ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| должность |  |  |  | Путилова Н.В. |
| [старший преподаватель](https://guap.ru/rasp/?p=317) |  | подпись, дата |  | инициалы, фамилия |

|  |  |
| --- | --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №7 |  |
| «Хранимые процедуры. Управление доступом» |  |
| по дисциплине: Проектирование баз данных |  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. | 4236 |  |  |  | Л. Мвале |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2024

**Цель работы:** Получения навыков по серверному программированию и управлению доступом в базе данных.

**Описание задания**

1. Создать в БД хранимые процедуры, реализующие:

— вставку с пополнением справочников (вставляется информация о студенте, если указанный номер группы отсутствует в БД, запись добавляется в таблицу с 84 перечнем групп) (получаем ссылку на внешний ключ по значению данного из родительской таблицы);

— хранимую функцию любого назначения, соответствующего предметной области.

— каскадное удаление: удаление всех зависимых данных (перед удалением записи о группе удаляются записи обо всех студентах этой группы и их секциях);

2)Создать пользователя, который обладает правами, указанными в варианте задания

**Текст задания Вариант 15**

1. вузы для абитуриента: город, вуз, факультеты, направления, направленности, ЕГЭ которые нужно сдать, дата начала/конца приемной кампании.

(Направление -09.03.04 «Программная инженерия», Направленность — его конкретизация «Разработка программно-информационных систем», именно направленность закреплена за кафедрой и соответственно факультетом)

а. направленности, в которых есть слово «систем», но оно не первое

б. Кафедра, не принимающая ни на одну направленность

в. направление, на которое надо сдавать математику и информатику

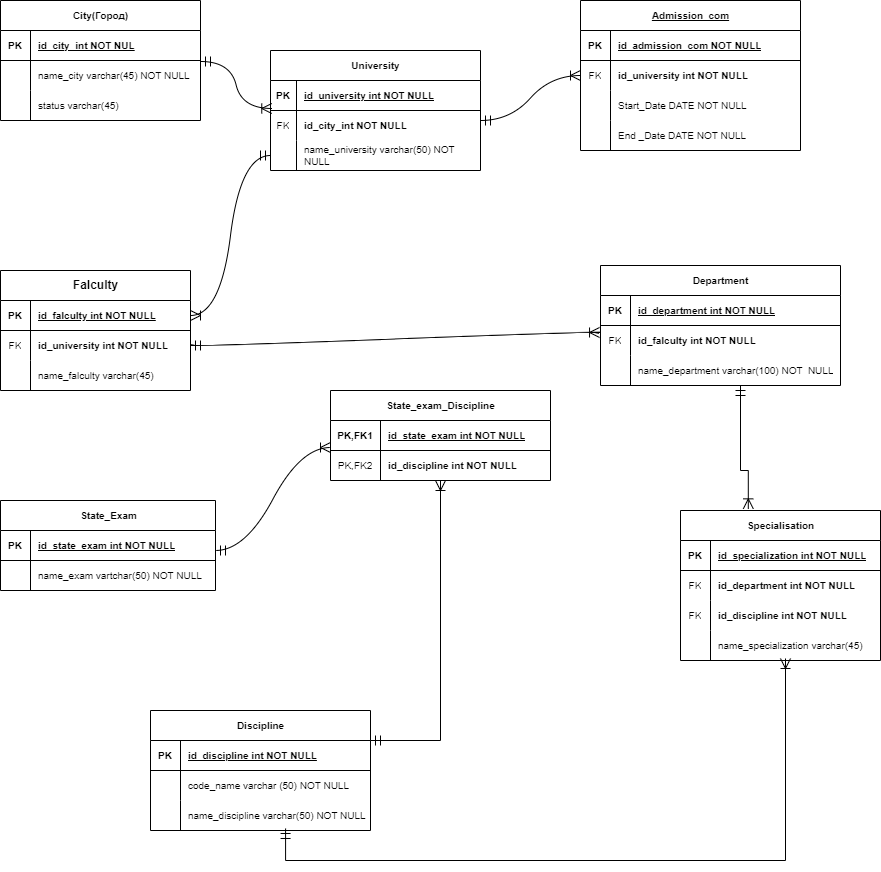
г. факультет, принимающий на количество направлений больше среднего

д. город, в котором есть все укрупненные группы направлений и специальностей(УГСН) (первые 2 цифры номера специальности, т.е у 09.03.04 УГСН=09, а у 02.03.03-02)

