МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«Национальный исследовательский университет ИТМО»

ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ И
КОМПЬЮТЕРНОЙ ТЕХНИКИ

ЛАБОРАТОРНАЯ РАБОТА №3

по дисциплине

«Базы данных»

Выполнил:
Студент группы P3107
Чусовлянов Максим Сергеевич
Преподаватель:
Байрамова Хумай Бахруз Кызы

Задание

Лабораторная работа #3

Задание.

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 основеNF (как минимум).
- Опишите изменения в функциональных зависимостях, произошедшие после преобразования в 3NF (как минимум). Постройте схему на основеNF;
- 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.

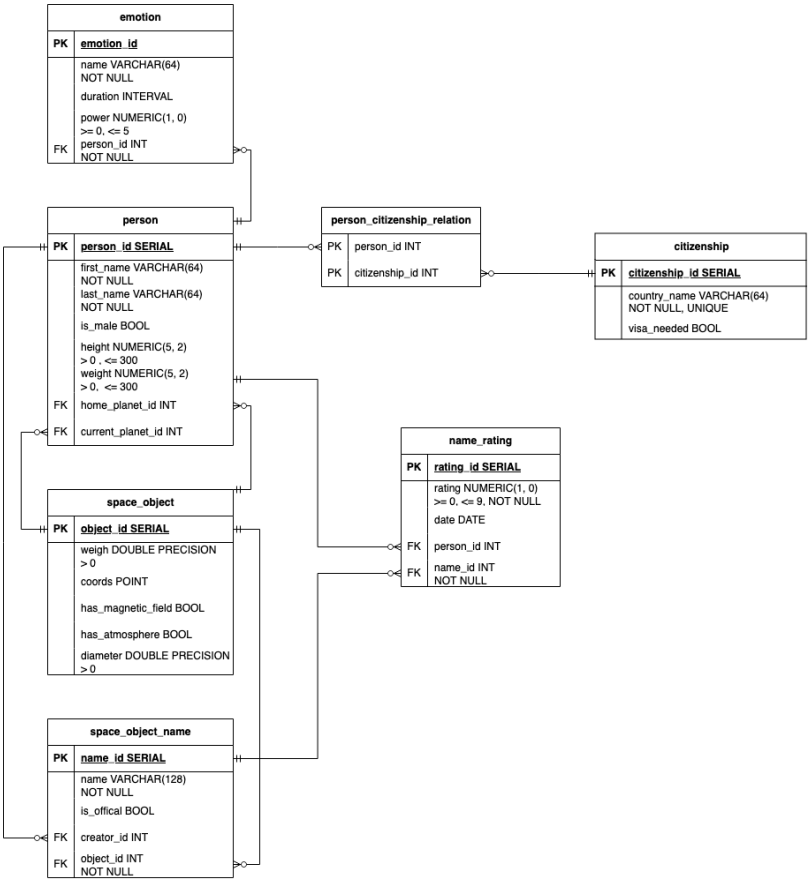
Придумайте триггер и связанную с ним функцию, относящиеся к вашей предметной области, согласуйте их с преподавателем и реализуйте на языке PL/pgSQL.

Отчёт по лабораторной работе должен содержать:

1. Текст задания.
2. Исходная, нормализованная и денормализованная модели.
3. Ответы на вопросы, представленные в задании.
4. Функция и триггер на языке PL/pgSQL
5. Выводы по работе.

Темы для подготовки к защите лабораторной работы:

1. Нормализация. Формы
2. Функциональные зависимости. Виды
3. Денормализация
4. Язык PL/pgSQL

