Санкт-Петербургский Национальный Исследовательский Университет ИТМО
Факультет Программной Инженерии и Компьютерной Техники

Лабораторная работа №3
По дисциплине
Базы Данных

Выполнил студент группы P3117:
Кудрявцева Руслана Сергеевна

Преподаватель:
Чупанов Аликылыч Алибекович

Санкт-Петербург 2025 г.

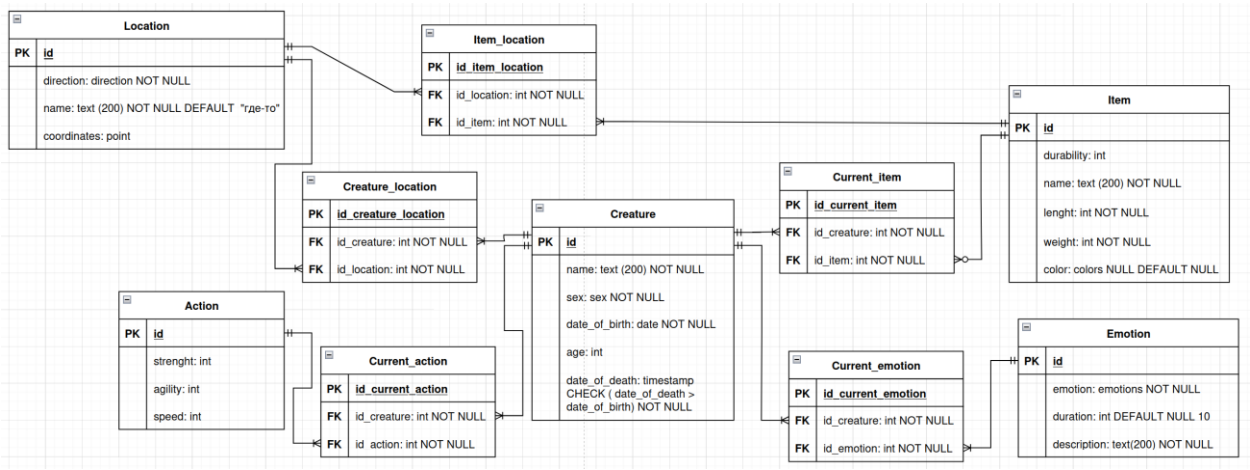
1. Текст задания

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 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 (исходная модель)



Функциональные зависимости (изначальные)

- **location:** id → (name, coordinates, direction)
- **action:** id → (action, strenght, agility, speed, duration)
- **emotion:** id → (emotion, duration, description)
- **item:** id → (name, lenght, weight, durability, color, id_location)
- **creature:** id → (name, age, sex, date_of_birth, date_of_death, id_location)
- **current_item:** id_current_item → (id_creature, id_item)
- **creature_action:** id_creature_action → (id_creature, id_action)
- **creature_emotion:** id_creature_emotion → (id_creature, id_emotion)

2. Преобразование к 1НФ

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

3. Преобразование к 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.
Это требование уже выполняется для всех таблиц, так как у всех первичных ключей нет подмножеств.

4. Преобразование к 3НФ

Отношение находится в 3НФ, если оно находится в 2НФ и не содержит транзитивных зависимостей.
Уже выполняется. Все атрибуты, которые могли бы находится в такой зависимости оформлены отдельными сущностями

5. Преобразование к BCNF

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

6. Денормализация

Объединение таблиц может ускорить обработку запросов. Например, можно рассмотреть объединение таблиц creature_action и action, если часто требуется одновременно знать, какое существо выполняет какое действие и с какими характеристиками (сила, ловкость, скорость и т.д.). В этом случае можно создать отдельную денормализованную таблицу или материализованное представление, которое будет содержать все нужные поля для быстрого доступа без дополнительных соединений.
Также можно добавить несколько избыточных атрибутов, что может улучшить производительность запросов. Например, если часто запрашивается количество предметов, принадлежащих существу, можно добавить в таблицу creature атрибут item_count. Это позволит избежать подсчета при каждом запросе, но потребует обновления этого атрибута при добавлении или удалении записей в таблице current_item.

7. Триггер

Триггер вызывет функцию, которая устанавливает нужные id у мертвых creature. Как только мы поменяем время смерти у creature, то срабатывает триггер, который меняет id_action в таблице creature_action (на значение «3», которое соответсвует стаусу «Мертв»), id_item в таблице current_item (на id со значением «Книга "Как выжить в ИТМО"») и id_location в таблице creature (id = «Граница мира»).

```
CREATE OR REPLACE FUNCTION on_creature_death_update()
RETURNS TRIGGER AS $$
BEGIN
  UPDATE creature_action
  SET id_action = 3
  WHERE id_creature = NEW.id;

  UPDATE current_item
  SET id_item = 4
  WHERE id_creature = NEW.id;
```

```
  UPDATE creature
  SET id_location = 15
  WHERE id = NEW.id;

RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trg_on_creature_death
AFTER UPDATE ON Creature
FOR EACH ROW
WHEN (OLD.date_of_death IS NULL AND NEW.date_of_death IS NOT NULL)
EXECUTE FUNCTION on_creature_death_update();
```

8. Вывод

При выполнении лабораторной работы я узнала, что из себя представляет функциональная зависимость в базах данных, познакомилась с понятием нормализации и денормализации, научилась определять функциональные зависимости модели и анализировать модели на соответствие нормальным формам. Также научилась писать собственные триггеры.