Санкт-Петербургский Национальный Исследовательский Университет ИТМО

Факультет Программной Инженерии и Компьютерной Техники



Вариант №492541

Лабораторная работа №3

По дисциплине

Базы Данных

Выполнил студент группы P3117:

Изаак Герман Константинович

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.

1. **Функциональные зависимости**

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

Контент, сгенерированный ИИ, может содержать ошибки.**

**Location:** id\_location → (name, coordinates, is\_active)

**Trajectory:** id\_trajectory → (start\_time, end\_time, arrival\_location\_id, place\_of\_departure\_id)

**Groups:** id\_group → (name, association, quantity, trajectory\_id)

**Fish:** id\_fish → (weight, size, location\_id, group\_id)

**Interaction:** id\_interaction → (time, description, fish\_1\_id, fish\_2\_id)

**State:** id\_state → (brightness, speed, state\_time, temperature, fish\_id)

**Cycle:** id\_cycle → (type, frequency, duration)

**Cycle\_Group:** (id\_cycle, id\_group) → ()

**Cycle\_Fish:** (id\_fish, id\_cycle) → ()

1. **Нормальные формы**

**1NF:** Убедимся, что модель находится в 1NF. Это действительно так, потому что все поля являются атомарными и нет повторяющихся наборов столбцов.

**2NF:** Модель удовлетворяет 1NF и атрибуты, не входящие в первичный ключ, находятся в полной функциональной зависимости от первичного ключа. Единственные отношения с составными ключами — cycle\_group(id\_cycle, id\_group) и cycle\_fish(id\_fish, id\_cycle). В них нет других столбцов, поэтому частичных зависимостей не существует. Во всех остальных таблицах PK одинарный → частичные зависимости невозможны. Значит, модель находится в 2NF.

**3NF:** Модель удовлетворяет 3NF, так как неключевые столбцы никак не определяют друг друга внутри одной таблицы. Например: в groups атрибуты name, association, quantity, trajectory\_id независимы друг от друга и зависят только от id\_group. Аналогично для fish, state, trajectory, cycle и др. Следовательно транзитивных зависимостей нет.

1. **BCNF**

Все существующие функциональные зависимости детерминированы ключами, значит требования BCNF выполняются без дополнительной декомпозиции.

1. **Денормализация**

* **Координаты локации прямо в таблице fish**

**Где?** Добавляем столбец coord POINT в fish.  
**Как?** ALTER TABLE fish ADD COLUMN coord POINT; + триггер, который при вставке/изменении location\_id копирует location.coordinates в fish.coord.  
**Зачем?** Запросы «покажи всех рыб в радиусе R» больше не требуют JOIN с location.

* **Названия пунктов прямо в trajectory**

**Где?** Добавляем столбцы departure\_name VARCHAR(255) и arrival\_name VARCHAR(255) в trajectory.  
**Как?** ALTER TABLE trajectory ADD COLUMN … + триггер BEFORE INSERT/UPDATE, который по place\_of\_departure\_id и arrival\_location\_id подтягивает location.name и заносит копии.  
**Зачем?** Список маршрутов приходит без двух JOIN location, запросы становятся короче и работают быстрее.

* **Численность группы в самой таблице groups**

**Где?** Поле quantity уже есть, нужно поддерживать его актуальным.  
**Как?** Триггер AFTER INSERT/DELETE ON fish увеличивает/уменьшает groups.quantity.  
**Зачем?** Отчёт «топ‑10 самых больших групп» выполняется мгновенно — без подсчёта каждую секунду.

1. **Триггер**

Таблица **groups** уже содержит поле **quantity** (число рыб). Чтобы это значение всегда оставалось актуальным и не требовало отдельного UPDATE со стороны приложения, создадим триггер на таблицу **fish**, который автоматически корректирует groups.quantity при INSERT, DELETE и смене group\_id у рыбы.

CREATE OR REPLACE FUNCTION *trg\_update\_group\_quantity*()  
 RETURNS TRIGGER AS  
$$  
BEGIN  
 -- Added a new fish  
 IF TG\_OP = 'INSERT' THEN  
 UPDATE groups  
 SET quantity = quantity + 1  
 WHERE id\_group = NEW.group\_id;  
  
 -- The fish was removed  
 ELSIF TG\_OP = 'DELETE' THEN  
 UPDATE groups  
 SET quantity = quantity - 1  
 WHERE id\_group = OLD.group\_id;  
  
 -- Changed the fish's group  
 ELSIF TG\_OP = 'UPDATE' THEN  
 IF NEW.group\_id <> OLD.group\_id THEN  
 UPDATE groups  
 SET quantity = quantity - 1  
 WHERE id\_group = OLD.group\_id;  
  
 UPDATE groups  
 SET quantity = quantity + 1  
 WHERE id\_group = NEW.group\_id;  
 END IF;  
 END IF;  
  
 RETURN NULL;  
END;  
$$ LANGUAGE plpgsql;  
  
  
DROP TRIGGER IF EXISTS trg\_fish\_group\_quantity ON fish;  
  
CREATE TRIGGER trg\_fish\_group\_quantity  
 AFTER INSERT OR DELETE OR UPDATE OF group\_id  
 ON fish  
 FOR EACH ROW  
EXECUTE FUNCTION *trg\_update\_group\_quantity*();

1. **Вывод по работе**

В ходе выполнения работы было установлено, что разработанная модель соответствует 3НФ и НФ Бойса–Кодда. Предложены денормализации для оптимизации частых запросов, включая кэширование координат и численности групп. Также реализован триггер на языке PL/pgSQL, автоматически поддерживающий актуальное количество рыб в каждой группе.