УНИВЕРСИТЕТ ИТМО

Факультет программной инженерии и компьютерной техники

Направление подготовки 09.03.04 Программная инженерия

Дисциплина «Информационный системы и базы данных»

**Лабораторная работа №3**

*Вариант 1416*

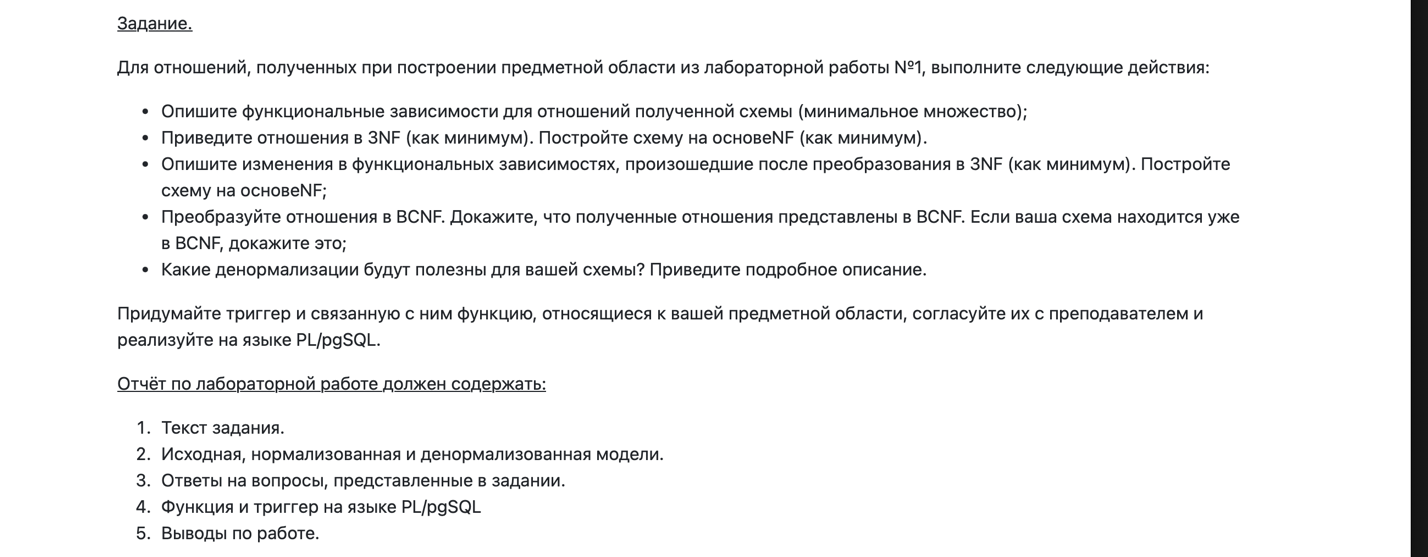
Студент

Армут Алина

*P3114*

Преподаватель

Санкт-Петербург, 2023 г.



Описание предметной области:

Вот он плывет в исполинском потоке солнц, по одну сторону густо пламенеющая россыпь - сердце Галактики, по другую - одинокие, разбросанные по ее обочине звезды-часовые. Здесь бы и остаться, на краю этой небесной расселины, змеящейся беззвездной полосы мрака. Он знал, этот бесформенный хаос, различимый лишь при отсвете, отделяющем его от простершихся в бесконечную даль огненных туманов, - это первооснова творения, сырье для грядущих превращений, оно еще ждет своего часа. Здесь Время еще и не начиналось; лишь долго спустя после того, как сгинут пылающие ныне солнца, возникнут в этой пустынной бездне свет и жизнь и преобразят ее.

Список сущностей

Стержневые:

* Человек – *имя, возраст*
* Космический объект – *название, тип*
* Космический корабль – название, дата создания

Характеристические:

* *местоположение – текущее местоположение, сторона*
* *Сторона – правая или левая и описание стороны*

Ассоциативные:

Инфологическая модель

Изображение выглядит как текст, в помещении, снимок экрана

Автоматически созданное описание

Даталогическая модель

Изображение выглядит как стол

Автоматически созданное описание

Функциональные зависимости для отношений схемы (минимальное множество):

(spaceship) id -> name

(spaceship) id -> cosmos\_object\_id

(cosmos\_object) id -> name

(cosmos\_object) id -> time\_of\_existence

human(id) -> name

human(id) -> spaceship\_id

human(id) -> cosmos\_object\_id

side(id) -> description

location ( spaceship\_id) -> right\_side\_id

location ( spaceship\_id) -> left\_side\_id

location ( spaceship\_id) -> current\_object\_location

Нормализация:

1 Нормальная форма (1НФ)

* В таблице нет дублирующих строк
* Отсутствуют списки и массивы
* В столице хранятся данные одного типа
* В каждой ячейке хранится атомарное значение(= одно не составное значение)

Это ключевое определение реляционной базы данных и при выполнении первой лабораторной работы это было учтено.

