Санкт-Петербургский Национальный Исследовательский Университет ИТМО

Факультет Программной Инженерии и Компьютерной Техники



Вариант **№1503**

Лабораторная работа **№3**

По дисциплине

**Базы Данных**

*Выполнил*:

Студент группы P3115

Барсуков Максим

Андреевич

*Преподаватель*:

Горбунов Михаил

Витальевич

Николаев Владимир

Вячеславович

Санкт-Петербург 2023 г.

1. **Текст задания**

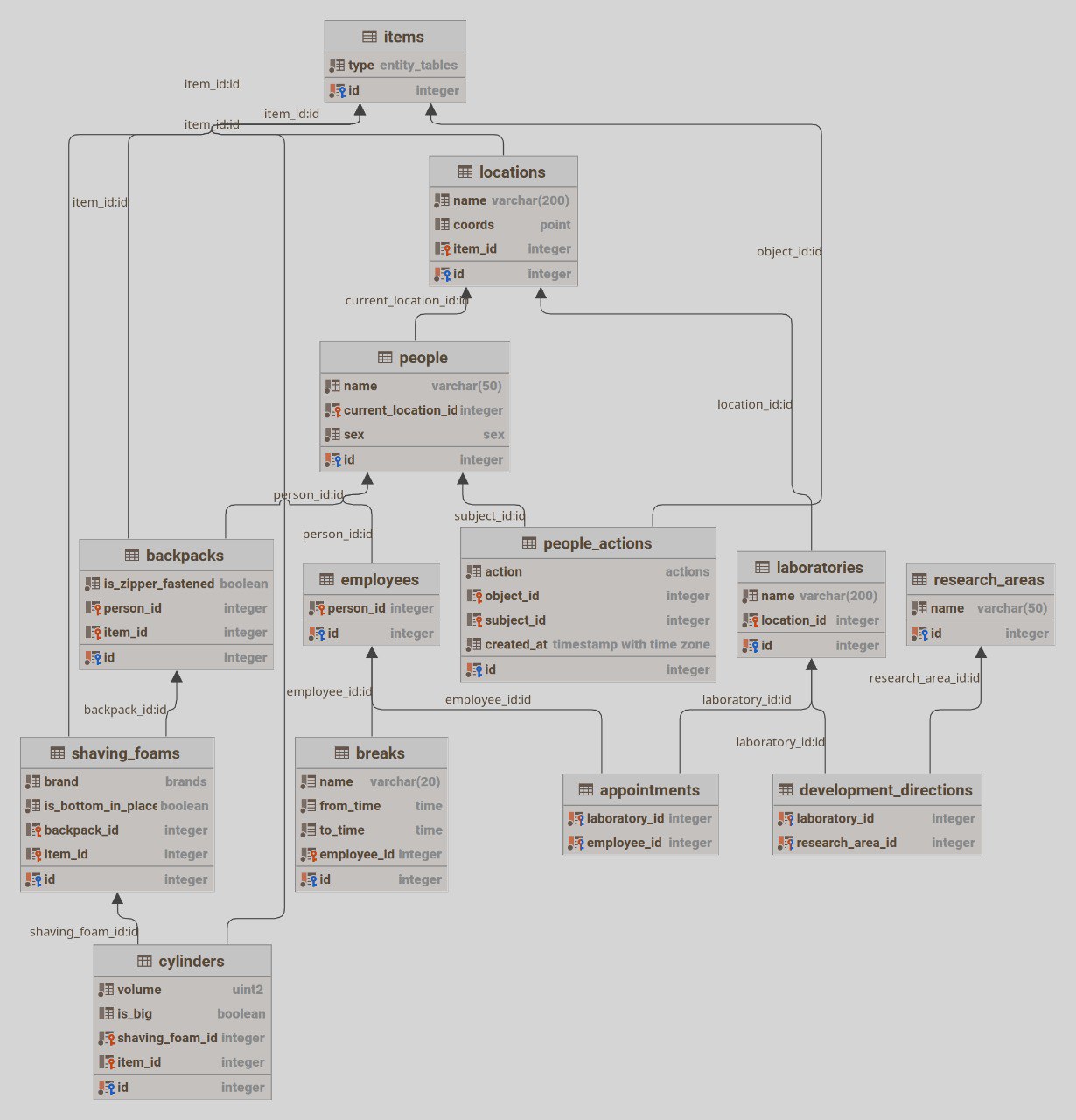
Для отношений, полученных при построении предметной области из лабораторной работы №1, выполните следующие действия:

* опишите функциональные зависимости для отношений полученной схемы (минимальное множество);
* приведите отношения в 3NF (как минимум). Постройте схему на основе 3NF (как минимум). Постройте схему на основе полученных отношений;
* опишите изменения в функциональных зависимостях, произошедшие после преобразования в 3NF (как минимум). Постройте схему на основе 3NF;
* 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 функцию, связанную с вашей предметной областью, согласуйте ее с преподавателем и реализуйте на языке PL/pgSQL.