е. вуз, с последним по алфавиту названием

ж. направление, на которое не надо сдавать ЕГЭ по математике, но надо по иностранному языку

**Физическую модель базы данных**



**Назначение разработанных хранимых процедур или функций текстом**

**1. Процедура add\_university\_with\_city**

**Цель:**  
 Добавление университета в базу данных, с автоматическим созданием города, если он еще не существует.

**Описание:**  
 Процедура выполняет проверку существования города и университета перед добавлением новой записи:

* Если город с указанным именем (p\_city\_name) отсутствует в таблице City, процедура добавляет новый город с активным статусом, а затем получает его id\_city.
* Если университет с указанным именем (p\_university\_name) отсутствует в таблице University, процедура добавляет университет, связывая его с соответствующим городом.

**Пример использования:**

**CALL add\_university\_with\_city('Harvard University', 'Cambridge');**

**Результат:**  
 Если города "Cambridge" еще нет в базе данных, он будет добавлен в таблицу City. Затем будет добавлен университет "Harvard University", связанный с городом "Cambridge".

**2. Функция count\_faculties**

**Цель:**  
 Подсчёт количества факультетов в указанном университете.

**Описание:**  
 Функция возвращает количество записей в таблице Falculty, связанных с университетом, имя которого передается в параметре p\_university\_name.

* Сначала выполняется соединение таблиц Falculty и University по их внешнему ключу.
* Затем вычисляется количество факультетов для заданного университета.

**Пример использования:**

**SELECT count\_faculties('Harvard University') AS total\_faculties;**

**Результат:**   
 Возвращает количество факультетов, связанных с университетом "Harvard University". Например, результат может быть 5, если в университете 5 факультетов.

**3. Процедура delete\_falculty\_and\_dependencies**

**Цель:**  
 Удаление факультета и всех связанных с ним данных.

**Описание:**  
 Процедура удаляет факультет по его имени (p\_falculty\_name) и каскадно удаляет данные, связанные с факультетом, включая:

1. Специализации (Specialization), которые привязаны к департаментам факультета.
2. Департаменты (Department), связанные с факультетом.
3. Сам факультет из таблицы Falculty.

**Пример использования:**

**CALL delete\_falculty\_bi\_ID(4);**

**Результат:**  
 Процедура удаляет факультет с именем "Engineering", а также все департаменты и специализации, связанные с этим факультетом. Если факультет с таким именем не существует, процедура ничего не делает.

**Скрипт для создания хранимых процедур или функций**

-- Удаляем процедуру, если она уже существует

DROP PROCEDURE IF EXISTS add\_university\_with\_city;

DELIMITER $$

-- Создаем процедуру для добавления университета с городом

CREATE PROCEDURE add\_university\_with\_city(

    p\_university\_name *VARCHAR*(150), -- Имя университета

    p\_city\_name *VARCHAR*(20)         -- Имя города

)

BEGIN

    DECLARE city\_id *INT*;            -- Переменная для хранения ID города

    DECLARE university\_id *INT*;      -- Переменная для хранения ID университета

    -- Проверяем, существует ли город с указанным именем

    SELECT id\_city INTO city\_id

    FROM City

    WHERE name\_city = p\_city\_name;

    -- Если город не существует, добавляем его в таблицу City

    IF city\_id IS NULL THEN

        INSERT INTO City (name\_city, status)

        VALUES (p\_city\_name, 'active'); -- Добавляем город со статусом "active"

        SET city\_id = LAST\_INSERT\_ID(); -- Получаем ID только что добавленного города

    END IF;

    -- Проверяем, существует ли университет с указанным именем

    SELECT id\_university INTO university\_id

    FROM University

    WHERE name\_university = p\_university\_name;

    -- Если университет не существует, добавляем его в таблицу University

    IF university\_id IS NULL THEN

        INSERT INTO University (name\_university, id\_city)

        VALUES (p\_university\_name, city\_id); -- Связываем университет с городом

    END IF;

END$$

DELIMITER ;

DELIMITER $$

-- Создаем функцию для подсчета количества факультетов в университете

CREATE FUNCTION count\_faculties(p\_university\_name *VARCHAR*(150))

RETURNS *INT* -- Возвращаемое значение типа INT

reads sql data

BEGIN

    DECLARE faculty\_count *INT*; -- Переменная для хранения количества факультетов

    -- Выполняем подсчет факультетов для указанного университета

    SELECT COUNT(\*)

