feeling.id\_location  join emotion on feeling.id\_emotion = emotion.id where emotion.name = 'Злость' AND human.age > 50 group by location.name) as n); end;  $$  LANGUAGE plpgsql;      CREATE OR REPLACE FUNCTION *ifOne*() RETURNS text  AS $$ begin  return (select lphfen.name from(select location.name from location join presence on location.id = presence.id\_location join human on human.id = presence.id\_human  join feeling on human.id = feeling.id\_human and location.id = feeling.id\_location  join emotion on feeling.id\_emotion = emotion.id where emotion.name = 'Злость' AND human.age > 50) as lphfen group by lphfen.name); end;  $$ |
| LANGUAGE plpgsql;  CREATE OR REPLACE FUNCTION *ifMore*() RETURNS table(locname varchar(32), fname varchar(32), lname varchar(32))  AS $$ begin  return query (select name, firstname, lastname from (select lphfef.name, lphfef.firstname, lphfef.lastname, *count*(lphfef.id\_human) from (select location.name, human.firstname, human.lastname, flashback.id\_human from location  join presence on location.id = presence.id\_location  join human on human.id = presence.id\_human  join feeling on human.id = feeling.id\_human and location.id = feeling.id\_location  join emotion on feeling.id\_emotion = emotion.id join flashback on human.id = flashback.id\_human where emotion.name = 'Злость' AND human.age > 50) as lphfef  group by lastname, firstname, name) as l order by l.count desc limit 1); end;  $$  LANGUAGE plpgsql;  do $$ BEGIN  if (select res from *containsTwoOrMore*()) is true then create table result\_table as select \* from *ifmore*(); else  create table result\_table as select \* from *ifone*(); end if; end;  $$;  drop table result\_table; select \* from result\_table; |

Разберем по пунктам:

|  |
| --- |
| CREATE OR REPLACE FUNCTION *containsTwoOrMore*() RETURNS table(res boolean)  AS $$ begin  return query select *count*(nc.name) > 1 from (select location.name, *count*(location.name) from location  join presence on location.id = presence.id\_location join human on human.id = presence.id\_human  join feeling on human.id = feeling.id\_human and location.id = feeling.id\_location  join emotion on feeling.id\_emotion = emotion.id where emotion.name = 'Злость' AND human.age > 50 group by location.name  ) as nc  where nc.count = (select *max*(n.count) from (select *count*(location.name) from location  join presence on location.id = presence.id\_location join human on human.id = presence.id\_human  join feeling on human.id = feeling.id\_human and location.id = feeling.id\_location  join emotion on feeling.id\_emotion = emotion.id where emotion.name = 'Злость' AND |

human.age > 50 group by location.name) as n); end;

$$

LANGUAGE plpgsql;

Здесь мы создаем функцию containsTwoOrMore(), которая поможет нам определить, одна ли найденная локация под подходящие условия, или таких две и более. Мы join-им таблицы, чтобы в итоге можно было их проанализировать и отобрать нужных людей.

CREATE OR REPLACE FUNCTION *ifOne*() RETURNS text

AS $$

begin

return (select lphfen.name from(select location.name from location join presence on location.id = presence.id\_location join human on human.id = presence.id\_human

join feeling on human.id = feeling.id\_human and location.id = feeling.id\_location join emotion on feeling.id\_emotion = emotion.id where emotion.name = 'Злость' AND human.age > 50) as lphfen group by lphfen.name); end;

$$

LANGUAGE plpgsql;

Функция ifOne() будет вызываться, если локация такая нашлась только одна. Тогда просто выведем название локации (Пример будет ниже для данного случая).

varchar(32), lname varchar(32))

AS $$

begin

return query (select name, firstname, lastname from (select lphfef.name, lphfef.firstname, lphfef.lastname, *count*(lphfef.id\_human) from

(select location.name, human.firstname, human.lastname, flashback.id\_human from location join presence on location.id = presence.id\_location join human on human.id = presence.id\_human

join feeling on human.id = feeling.id\_human and location.id = feeling.id\_location join emotion on feeling.id\_emotion = emotion.id join flashback on human.id = flashback.id\_human where emotion.name = 'Злость' AND human.age > 50) as lphfef group by lastname, firstname, name) as l order by l.count desc limit 1); end;

$$

LANGUAGE plpgsql;

