Федеральное государственное автономное образовательное учреждение

высшего образования «Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники

Направление подготовки: 09.03.04 Программная инженерия

**Отчет**

**по лабораторной работе №3**

**по дисциплине «Базы данных»**

Вариант: 489568

Выполнила

Махнева Ирина Александровна,

Группа Р3117

Преподаватель

Чупанов Аликылыч Алибекович

г. Санкт-Петербург, 2025г

Оглавление

[Текст задания 3](#_Toc196084520)

[Даталогическая модель 4](#_Toc196084521)

[Функциональные зависимости (исходные) 4](#_Toc196084522)

[1НФ 6](#_Toc196084523)

[2НФ 6](#_Toc196084524)

[3НФ 6](#_Toc196084525)

[BCNF 6](#_Toc196084526)

[Денормализация 6](#_Toc196084527)

[Триггер 7](#_Toc196084528)

[Функция и триггер на языке PL/pgSQL 8](#_Toc196084529)

[Вывод 9](#_Toc196084530)

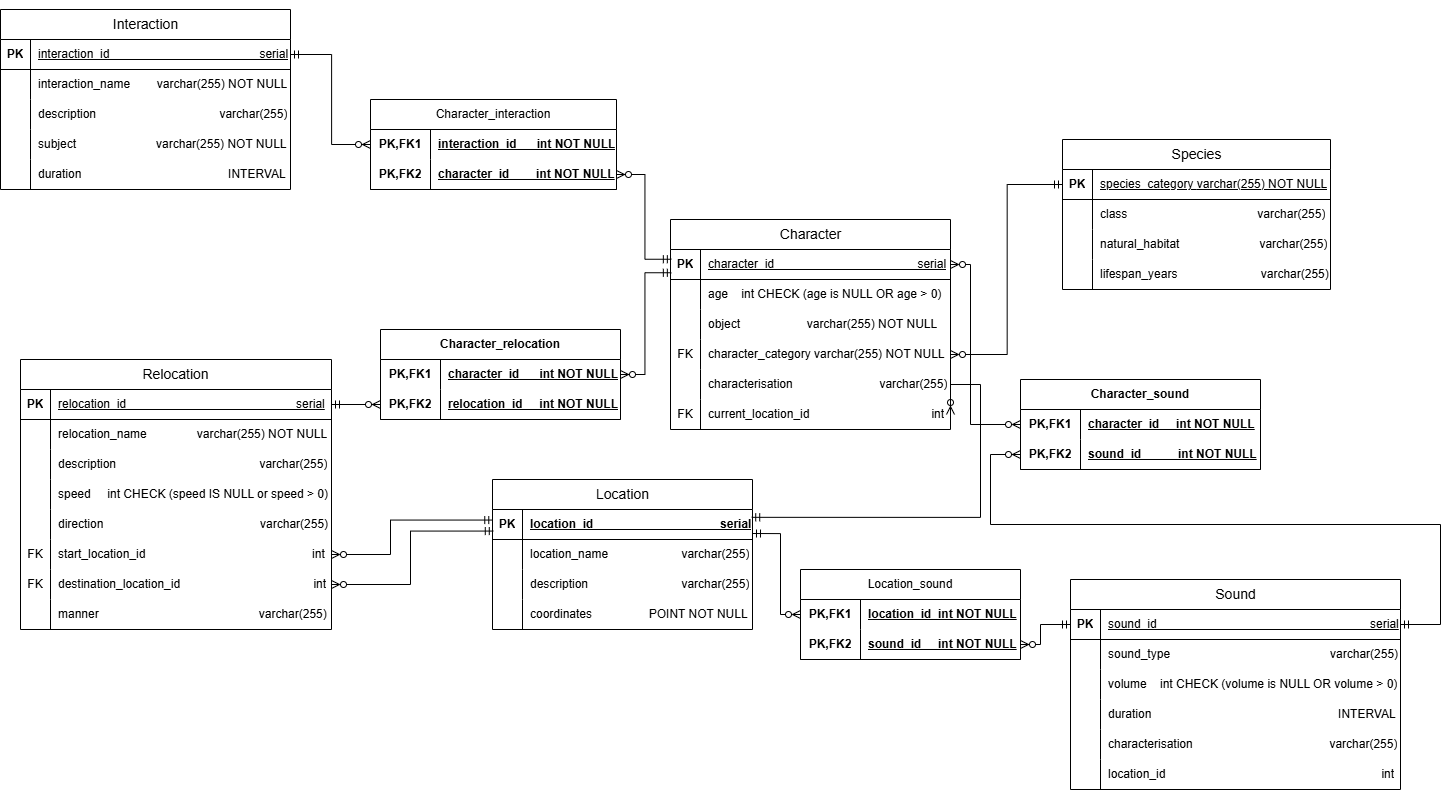
# Текст задания

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 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.

