Факультет Программной Инженерии и Компьютерной техники

**Лабораторная работа №3**

**Вариант №171304**

Выполнила:

Брель Мария Владимировна

Группа Р3107

Преподаватели:

Байрамова Хумай

Николаев Владимир Вячеславович

**Оглавление**

[Задание: 3](#__RefHeading___Toc1941_1931527120)

[Функциональные зависимости: 4](#__RefHeading___Toc3979_2680889073)

[Нормальные формы: 5](#__RefHeading___Toc3981_2680889073)

[Денормализация: 5](#__RefHeading___Toc3983_2680889073)

[Функция на языке PL/pgSQL: 6](#__RefHeading___Toc3985_2680889073)

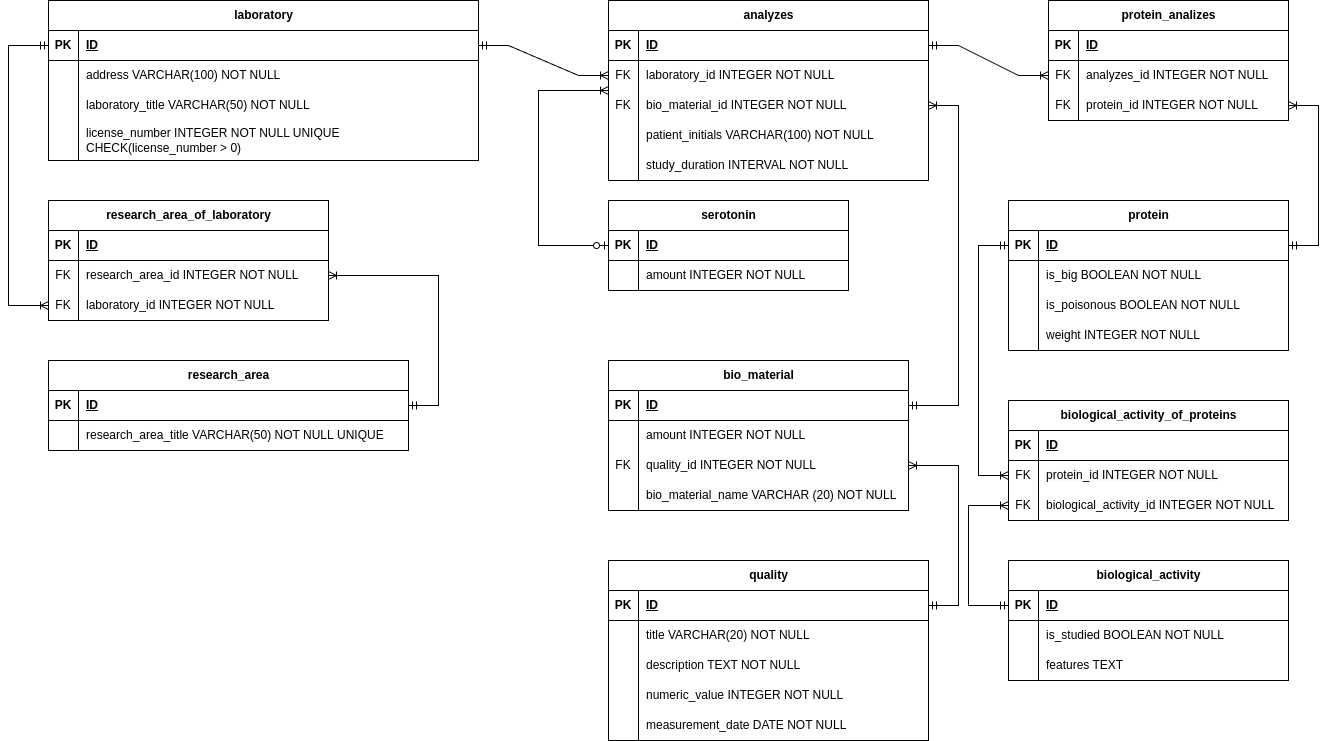
[Вывод: 7](#__RefHeading___Toc973_1545597831)

# **Задание:**

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.



# **Функциональные зависимости:**

***research\_area***: id→research\_area\_title

***research\_area\_of\_laboratory***: id → research\_area\_id

id → laboratory\_license\_number

***laboratory***: license\_number → address,

license\_number → laboratory\_title,

license\_number → id,

id → laboratory\_title,

id → address,

id → license\_number

Данные связи нарушают 3НФ, так как присутствуют транзитивные функциональные связи. Для нормализации можно сделать license\_number PRIMARY KEY.

