Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО»

Факультет Программной Инженерии и Компьютерной Техники

Лабораторная работа №3

Выполнил:

Шмунк Андрей Александрович

Группа P3108

Преподаватели:

Афанасьев Дмитрий Борисович

Николаев Владимир Вячеславович

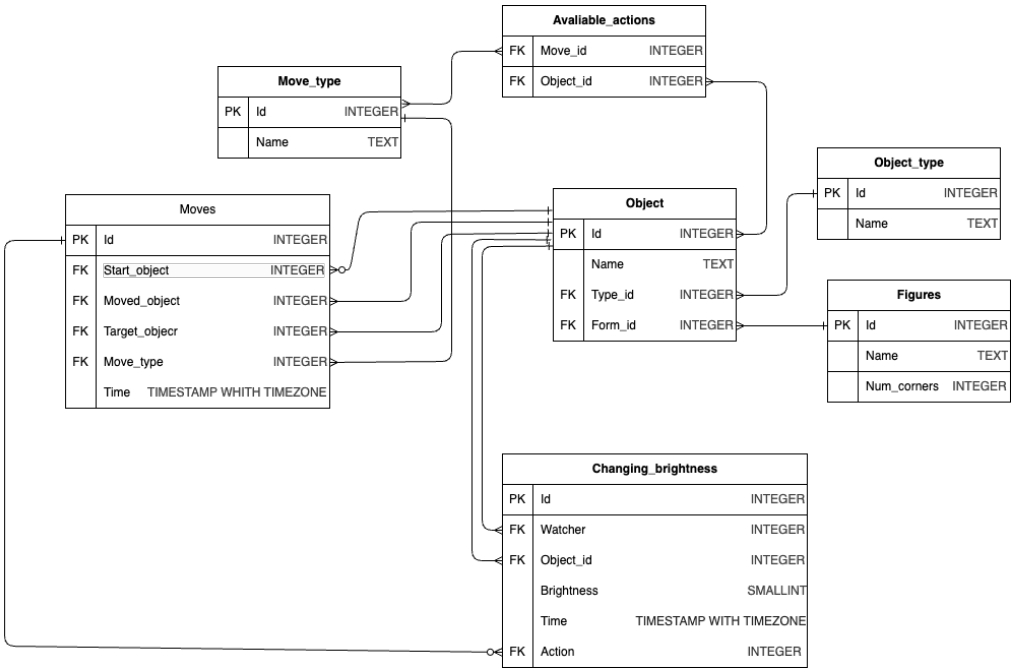
Санкт-Петербург 2024

Содержание

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.



Исходная модель

Функциональные зависимости:

Move_type	Id → (Name)
Object	Id → (Name, Type_id, Form_id)
Figures	Id → (Name, Num_corners)
Changing_brightness	Id → (Watcher, Object_id, Brightness, Time, Action)
Moves	Id → (Start_object, Moved_object, Target_object, Move_type, Time)
Object_type	Id → (Name)
Available_actions	(Move_id, Object_id) → ()

Нормализация

1NF: Отношение находится в 1NF, если все его атрибуты содержат только атомарные значения. Моя модель удовлетворяет 1NF, так как все атрибуты атомарны, и нет повторяющихся групп.

2NF: Отношение находится во 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.

