Санкт-Петербургский Национальный Исследовательский Университет ИТМО

Факультет Программной Инженерии и Компьютерной Техники



Лабораторная работа **№3**

По дисциплине

**Базы Данных**

*Выполнил*:

Студент группы P3116

Воронов Григорий

Алексеевич

*Преподаватель*:

Гаврилов Антон

Валерьевич

Николаев Владимир

Вячеславович

Санкт-Петербург, 2024 г.

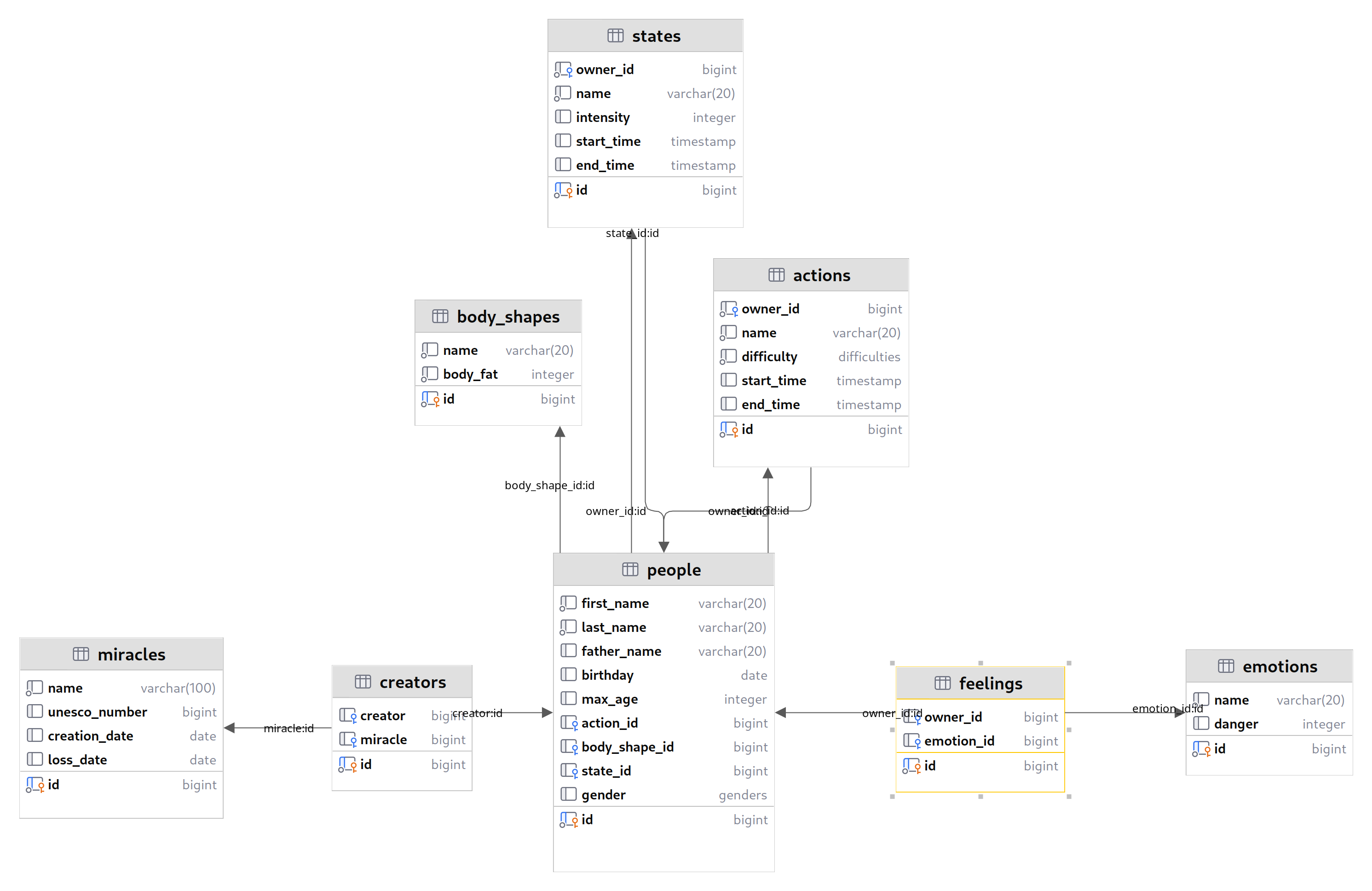
1. **Текст задания**

Для отношений, полученных при построении предметной области из лабораторной работы №1, выполните следующие действия:

* опишите функциональные зависимости для отношений полученной схемы (минимальное множество);
* приведите отношения в 3NF (как минимум). Постройте схему на основе 3NF (как минимум). Постройте схему на основе полученных отношений;
* опишите изменения в функциональных зависимостях, произошедшие после преобразования в 3NF (как минимум). Постройте схему на основе 3NF;
* преобразуйте отношения в BCNF. Докажите, что полученные отношения представлены в BCNF;

Если ваша схема находится уже в BCNF, докажите это.