***analyzes***: id → laboratory\_license\_number

id → bio\_material\_id

id → patient\_initials

id → study\_duration

***serotonin***: id → amount

***bio\_material***: id → amount

id → quality\_id

id → bio\_material\_name

***quality***: id → title

id → description

id →numeric\_value

id → measurment\_date

***protein\_analyzes***: id → analyzes\_id

id → protein\_id

***protein***: id → is\_big

id → is\_poisonous

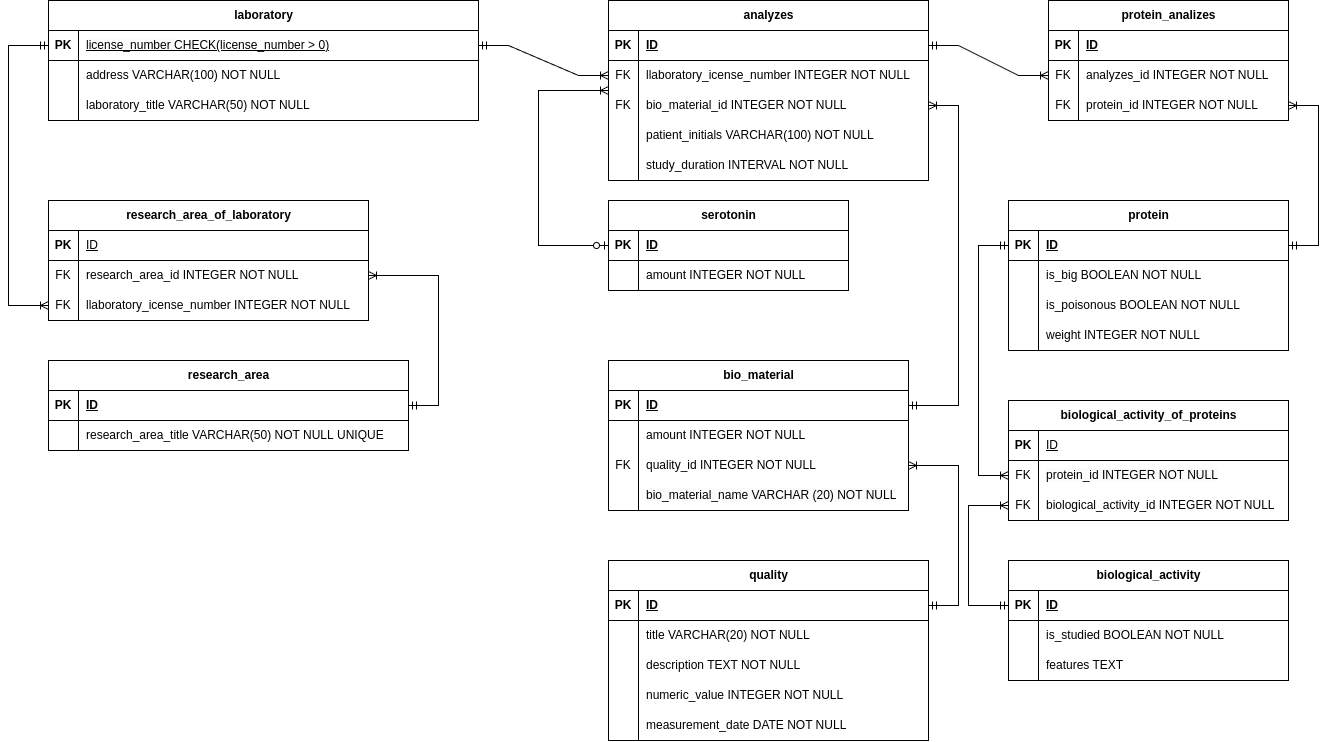
id → weight

***biological\_activity\_of\_proteins***: id → protein\_id

id → biological\_activity\_id

***biological\_activity***: id → is\_studied

id → features



# **Нормальные формы:**

***1NF:*** Отношение находится в 1NF, так как на пересечении каждой строки и столбца — одно значение.

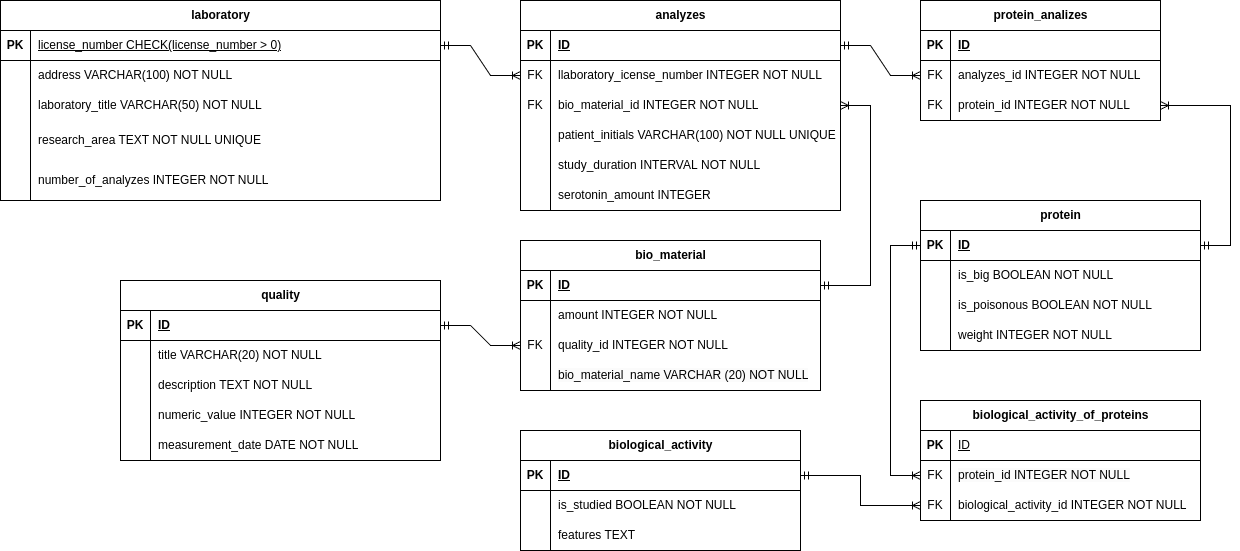
***2NF:*** Отношение находится во 2NF, так как все неключевые атрибуты в полной функциональной зависимости от первичного ключа отношения.

***3NF:*** Отношение находится в 3NF, так как все неключевые атрибуты не находятся в транзитивной функциональной зависимости от первичного ключа.

***BCNF***: Отношение находится в BCNF, так как для всех функциональных зависимостей выполняется условие: детерминант — потенциальный ключ.

# **Денормализация:**

Объединение связанных таблиц: объединение таблиц laboratory и research\_area, а так же analyzes и serotonin может быть полезно в том случае, когда данные из этих таблиц часто запрашиваются вместе, так как в таком случае без использования JOIN исполнение операций будет быстрее. Нарушает 3 нормальную форму, тк появляются атрибуты, зависящие от вторичных атрибутов.

