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 по

Базам данных №3

Вариант 3

Работу выполнила:

Касьяненко В. М.

Группа:

P3120

Преподаватель:

Королева Ю. А.

Санкт-Петербург,

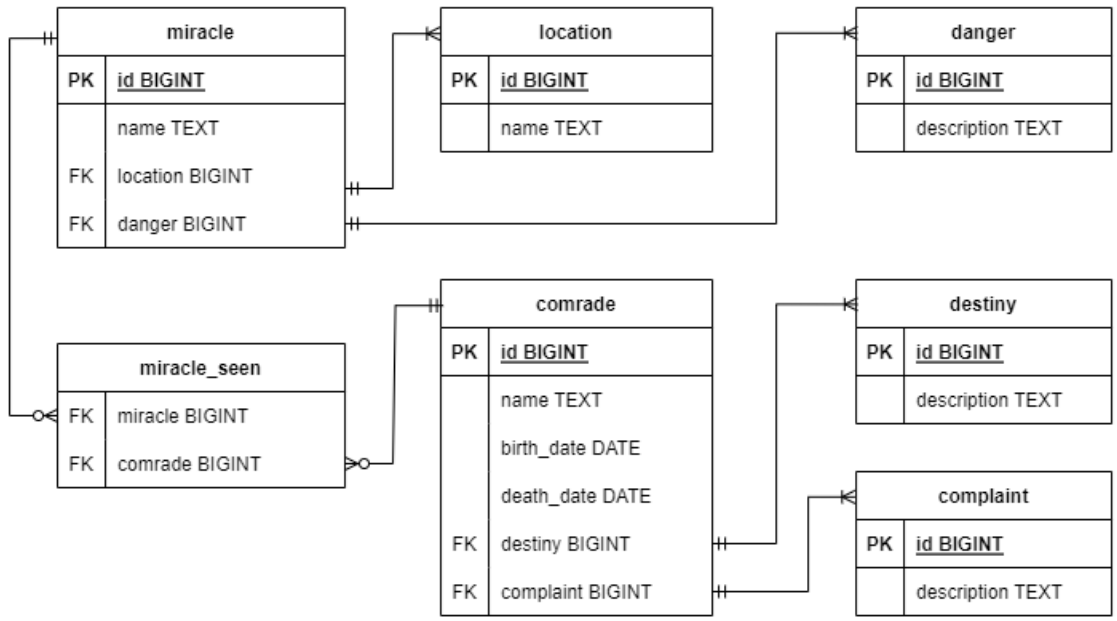
2023

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 NF (как минимум). Постройте схему на основе полученных отношений;
- опишите изменения в функциональных зависимостях, произошедшие после преобразования в 3NF (как минимум). Постройте схему на основе NF;
- преобразуйте отношения в BCNF. Докажите, что полученные отношения представлены в BCNF;
- какие денормализации будут полезны для вашей схемы? Приведите подробное описание.

Даталогическая модель (исходная)



Функциональные зависимости (изначальные)

miracle:

- id -> name
- id -> location
- id -> danger
- name -> id
- name -> location
- name -> danger

danger:

- id -> description
- description -> id

location:

- id -> name
- name -> id

comrade:

- id -> name
- id -> birth_date
- id -> death_date
- id -> destiny
- id -> complaint

complaint:

- id -> description
- description -> id

destiny:

- id -> description
- description -> id

Преобразование к 1НФ

Не потребовалось, условие “на пересечении каждой строки и столбца – 1 значение” и так выполнялось.

Преобразование к 2НФ

Не потребовалось, поскольку у всех первичных ключей нет подмножеств, а значит атрибуты всех отношений – в полной функциональной зависимости от соответствующих первичных ключей.

Преобразование к 3НФ

В некоторых отношениях наблюдались транзитивные зависимости:

- miracle: id -> name, name -> location, name -> danger;

По сути, транзитивность здесь даёт то, что в этих отношениях как бы два первичных ключа, только один указан явно (id), а другой получается в силу того, что значения атрибута должно быть уникальными (name). Поэтому:

- в miracle убираем id и делаем первичным ключом name.

Преобразование к BCNF

Не потребовалось, поскольку “ключевые” атрибуты в наших отношениях не зависят от “неключевых”. Например, по дате рождения (date_birth) мы не можем однозначно определить имя товарища (name), по опасности чуда (danger) – его имя и местоположение (name, location) и т.д.