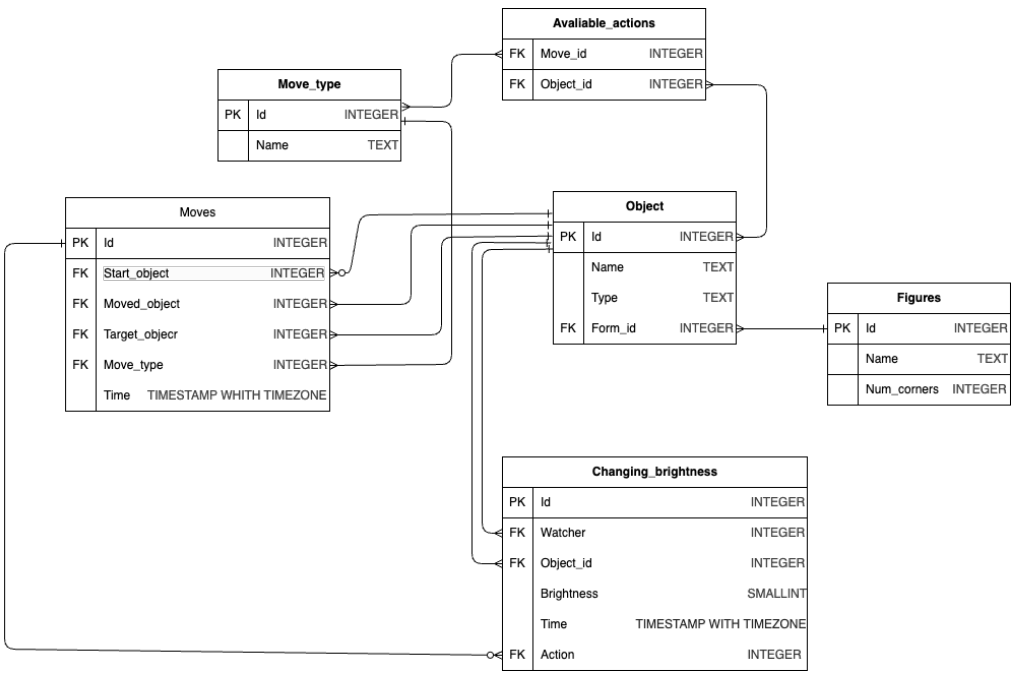
3NF: Отношение находится в 3NF, если оно находится во 2NF и не содержит транзитивных зависимостей. Моя модель удовлетворяет 3NF, так как все неключевые атрибуты зависят только от первичных ключей, и не содержат транзитивных зависимостей.

BCNF: Отношение находится в BCNF, когда для всех функциональных зависимостей отношения выполняется условие: детерминант – потенциальный ключ. Моя модель удовлетворяет BCNF, так как для всех функциональных зависимостей детерминант является потенциальным ключом и достигнута 3NF.

Возможная денормализация

Объединение связанных таблиц: В некоторых случаях, объединение таблиц может уменьшить количество операций JOIN и ускорить обработку запросов. Например, можно рассмотреть объединение таблиц Object и Object_type, если часто запрашиваются данные о том, что из себя представляет объект.

Добавление избыточных атрибутов: В некоторых случаях добавление избыточных атрибутов может улучшить производительность запросов, но я считаю, что добавление каких-то атрибутов здесь не нужно, поскольку вся информация уже присутствует в базе данных.



Денормализованная модель

Функция и триггер на языке PL/pgSQL

Функция, которая запрещает изменение типа объекта:

```
CREATE OR REPLACE FUNCTION typeChecker() RETURNS TRIGGER AS
$$
BEGIN
  IF OLD.type_id != NEW.type_id THEN
    RAISE EXCEPTION 'Вы не имеете права изменять тип объекта!';
  END IF;
END ;
$$ LANGUAGE plpgsql;

CREATE TRIGGER typeTrigger
  BEFORE UPDATE
  ON object
  FOR EACH ROW
EXECUTE PROCEDURE typeChecker();
```

Функция, которая запрещает смену смотрящего:

```
CREATE OR REPLACE FUNCTION watcherChecker() RETURNS  TRIGGER AS
$$
BEGIN
  IF OLD.watcher !=  NEW.watcher THEN
    RAISE EXCEPTION 'Вы не имеете права изменять смотрящего!';
  END IF;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER watcherTrigger
  BEFORE UPDATE
  ON changing_brightness
  FOR EACH ROW
EXECUTE watcherChecker();
```

Функция, которая запрещает запись изменения яркости, произошедшие ранее уже имеющихся:

```
CREATE OR REPLACE FUNCTION timeChecker() RETURNS TRIGGER AS
$$
DECLARE
  max_time TIMESTAMP WITH TIME ZONE := NULL;
BEGIN
  SELECT MAX(time) INTO max_time FROM changing_brightness;
  IF max_time > NEW.time THEN
    RAISE EXCEPTION 'Вы не имеете права добавлять запись задним числом!';
  END IF;
  RETURN NEW;
```

```
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER timeTrigger
   BEFORE INSERT
   ON changing_brightness
   FOR EACH ROW
EXECUTE PROCEDURE timeChecker();
```

Вывод

В ходе выполнения данной лабораторной работы я познакомился с различными нормальными формами и процессом приведения к ним, а также с процессом денормализации модели и написанием параметризированных функций на языке plpgsql.