Добавление атрибутов: добавление атрибута number\_of\_analyzes в таблицу laboratory в том случае, если данные о лаборатории и количестве анализов в ней часто запрашиваются вместе. Это так же экономит время выполнения операции. Однако этот атрибут требуется часто обновлять. Нарушает 3 НФ тк он не зависит от PK.

# **Функция на языке PL/pgSQL:**

1. Данный триггер необходим для изменения данных при соблюдении определенного условия, в данном случае для того, чтобы обнулить массу протеинов в анализе если качество био материала в нем менее 50.
2. Триггер вызывается после операции UPDATE и обновляет данные в случае, если условие соблюдается.

|  |
| --- |
| create function *quality\_checker*() returns trigger  language plpgsql as $$ BEGIN  UPDATE protein SET weight = 0 WHERE protein.id IN (  SELECT protein.id FROM protein  JOIN protein\_analyzes ON protein.id = protein\_analyzes.protein\_id  JOIN analyzes ON protein\_analyzes.analyzes\_id = analyzes.id  JOIN bio\_material ON analyzes.bio\_material\_id = bio\_material.id  JOIN quality ON bio\_material.quality\_id = quality.id  WHERE quality.numeric\_value < 50 AND quality.id = NEW.**quality\_id** );  RETURN NEW; END $$;  CREATE CONSTRAINT TRIGGER quality\_trigger AFTER UPDATE OF quality\_id ON bio\_material FOR EACH ROW  EXECUTE PROCEDURE *quality\_checker*(); |

# **Вывод:**

При выполнении данной лабораторной работы я познакомилась с нормализованными и денормализованными формами, научилась определять функциональные зависимости и проверять их на соответствие нормальным формам. Так же научилась писать функции на языке PL/pgSQL и реализовывать триггеры.