Функциональные зависимости (после преобразований)

miracle:

- name -> location
- name -> danger

danger:

- id -> description
- description -> id

location:

- id -> name
- name -> id

comrade:

- id -> name
- id -> birth_date
- id -> death_date
- id -> destiny
- id -> complaint

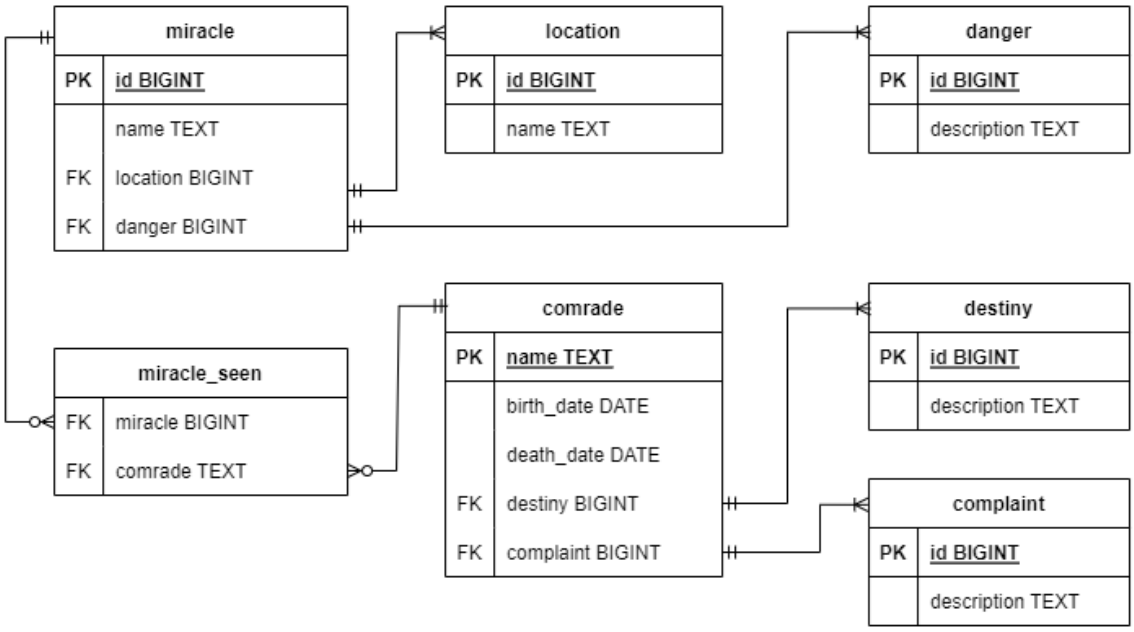
complaint:

- id -> description
- description -> id

destiny:

- id -> description
- description -> id

Даталогическая модель (после преобразований)

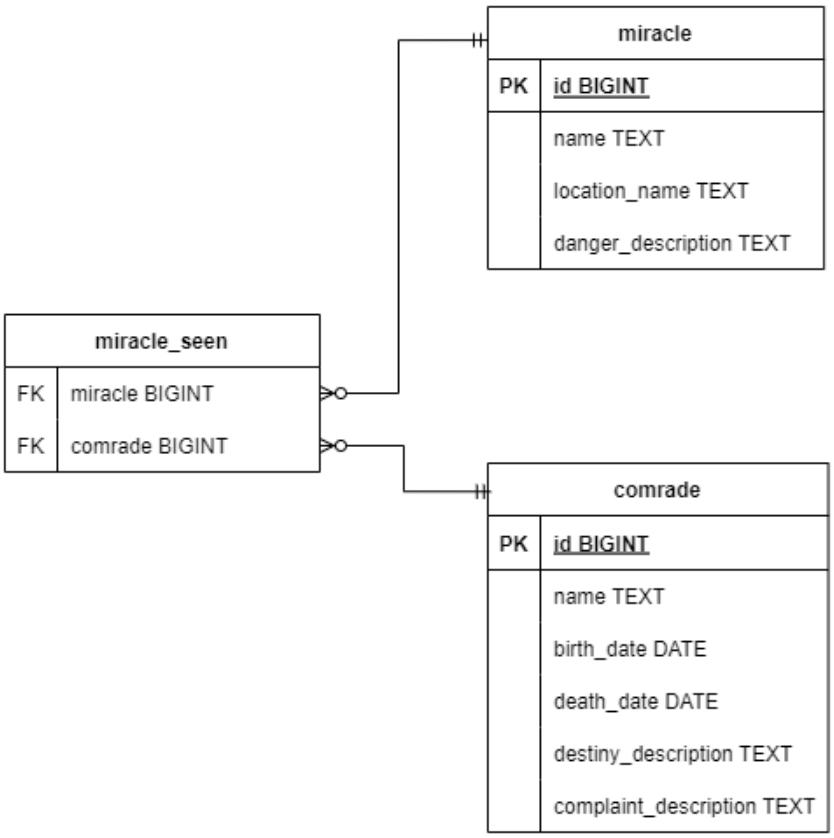


Полезная денормализация

Теоретически, можно увеличить скорость выполнения некоторых запросов к нашей бд за счёт сокращения количества сущностей, а именно – уничтожения всех характеристических сущностей и добавления их атрибутов (кроме id, естественно) в стержневые сущности. Минусы такого подхода:

- Увеличится избыточность данных;
- Сложнее контролировать константы (можно сделать enum, но тогда придётся новые значения вписывать в enum каждый раз).