**Изменения не требуются**

1. *Нормальная форма (2НФ):*

* Таблица должна находиться в 1NF
* Таблица должна иметь ключ
* Все неключевые столбцы таблицы должны зависеть от полного ключа(в случае, если ключ составной)

Чтобы привести к 2НФ необходимо убрать частичные зависимости от первичного ключа (т.е. в случае наличия составного первичного ключа, все атрибуты полностью зависят от ключа, а не только от его части).

В каждой таблице атрибуты не включенные в первичный ключ непосредственно зависят от него. Следовательно схема удовлетворяет условиям 2НФ.

(spaceship) id -> name

(spaceship) id -> cosmos\_object\_id

(cosmos\_object) id -> name

(cosmos\_object) id -> time\_of\_existence

human(id) -> name

human(id) -> spaceship\_id

human(id) -> cosmos\_object\_id

side(id) -> description

location ( spaceship\_id) -> right\_side\_id

location ( spaceship\_id) -> left\_side\_id

location ( spaceship\_id) -> current\_object\_location

**Изменения не требуются**

*3 Нормальная форма (3НФ):*

* Таблица должна находиться в 2NF
* В таблице отсутствует транзитивная зависимость (когда не ключевые столбцы зависят от значений других не ключевых столбцов)

(spaceship) id -> name

(spaceship) id -> cosmos\_object\_id

Атрибуты name, cosmos\_object\_id не зависят друг от друга, следовательно нет случая транзитивной зависимости, все атрибуты напрямую зависят от первичного ключа и только от него.

(cosmos\_object) id -> name

(cosmos\_object) id -> time\_of\_existence

Атрибуты name, time\_of\_existence не зависят друг от друга, следовательно нет случая транзитивной зависимости, все атрибуты напрямую зависят от первичного ключа и только от него.

human(id) -> name

human(id) -> spaceship\_id

human(id) -> cosmos\_object\_id

Атрибуты name, spaceship\_id, cosmos\_object\_id не зависят друг от друга, следовательно нет случая транзитивной зависимости, все атрибуты напрямую зависят от первичного ключа и только от него.

location ( spaceship\_id) -> right\_side\_id

location ( spaceship\_id) -> left\_side\_id

location ( spaceship\_id) -> current\_object\_location

Атрибуты right\_side\_id, left\_side\_id, сurrent\_object\_locationне зависят друг от друга, следовательно нет случая транзитивной зависимости, все атрибуты напрямую зависят от первичного ключа и только от него.

Можем сделать вывод, что нет ни одного отношения с транзитивными зависимостями, значит база данных находится в 3НФ.

**Изменения не требуются**

BCNF (Нормальная форма Бойса-Кодда)

* Таблица должна находиться в 3NF
* Ключевые атрибуты составного ключа не должны зависеть от не ключевых атрибутов.

Требования предъявляются к таблицам, у которых первичный ключ составной

Таблицы, у которых первичный ключ простой и они находятся в 3NF, автоматически относятся к BCNF.

**Изменения не требуются**

*Полезные денормализации:*

**Объединение связанных таблиц**: в некоторых случаях, объединение таблиц может уменьшить количество операций JOIN и ускорить обработку запросов.

В представленной базе данных можно использовать денормализацию для улучшения производительности и упрощения запросов. Одним из способов денормализации, может быть, добавление некоторых атрибутов из связанных таблиц в основную таблицу.

Добавление избыточных атрибутов: "name" и "time\_of\_existence" из таблицы "cosmos\_object" в таблицу "Human":

В этом случае вы сможете получить информацию о космическом объекте, связанном с человеком, без необходимости выполнения дополнительного запроса к таблице "cosmos\_object".

*Триггер:*

Для использования триггерной функции я добавила атрибут под названием spaceship\_location, в котором содержится spaceship\_id и current\_location, а также добавила в атрибут spaceship колонку под названием current\_location. Когда в таблице spaceship происходит обновление столбца current\_location срабатывает триггерная функция и в таблице spaceship\_location обновляется значение столбца current\_location.

CREATE OR REPLACE FUNCTION update\_spaceship\_location()

RETURNS TRIGGER AS $$

BEGIN

IF TG\_OP = 'UPDATE' THEN

UPDATE spaceship\_location

SET current\_location = NEW.current\_location

WHERE spaceship\_id = NEW.id;

END IF;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE OR REPLACE TRIGGER update\_spaceship\_location\_trigger

AFTER UPDATE ON spaceship

FOR EACH ROW

EXECUTE FUNCTION update\_spaceship\_location();

Выводы

При выполнении лабораторной работы я познакомилась с нормальными формами для нормализации базы данных, написала функцию, которая может быть полезной для моей базы данных.