Аналогично, функция ifMore() будет вызываться, если локаций было две и более. Разные функции нам нужны, так как по заданию нужно вывести совершенно другое для данного случая (пример ниже).

do $$

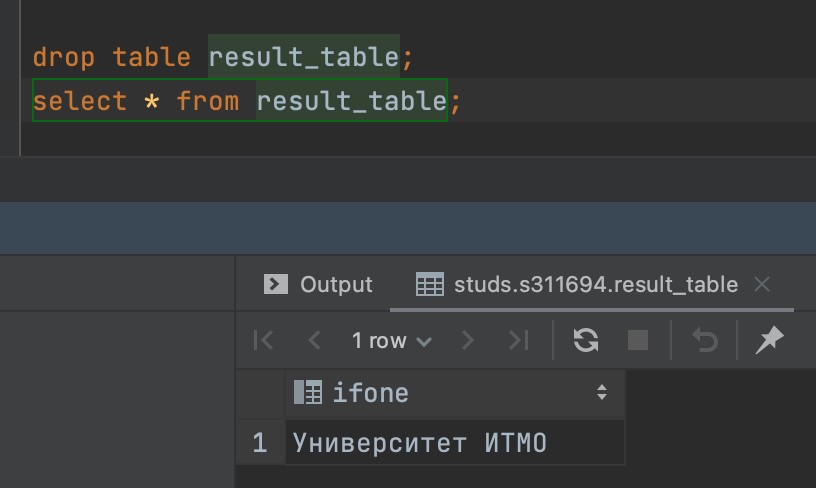
BEGIN

if (select res from *containsTwoOrMore*()) is true then create table result\_table as select \* from *ifmore*(); else

create table result\_table as select \* from *ifone*(); end if; end; $$;

Ad-hoc, который запускает «запрос». Простой if, который в зависимости от результата вызовет определенную функцию.

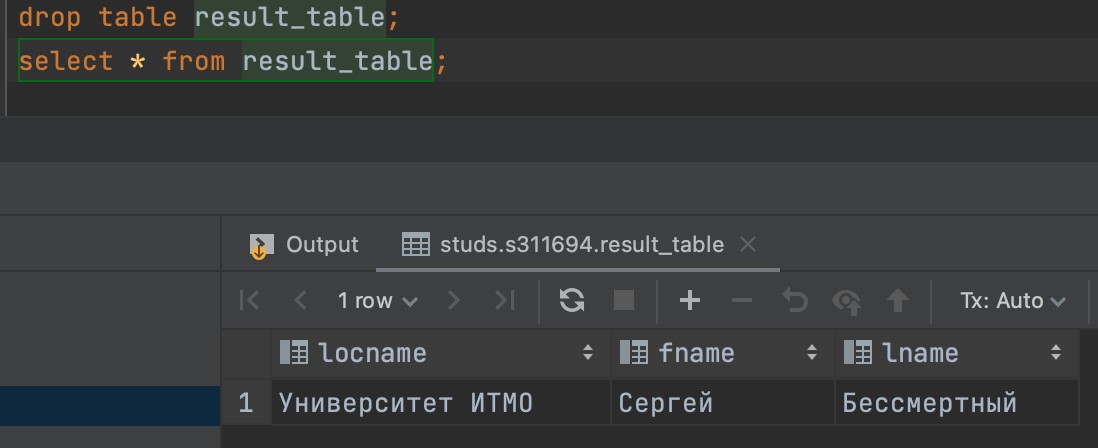
Примеры работы для ifOne() и ifMore(): ifOne():



Теперь добавим еще одну локацию, чтобы таких локаций было две:



И вызовем еще раз код (прим.: вывелся Университет ИТМО, а не Капитолий, так как у Сергея Бессмертного больше воспоминаний) ifMore()



## Выводы

При выполнении лабораторной работы я:

* Изучил базовые команды работы с SQL
* Спроектировал примерную модель базы данных по заданной предметной области
* Попрактиковался в реализации инфо- и даталогических моделей