Даталогическая модель (после денормализации)

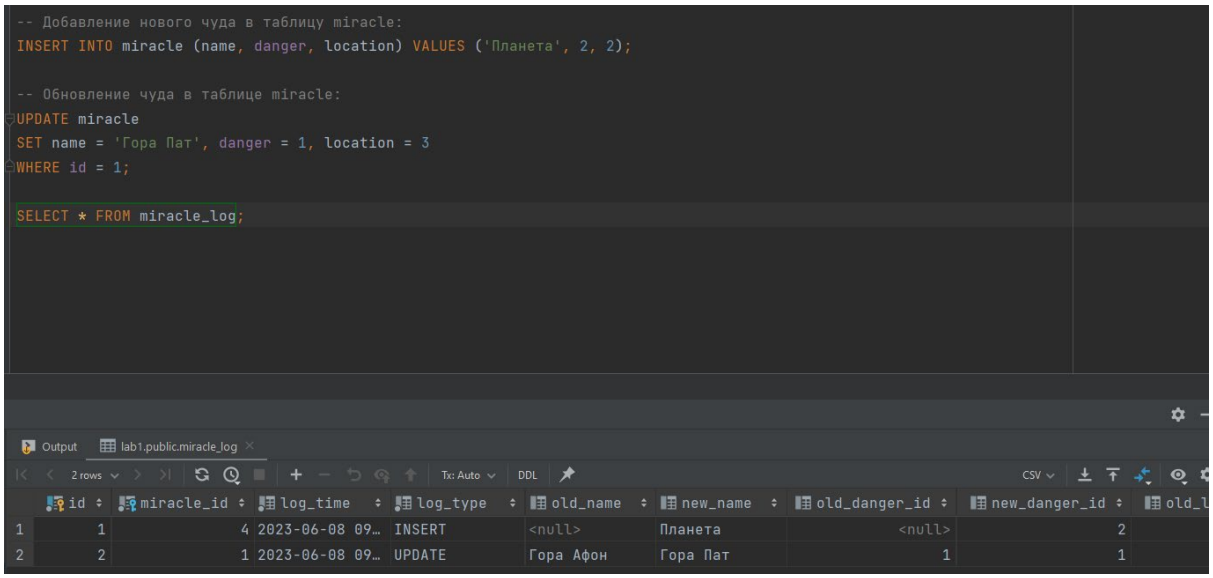


Дополнительное задание

Сделать логирование для таблицы, которая часто обновляется (miracle).

```sql
CREATE TABLE miracle_log (
    id BIGSERIAL PRIMARY KEY,
    miracle_id BIGINT REFERENCES miracle(id) NOT NULL,
    log_time TIMESTAMP NOT NULL,
    log_type TEXT NOT NULL,
    old_name TEXT,
    new_name TEXT,
    old_danger_id BIGINT,
    new_danger_id BIGINT,
    old_location_id BIGINT,
    new_location_id BIGINT
);
```

```sql
CREATE OR REPLACE FUNCTION miracle_log_insert_trigger()
RETURNS TRIGGER AS $$
BEGIN
    INSERT INTO miracle_log (miracle_id, log_time, log_type, new_name, new_danger_id, new_location_id)
    VALUES (NEW.id, now(), 'INSERT', NEW.name, NEW.danger, NEW.location);
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER miracle_insert_trigger
AFTER INSERT ON miracle
FOR EACH ROW
EXECUTE FUNCTION miracle_log_insert_trigger();

CREATE OR REPLACE FUNCTION miracle_log_update_trigger()
RETURNS TRIGGER AS $$
BEGIN
    INSERT INTO miracle_log (miracle_id, log_time, log_type, old_name, new_name, old_danger_id, new_danger_id, old_location_id, new_location_id)
    VALUES (NEW.id, now(), 'UPDATE', OLD.name, NEW.name, OLD.danger, NEW.danger, OLD.location, NEW.location);
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER miracle_update_trigger
AFTER UPDATE ON miracle
FOR EACH ROW
EXECUTE FUNCTION miracle_log_update_trigger();

CREATE OR REPLACE FUNCTION miracle_log_delete_trigger()
RETURNS TRIGGER AS $$
BEGIN
    INSERT INTO miracle_log (miracle_id, log_time, log_type, old_name, old_danger_id, old_location_id)
    VALUES (OLD.id, now(), 'DELETE', OLD.name, OLD.danger, OLD.location);
    RETURN OLD;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER miracle_delete_trigger
AFTER DELETE ON miracle
FOR EACH ROW
EXECUTE FUNCTION miracle_log_delete_trigger();
```



Вывод

При выполнении данной лабораторной работы я узнала, что из себя представляет функциональная зависимость в базах данных, познакомилась с сопутствующей терминологией. Также, я выяснила, как находить функциональные зависимости в отношениях и как приводить отношения к 1NF, 2NF, 3NF и BCNF.