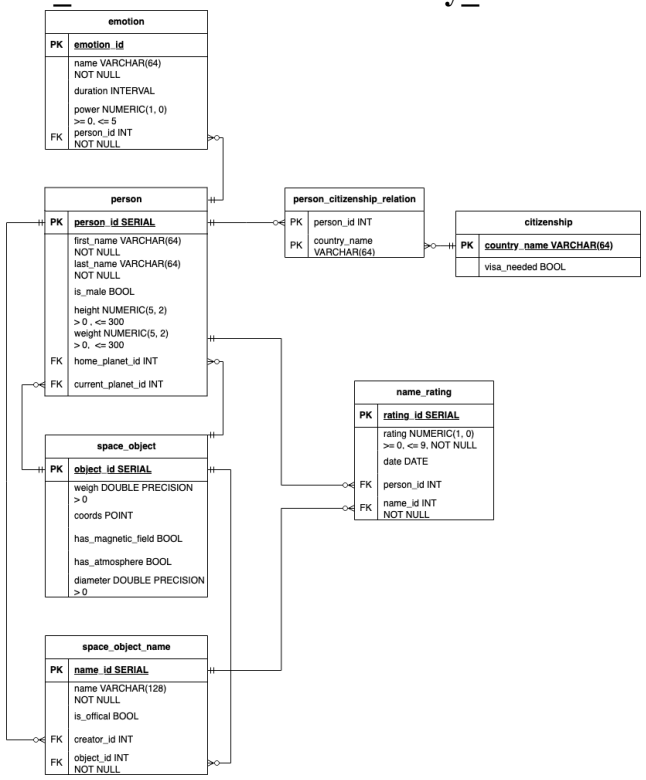


Функциональные зависимости

- **emotion**: (emotion_id) -> (name, duration, power, person_id)
- **person**: (person_id) -> (first_name, last_name, is_male, height, weight, home_planet_id, current_planet_id)
- **citizenship**: (citizenship_id) -> (country_name, visa_needed)
  (country_name) -> (visa_needed)
- **person_citizenship_relation**: (person_id, citizenship_id) -> ()
- **space_object**: (object_id) -> (weight, coords, has_magnetic_field, has_atmosphere, diameter)
- **space_object_name**: (name_id) -> (name, is_official, creator_id, object_id)
- **name_rating**: (rating_id) -> (rating, date, person_id, name_id)

Нормальные формы

- **1NF**: Отношение находится в 1NF, если все его атрибуты содержат только атомарные значения и отсутствуют повторяющиеся группы. Мои отношения удовлетворяет 1NF, так как все атрибуты атомарны, и нет повторяющихся групп.
- **2NF**: Отношение находится в 2NF, если оно находится в 1NF и все его неключевые атрибуты полностью функционально зависят от первичного ключа. Моя модель удовлетворяет 2NF, так как все неключевые атрибуты полностью функционально зависят от первичных ключей.
- **3NF**: Отношение находится в 3NF, если оно находится во 2NF и не содержит транзитивных зависимостей. Моя модель не удовлетворяет 3NF, так как **visa_needed** зависит от **country_name** в таблице **citizenship.**



- **emotion**: (emotion_id) -> (name, duration, power, person_id)
- **person**: (person_id) -> (first_name, last_name, is_male, height, weight, home_planet_id, current_planet_id)
- **citizenship**: (country_name) -> (visa_needed)
- **person_citizenship_relation**: (person_id, country_name) -> ()
- **space_object**: (object_id) -> (weight, coords, has_magnetic_field, has_atmosphere, diameter)
- **space_object_name**: (name_id) -> (name, is_official, creator_id, object_id)
- **name_rating**: (rating_id) -> (rating, date, person_id, name_id)

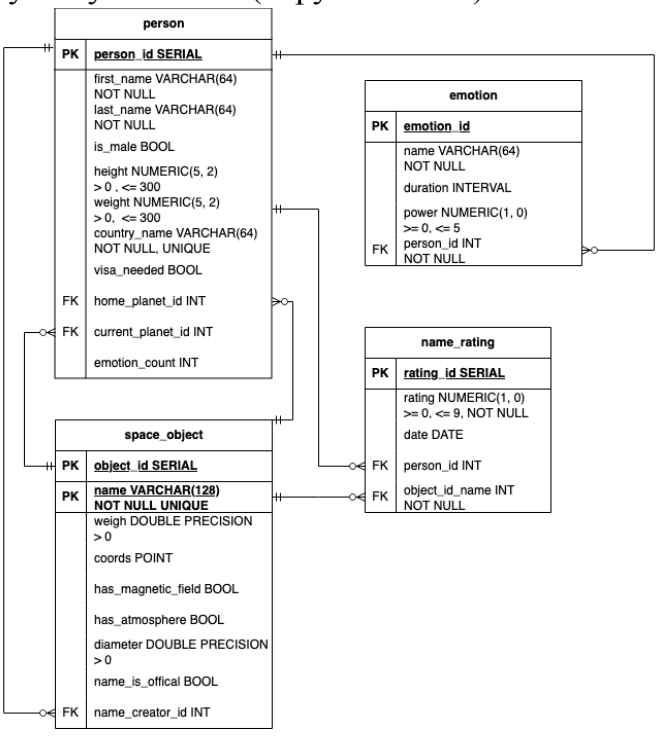
BCNF

- Отношение находится в BCNF, если для каждой функциональной зависимости X →Y, X является суперключом. Моя модель удовлетворяет BCNF, так как для всех функциональных зависимостей X является суперключом.