# Даталогическая модель



## Функциональные зависимости (исходные)

**Character:**

character\_id → (age, object, character\_category, characterization, current\_location\_id)

**Character\_interaction:**

(interaction\_id, character\_id) → ()

**Interaction:**

interaction\_id → (interaction\_name, description, subject, duration)

**Character\_relocation:**

(interaction\_id, relocation\_id) → ()

**Relocation:**

relocation\_id → (relocation\_name, description, speed, direction, start\_location\_id, destination\_location\_id, manner)

**Location:**

location\_id → location\_name, description, coordinates

**Location\_sound:**

(location\_id, sound\_id) → ()  
**Sound:**

sound\_id → (sound\_type, volume, duration, characterization, location\_id)

**Character\_sound:**

(character\_id, sound\_id) → ()

**Species:**

species\_category → (class, natural\_habitat, lifespan\_years)

# 1НФ

Отношение находится в 1НФ, если на пересечении каждой строки и столбца одно значение. Это требование уже выполняется для всех таблиц.

# 2НФ

Отношение находится в 2НФ, если оно уже находится в 1НФ, а также атрибуты, не входящие в первичный ключ, находятся в полной функциональной зависимости от первичного ключа. То есть если эта функциональная зависимость минимальная.

Это требование уже выполняется для всех таблиц, так как у всех первичных ключей нет подмножеств, значит все атрибуты в полной функциональной зависимости от первичных ключей.

# 3НФ

Отношение находится в 3НФ, если оно находится в 2НФ, а также все атрибуты, не входящие в первичный ключ, не находятся в транзитивной функциональной зависимости от первичного ключа.  
Моя модель находится в 3НФ, так как все неключевые атрибуты зависят напрямую от первичных ключей, и не находятся в транзитивных зависимостях. Все атрибуты, которые могли бы находится в такой зависимости оформлены отдельными сущностями

# BCNF

Отношение находится в BCNF, если оно находится в 3НФ и для всех зависимостей X -> Y, Х является потенциальным ключом. Для моей модели BCNF выполняется, так как для всех зависимостей Х является потенциальным ключом

# Денормализация

Название локации в таблице Character

Если часто выполняются запросы по типу “Сколько человек находится в различных парках суммарно?”, то есть смысл добавить избыточный атрибут “location\_name” из Location в таблицу Character. Таким образом, такие запросы не будут требовать join с location.

Количество персонажей, находящихся в одной локации.

Если есть необходимость в том, чтобы часто выполнять запросы по типу “Сколько персонажей находятся в локации R?”, то можно добавить атрибут “quantity” (счётчик) в таблицу Location, который будет поддерживать с помощью триггера количество персонажей, находящихся в одной локации.

Названия стартовой и конечной локаций в таблице Relocation

Можно добавить в таблицу “Relocation” атрибуты “start\_location\_name” и “end\_location\_name”, для того чтобы быстрее выполнять запросы по типу “С помощью каких передвижений можно добраться от парка до пляжа?”.

# Триггер

В денормализации уже был рассмотрен вариант, как оптимизировать работу при частых запросах типа “Сколько персонажей находятся в локации R?” с помощью добавления атрибута “quantity” (счётчик) в таблицу Location. Однако этот счётчик должен поддерживаться для того, чтобы значение было всегда актуальным. Чтобы это не требовало постоянного update со стороны приложения, можно создать триггер, который при смене локации у character, удалении или добавлении нового character будет обновлять счётчик у прошлой и у новой локаций.

CREATE OR REPLACE FUNCTION *upd\_quantity\_location*()  
 returns trigger as $$  
BEGIN  
 IF TG\_OP = 'UPDATE' AND OLD.current\_location\_id IS DISTINCT FROM NEW.current\_location\_id THEN  
 UPDATE Location  
 SET quantity = quantity - 1  
 WHERE location\_id = OLD.current\_location\_id;  
  
 UPDATE Location  
 SET quantity = quantity + 1  
 WHERE location\_id = NEW.current\_location\_id;  
 ELSIF TG\_OP = 'DELETE' THEN  
 UPDATE Location  
 SET quantity = quantity - 1  
 WHERE location\_id = OLD.current\_location\_id;  
 ELSIF TG\_OP = 'INSERT' THEN  
 UPDATE Location  
 SET quantity = quantity + 1  
 WHERE location\_id = NEW.current\_location\_id;  
 END IF;  
 RETURN NEW;  
END;  
$$ LANGUAGE plpgsql;  
  
CREATE TRIGGER upd\_location\_quantity  
 AFTER INSERT OR DELETE OR UPDATE OF "current\_location\_id" ON Character  
 FOR EACH ROW  
EXECUTE FUNCTION *upd\_quantity\_location*();

# Функция и триггер на языке PL/pgSQL

# Вывод

В ходе лабораторной работы я научилась проверять модель на соответствие 1NF, 2NF, 3NF, BCNF, познакомилась с денормализацией и научилась писать триггеры.