Какие денормализации будут полезны для вашей схемы? Приведите подробное описание;

Придумайте функцию, связанную с вашей предметной областью, согласуйте ее с преподавателем и реализуйте на языке PL/pgSQL.

1. **Функциональные зависимости**

people: id → (first\_name, last\_name, father\_name, birthday, max\_age, action\_id, body\_shape\_id, state\_id, gender)

miracles: id → (name, unesco\_number, creation\_date, loss\_date)

emotions: id → (name, danger)

feelings: id → (owner\_id, emotion\_id)

creators: id → (creator, miracle)

states: id → (owner\_id, name, intensity, start\_time, end\_time)

body\_shapes: id → (name, body\_fat)

actions: id → (owner\_id, name, difficulty, start\_time, end\_time)

1. **Нормальные формы**

**Таблица находится в 1НФ** <=> ни одна из строк не содержит в любом своем поле более одного значения и ни одно из ее ключевых полей не пусто. *Моя модель удовлетворяет 1НФ, поскольку все атрибуты атомарны, и нет повторяющихся групп.*

**Таблица находится в 2НФ** <=> удовлетворяет 1НФ и все поля, не входящие в первичный ключ, связаны полной функциональной зависимостью с первичным ключом. *Моя модель удовлетворяет 2НФ, поскольку все неключевые атрибуты полностью функционально зависят от первичных ключей.*

**Таблица находится в 3НФ** <=> удовлетворяет 2НФ и ни одно из ее неключевых полей не зависит функционально от любого другого неключевого поля. *Моя модель удовлетворяет 2НФ, поскольку все неключевые атрибуты функционально зависят только от первичных ключей.*

1. **Нормальная форма Бойса-Кодда**

**Таблица находится в НФБК** <=> ключевые атрибуты составного ключа не должны зависеть от неключевых атрибутов. *Модель удовлетворяет НФБК*

1. **Денормализация**

**Добавление избыточных атрибутов**:

В некоторых случаях добавление избыточных атрибутов может улучшить производительность запросов.

Например, если часто запрашивается количество чудес, созданных человеком, то разумно добавить атрибут miracle\_count, дабы сократить время выполнения такого запроса. Однако нужно помнить, что в таком случае необходимо обновлять такой атрибут при каждом добавлении или удалении чуда.

Или же если часто запрашивается текущее состояние человека, то можно добавить current\_state в таблицу people.

**Объединение связанных таблиц**:

В некоторых случаях, объединение таблиц может уменьшить количество операций JOIN и ускорить обработку запросов. Например, если часто запрашиваются данные о человеке и его действии, то можно объединить эти две таблицы в people\_actions. Однако при добавлении или изменении записей о действиях в таблице actions, соответствующие данные в таблице people\_actions также должны быть обновлены.

1. **Функция на языке PL/pgSQL**

Функция на языке PL/pgSQL для вычисления среднего возраста всех людей.

CREATE OR REPLACE FUNCTION update\_average\_age()

RETURNS TRIGGER AS $$

DECLARE

avg\_age NUMERIC;

BEGIN

-- Вычисляем средний возраст всех людей

SELECT AVG(EXTRACT(YEAR FROM age(current\_date, p.birthday)))

INTO avg\_age

FROM people p;

-- Выводим результат на экран

RAISE NOTICE 'Average age of people: %', avg\_age;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER update\_average\_age\_trigger

AFTER INSERT OR UPDATE OF birthday ON people

FOR EACH ROW

EXECUTE FUNCTION update\_average\_age();

1. **Вывод**

При выполнении лабораторной работы я познакомился с понятием нормализации и денормализации. Научился определять функциональные зависимости модели, а также анализировать последнюю на соответствие различным нормальным формам. Познакомился с процедурным языком PL/pgSQL. Изучил эффективные способы денормализации схемы базы данных и ситуации, в которых возможно их применение.