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

Факультет Программной инженерии и компьютерной техники

Отчёт по лабораторной работе №3

Информационные системы и базы данных

Вариант № 3910

Выполнил: студент группы P3131

Серов А. А.

Проверил: Вербовой А.А.

Санкт-Петербург 2025г.

1. **Текст задания**

Для отношений, полученных при построении предметной области из лабораторной работы №1, выполните следующие действия:

1. Опишите функциональные зависимости для отношений полученной схемы (минимальное множество);
2. Приведите отношения в 3NF (как минимум). Постройте схему на основеNF (как минимум).
3. Опишите изменения в функциональных зависимостях, произошедшие после преобразования в 3NF (как минимум). Постройте схему на основеNF;
4. Преобразуйте отношения в BCNF. Докажите, что полученные отношения представлены в BCNF. Если ваша схема находится уже в BCNF, докажите это;
5. 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.

**Изображение выглядит как текст, диаграмма, План, Параллельный

Контент, сгенерированный ИИ, может содержать ошибки.**

1. **Функциональные зависимости**

PERSON:

PERSON\_ID → (NAME, HEIGHT, AGE, GENDER)

BUILDING:

BUILDING\_ID → (NAME, HEIGHT, MATERIAL)

PANEL:

PANEL\_ID → (TYPE, OPACITY, TEMPERATURE, BUILDING\_ID)

SHARP:

SHARP\_ID → (COUNT)

LIGHT\_DOT:

LIGHT\_DOT\_ID → (BRIGHTNESS, COLOR, POS\_X, POS\_Y, POS\_Z, BUILDING\_ID, SHARP\_ID)

VIEW:

(PERSON\_ID, BUILDING\_ID) → ()

1. **Приведение отношений в 3NF и построение схемы**

Все отношения уже находятся в 3NF, так как:

1. Все атрибуты атомарны (1NF)
2. Нет частичных зависимостей от первичного ключа (2NF)
3. Нет транзитивных зависимостей (3NF)

Схема на основе 3NF:

Схема остается без изменений, так как она уже соответствует требованиям 3NF.

1. **Преобразование отношений в BCNF**

Анализ на соответствие BCNF:

Отношение находится в BCNF, если для каждой функциональной зависимости X → Y, X является суперключом. В нашей схеме:

* Для всех сущностей (PERSON, BUILDING, PANEL, SHARP, LIGHT\_DOT) первичный ключ (PERSON\_ID, BUILDING\_ID и т.д.) определяет все атрибуты, и нет других функциональных зависимостей.
* Для ассоциативной сущности VIEW составной первичный ключ (PERSON\_ID, BUILDING\_ID) не имеет других атрибутов.

Таким образом, все отношения уже находятся в BCNF.

1. **Полезные денормализации для схемы**

**Объединение таблиц LIGHT\_DOT и SHARP:**

Если часто запрашиваются данные о световых точках вместе с информацией о их форме (SHARP), можно объединить эти таблицы. Плюсы: уменьшение количества JOIN операций при запросах. Минусы: дублирование данных COUNT для точек с одинаковым SHARP\_ID.

CREATE TABLE LIGHT\_DOT\_DENORM AS

SELECT LD.\*, S.COUNT

FROM LIGHT\_DOT LD JOIN SHARP S ON LD.SHARP\_ID = S.SHARP\_ID;

**Добавление количества панелей в таблицу BUILDING:**

Можно добавить атрибут PANEL\_COUNT в таблицу BUILDING. Плюсы: быстрый доступ к количеству панелей без подсчета. Минусы: необходимо обновлять значение при добавлении/удалении панелей.

ALTER TABLE BUILDING ADD COLUMN PANEL\_COUNT INT DEFAULT 0;

1. **Функция на языке PL/pgSQL**

**Функция для проверки допустимого значения температуры у панели**

-- Проверка допустимого значения температуры у панели

CREATE OR REPLACE FUNCTION check\_panel\_temperature()  
RETURNS TRIGGER AS $$  
BEGIN  
 -- Запрет на температуру выше 100 или ниже -20  
 IF NEW.TEMPERATURE > 100 THEN  
 RAISE EXCEPTION 'Ошибка: Температура панели %.1f°C превышает 100°C', NEW.TEMPERATURE;  
 ELSIF NEW.TEMPERATURE < -20 THEN  
 RAISE EXCEPTION 'Ошибка: Температура панели %.1f°C ниже -20°C', NEW.TEMPERATURE;  
 END IF;  
  
 -- Предупреждение для непрозрачных панелей (OPACITY = 1) с температурой меньше 10  
 IF NEW.OPACITY = 1 AND NEW.TEMPERATURE < 10 THEN  
 RAISE WARNING 'Внимание: Непрозрачная панель (ID=%) имеет низкую температуру %.1f°C',   
 NEW.PANEL\_ID, NEW.TEMPERATURE;  
 END IF;  
  
 RETURN NEW;   
END;  
$$ LANGUAGE plpgsql;

**Триггер: автоматическая проверка при изменении данных**

--Триггер на автоматическую проверку при изменении данных

CREATE TRIGGER panel\_temperature\_control

BEFORE INSERT OR UPDATE ON PANEL

FOR EACH ROW

EXECUTE FUNCTION check\_panel\_temperature();

**Тесты для проверки скрипта**

-- Попытка вставить панель с температурой 150°C (должна вызвать ошибку)  
INSERT INTO PANEL (TYPE, OPACITY, TEMPERATURE, BUILDING\_ID)  
VALUES ('test', 0.5, 150, 1);  
  
-- Попытка вставить панель с температурой -25°C (должна вызвать ошибку)  
INSERT INTO PANEL (TYPE, OPACITY, TEMPERATURE, BUILDING\_ID)  
VALUES ('test', 0.5, -25, 1);  
  
-- Корректная панель (должна добавиться)  
INSERT INTO PANEL (TYPE, OPACITY, TEMPERATURE, BUILDING\_ID)  
VALUES ('valid', 0.5, 25, 1);  
  
-- Непрозрачная панель с низкой температурой (должна вызвать предупреждение)  
INSERT INTO PANEL (TYPE, OPACITY, TEMPERATURE, BUILDING\_ID)  
VALUES ('opaque', 1, 5, 1);

1. **Вывод**

В ходе выполнения лабораторной работы я проанализировал текущую схему базы данных на соответствие нормальным формам. Установил, что схема уже находится в BCNF, так как все функциональные зависимости определяются суперключами. Рассмотрел возможные варианты денормализации для улучшения производительности запросов. Разработал триггер для проверки температуры панелей и функцию, что демонстрирует практическое применение PL/pgSQL для реализации правил предметной области.