1. **Функциональные зависимости**

|  |
| --- |
| locations: id → (name, coords, item\_id)  people: id → (name, sex, current\_location\_id)  employees: id → (person\_id)  backpacks: id → (person\_id, item\_id, is\_zipper\_fastened)  shaving\_foams: id → (brand, is\_bottom\_in\_place, backpack\_id, item\_id)  cylinders: id → (volume, is\_big, shaving\_foam\_id, item\_id)  breaks: id → (name, from\_time, to\_time, employee\_id)  people\_actions: id → (action, object\_id, subject\_id, created\_at)  laboratories: id → (name, location\_id)  research\_areas: id → (name)  appointments: (laboratory\_id, employee\_id) → ()  development\_directions: (laboratory\_id, research\_area\_id) → () |

1. **Нормальные формы**

|  |
| --- |
| 1NF: Отношение находится в 1NF, если все его атрибуты содержат только атомарные значения. Моя модель удовлетворяет 1NF, так как все атрибуты атомарны, и нет повторяющихся групп.  2NF: Отношение находится во 2NF, если оно находится в 1NF и все его неключевые атрибуты полностью функционально зависят от первичного ключа. Моя модель удовлетворяет 2NF, так как все неключевые атрибуты полностью функционально зависят от первичных ключей.  3NF: Отношение находится в 3NF, если оно находится во 2NF и не содержит транзитивных зависимостей. Моя модель удовлетворяет 3NF, так как все неключевые атрибуты зависят только от первичных ключей, и не содержат транзитивных зависимостей. |

1. **BCNF**

|  |
| --- |
| Отношение находится в BCNF,  если для каждой функциональной зависимости X →Y,  X является суперключом. Моя модель удовлетворяет BCNF, так как  для всех функциональных зависимостей X является суперключом. |

1. **Денормализация**

|  |
| --- |
| **Объединение связанных таблиц**: В некоторых случаях, объединение таблиц может уменьшить количество операций JOIN и ускорить обработку запросов. Например, можно рассмотреть объединение таблиц people и employees, если часто запрашиваются данные о человеке и его работе одновременно.  **Добавление избыточных атрибутов**: В некоторых случаях добавление избыточных атрибутов может улучшить производительность запросов. Например, если часто запрашивается количество рюкзаков отдельных людей, можно добавить атрибут backpacks\_count в таблицу people. Это позволит избежать операций подсчета при каждом запросе, однако необходимо будет обновлять этот атрибут при добавлении или удалении рюкзаков. |

1. **Функция на языке PL/pgSQL**

Функция на языке PL/pgSQL для подсчета расстояние между двумя лабораториями.

|  |
| --- |
| -- Функция на языке PL/pgSQL для подсчета расстояние между двумя лабораториями.  -- lab1\_id, lab2\_id = ID лабораторий  -- unit = единица, которую вы хотите получить для результатов, где:  -- 'M' - статутные мили (по умолчанию)  -- 'K' — километры  -- 'N' — морские мили  **CREATE** **OR** REPLACE **FUNCTION** calculate\_laboratory\_distance(lab1\_id int, lab2\_id int, units varchar)  **RETURNS** float **AS** $dist$  **DECLARE**  dist float := 0;  loc1\_id int;  loc2\_id int;  point1 point;  point2 point;  -- lat1, lon1 = широта и долгота точки 1 (в десятичных градусах)  -- lat2, lon2 = широта и долгота точки 2 (в десятичных градусах)  lat1 int;  lat2 int;  lon1 int;  lon2 int;  radlat1 float;  radlat2 float;  theta float;  radtheta float;  **BEGIN**  **SELECT** laboratories.location\_id **INTO** loc1\_id **FROM** laboratories **WHERE** laboratories.id = lab1\_id;  IF **NOT** FOUND **THEN**  RAISE EXCEPTION 'laboratory % not found', lab1\_id;  **END** IF;  **SELECT** laboratories.location\_id **INTO** loc2\_id **FROM** laboratories **WHERE** laboratories.id = lab2\_id;  IF **NOT** FOUND **THEN**  RAISE EXCEPTION 'laboratory % not found', lab2\_id;  **END** IF;  IF loc1\_id **IS** **NULL** **OR** loc2\_id **IS** **NULL** **THEN**  RAISE EXCEPTION 'No location for laboratories';  **END** IF;  **SELECT** locations.coords **INTO** point1 **FROM** locations **WHERE** locations.id = loc1\_id;  **SELECT** locations.coords **INTO** point2 **FROM** locations **WHERE** locations.id = loc2\_id;  IF point1 **IS** **NULL** **OR** point2 **IS** **NULL** **THEN**  RAISE EXCEPTION 'Null coords in laboratory location';  **END** IF;  lat1 := point1[0];  lon1 := point1[1];  lat2 := point2[0];  lon2 := point2[1];  IF lat1 = lat2 **AND** lon1 = lon2  **THEN** **RETURN** dist;  **ELSE**  radlat1 := pi() \* lat1 / 180;  radlat2 := pi() \* lat2 / 180;  theta := lon1 - lon2;  radtheta := pi() \* theta / 180;  dist := sin(radlat1) \* sin(radlat2) + cos(radlat1) \* cos(radlat2) \* cos(radtheta);  IF dist > 1 **THEN** dist = 1; **END** IF;  dist := acos(dist);  dist := dist \* 180 / pi();  dist := dist \* 60 \* 1.1515;  IF units = 'K' **THEN** dist := dist \* 1.609344; **END** IF;  IF units = 'N' **THEN** dist := dist \* 0.8684; **END** IF;  **RETURN** dist;  **END** IF;  **END**;  $dist$ **LANGUAGE** plpgsql; |

1. **Вывод**

При выполнении лабораторной работы я познакомился с понятием нормализации и денормализации. Научился определять функциональные зависимости модели, а также анализировать последнюю на соответствие различным нормальным формам. Познакомился с процедурным языком PL/pgSQL. Изучил эффективные способы денормализации схемы базы данных и ситуации, в которых возможно их применение.