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 №3**

Вариант № 5744

Группа: P3112

Выполнил: Балин А. А.

Проверила: Лисицина В.В.

Оглавление

[Введение 3](#_Toc136445764)

[Текст задания 4](#_Toc136445765)

[Выполнение 5](#_Toc136445766)

[Триггер 7](#_Toc136445767)

[Заключение 8](#_Toc136445768)

[Список литературы 9](#_Toc136445769)

# Введение

В данной лабораторной работе я изучу виды нормализации, применю нормализацию к лабораторной работе №1 и напишу триггер для проверки сложных условий целостности таблиц.

# 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).

Опишите изменения в функциональных зависимостях, произошедшие после преобразования в 3NF (как минимум). Постройте схему на основеNF;

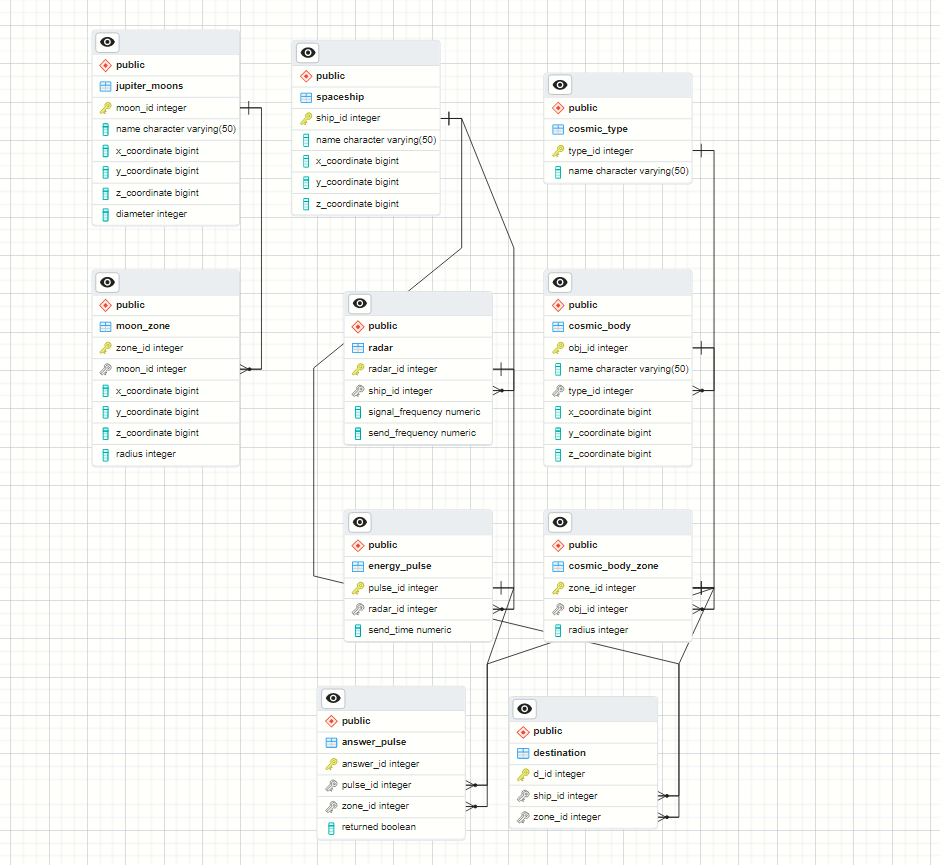
Преобразуйте отношения в BCNF. Докажите, что полученные отношения представлены в BCNF. Если ваша схема находится уже в BCNF, докажите это;

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.

# Выполнение

Моя модель:



Функциональные зависимости:

spaceship:

ship\_id -> name, x\_coordinate, y\_coordinate, z\_coordinate

radar:

radar\_id -> ship\_id, signal\_frequency, send\_frequency

energy\_pulse:

pulse\_id -> radar\_id, send\_time

cosmic\_type:

type\_id -> name

cosmic\_body:

obj\_id -> name, type\_id, x\_coordinate, y\_coordinate, z\_coordinate

moon\_zone:

zone\_id -> obj\_id, radius

destination:

d\_id -> ship\_id, zone\_id

answer\_pulse:

answer\_id -> pulse\_id, zone\_id, returned

Нормализация

* 1НФ: в отношениях нет групп из более чем 1 элемента, модель уже в 1НФ.
* 2НФ: выполняется полная функциональная зависимость, нет частичных функциональных зависимостей, модель в 2НФ.
* 3НФ: модель в 1НФ и 2НФ, в отношениях отсутствуют транзитивные функциональные зависимости, модель в 3НФ.
* НФБК: все детерминанты являются потенциальными первичными ключами.

Полезная денормализация

* Соединить таблицы destination и spaceship, то есть у каждого корабля при близком расположении к объекту будет указана зона, в которой он находится, что позволит избежать лишнего JOIN-запроса.
* Оставить связь «многие ко многим» в таблице answer\_pulse, но перенести атрибут returned, чтобы можно было выбрать только вернувшиеся сигналы.

# Триггер

При добавлении/изменении/удалении элементов spaceship при нахождении корабля в одной из зон объектов (сфера вокруг cosmic\_object), соответствующие данные будут удалены или добавлены в таблицу destination.

CREATE OR REPLACE FUNCTION check\_coordinates()

RETURNS TRIGGER AS $$

BEGIN

    IF TG\_OP = 'UPDATE' OR TG\_OP = 'DELETE' THEN

        DELETE FROM destination

        WHERE ship\_id = OLD.ship\_id;

    END IF;

    IF TG\_OP = 'UPDATE' OR TG\_OP = 'INSERT' THEN

        INSERT INTO destination (ship\_id, zone\_id)

        SELECT NEW.ship\_id, cz.zone\_id

        FROM cosmic\_body\_zone cz

        INNER JOIN cosmic\_body cb ON cz.obj\_id = cb.obj\_id

        WHERE ABS(cb.x\_coordinate - NEW.x\_coordinate) < cz.radius

          AND ABS(cb.y\_coordinate - NEW.y\_coordinate) < cz.radius

          AND ABS(cb.z\_coordinate - NEW.z\_coordinate) < cz.radius;

    END IF;

    RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER check\_coordinates\_trigger

AFTER INSERT OR UPDATE OF x\_coordinate, y\_coordinate, z\_coordinate ON spaceship

FOR EACH ROW

EXECUTE FUNCTION check\_coordinates();

# Заключение

Я разобрался в нормализации и денормализации, а также научился писать триггеры.

# Список литературы

**PostgreSQL 9.6.24 Documentation** [В Интернете]. - https://www.postgresql.org/docs/9.6/.