Денормализация

- **Объединение связанных таблиц**: В некоторых случаях, объединение таблиц может уменьшить количество операций JOIN, те уменьшить время обработки запросов. В моей схеме, можно рассмотреть объединение таблиц **space_object_name** и **space_object** в случае, если часто запрашиваются космические объекты по именам (нарушает 2NF). Объединение **person** с таблицей **citizenship** для избежания джойнов при выборке данных о гражданстве человека (нарушает 2NF).

- **Добавление избыточных атрибутов:** В некоторых случаях можно улучшить производительность благодаря добавлению избыточных атрибутов. Например добавить атрибут **emotion_count** в **person** для подсчета количества чувств которые человек чувствует сейчас (нарушает 2NF).



- **person**: (person_id) -> (first_name, last_name, is_male, height, weight, home_planet_id, current_planet_id, country_name, visa_needed, emotion_count)
  (country_name) -> (visa_needed)
- **space_object**: (object_id) -> (weight, coords, has_magnetic_field, has_atmosphere, diameter)
  (name) -> (name_is_official, name_creator_id)
- **name_rating**: (rating_id) -> (rating, date, person_id, object_id_name)

Триггер

Триггер при добавлении новой оценки имени обновляется рейтинг самых активных оценщиков имен. Если человек попадает в топ-10, то у него добавляется эмоции “Счастье” на 1 день. Если после добавление записей человек пропадает из топа, то у него добавляется эмоция “Тильт” на неделю. Для хранения топа используется отдельная табличка.

```
-- Триггер для обновления топа оценщиков и добавления эмоций
CREATE TABLE top_name_raters (
	person_id INT PRIMARY KEY,
	rating_count INT NOT NULL
);

CREATE OR REPLACE FUNCTION update_top_name_raters()
RETURNS TRIGGER AS $$
DECLARE
	current_count INT;
	new_top_count INT;
	person_id_to_delete INT;
BEGIN
	-- Увеличиваем счетчик оценок для человека
	SELECT COUNT(*) INTO current_count FROM name_rating WHERE person_id = NEW.person_id;

	-- Если человек уже в топе, обновляем его счетчик
	IF EXISTS (SELECT 1 FROM top_name_raters WHERE person_id = NEW.person_id) THEN
	UPDATE top_name_raters SET rating_count = current_count WHERE person_id = NEW.person_id;
	ELSE
	-- Вставляем новый счетчик, если человека нет в топе
	INSERT INTO top_name_raters (person_id, rating_count) VALUES (NEW.person_id, current_count);
	INSERT INTO emotion (name, duration, power, person_id) VALUES ('Счастье', INTERVAL '1 day', 5, NEW.person_id);
	RAISE NOTICE 'Человек с id=% добавлен в топ-10, он счастлив 1 день', NEW.person_id;
	END IF;

	-- Получаем айди человека, который вылетает из топа
	SELECT person_id FROM top_name_raters ORDER BY (rating_count, person_id) DESC OFFSET 11 LIMIT 1 INTO person_id_to_delete;
```

```
        -- Если человек существует, то удаляем его
        IF person_id_to_delete IS NOT NULL THEN
        RAISE NOTICE 'Человек с id=% удален из топа, он впал в тильт на 7 дней',
person_id_to_delete;
        RAISE NOTICE ' ';
        DELETE FROM top_name_raters WHERE person_id = person_id_to_delete;
        END IF;

        RETURN NEW;
    END;
    $$ LANGUAGE plpgsql;

    CREATE TRIGGER trigger_update_top_name_raters
    AFTER INSERT ON name_rating
    FOR EACH ROW
    EXECUTE FUNCTION update_top_name_raters();
```

Заключение

Во время выполнения лабораторной работы я познакомился с процессами нормализации и денормализации. Научился анализировать схему и находить в ней узкие места. Узнал что такое триггер и потренировался писать свои реализации триггеров.