    INTO faculty\_count

    FROM Falculty f

    JOIN University u ON f.id\_university = u.id\_university

    WHERE u.name\_university = p\_university\_name;

    -- Возвращаем количество факультетов

    RETURN faculty\_count;

END$$

DELIMITER ;

-- Удаляем процедуру, если она уже существует

DELIMITER $$

-- Создаем процедуру для удаления факультета и всех зависимых данных

CREATE PROCEDURE delete\_falculty\_and\_dependencies(

    IN p\_falculty\_name *VARCHAR*(50) -- Имя факультета

)

BEGIN

    DECLARE falculty\_id *INT*; -- Переменная для хранения ID факультета

    -- Получаем ID факультета по его имени

    SELECT id\_falculty INTO falculty\_id

    FROM Falculty

    WHERE name\_falculty = p\_falculty\_name;

    -- Если факультет существует, выполняем удаление зависимых данных

    IF falculty\_id IS NOT NULL THEN

        -- Удаляем специализации, связанные с департаментами факультета

        DELETE FROM Specialization

        WHERE id\_department IN (

            SELECT id\_department FROM Department WHERE id\_falculty = falculty\_id

        );

        -- Удаляем департаменты, связанные с факультетом

        DELETE FROM Department

        WHERE id\_falculty = falculty\_id;

        -- Удаляем сам факультет

        DELETE FROM Falculty

        WHERE id\_falculty = falculty\_id;

    END IF;

END$$

DELIMITER ;

**Скрипт создания пользователя, с возможностью изменять данные только через процедуры и функции**

-- Шаг 1: Создание пользователя с указанным паролем

CREATE USER 'university\_user'@'%' IDENTIFIED BY 'password123';

-- Шаг 2: Отмена всех привилегий, чтобы пользователь не имел доступа по умолчанию

REVOKE ALL PRIVILEGES, GRANT OPTION FROM 'university\_user'@'%';

-- Шаг 3: Предоставление пользователю привилегий EXECUTE только для определенных процедур и функций

GRANT EXECUTE ON PROCEDURE `add\_university\_with\_city` TO 'university\_user'@'%';

GRANT EXECUTE ON PROCEDURE `delete\_falculty\_and\_dependencies` TO 'university\_user'@'%';

GRANT EXECUTE ON FUNCTION `count\_faculties` TO 'university\_user'@'%';

-- Шаг 4: При необходимости предоставление привилегий SELECT для таблиц, которые могут потребоваться для выполнения процедур или функций

GRANT SELECT ON `City` TO 'university\_user'@'%';

GRANT SELECT ON `University` TO 'university\_user'@'%';

GRANT SELECT ON `Falculty` TO 'university\_user'@'%';

GRANT SELECT ON `Department` TO 'university\_user'@'%';

GRANT SELECT ON `Specialization` TO 'university\_user'@'%';

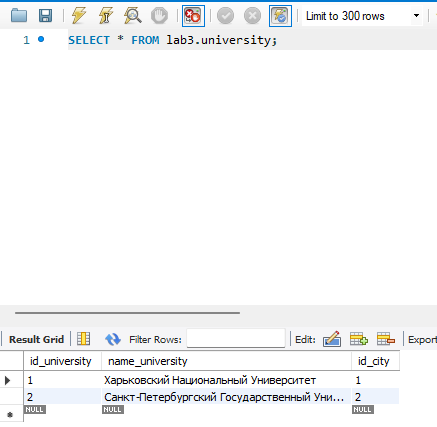
-- Шаг 5: Применение изменений

FLUSH PRIVILEGES;

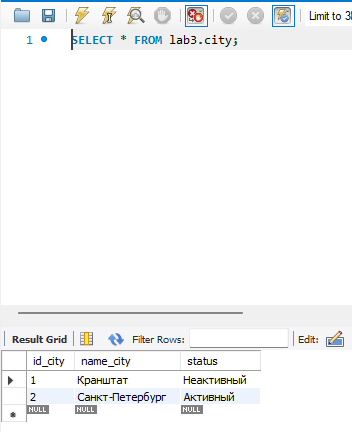
**SQL операторы и скриншоты наборов данных, иллюстрирующие работу процедур:**

**Первоначальная настройка**

***Таблица University***



***Таблица City***



**Пример 1: Добавление университета для уже существующего города**

**Ввод: CALL add\_university\_with\_city('Санкт-Петербургский Политехнический Университет', 'Санкт-Петербург');**

Процесс:

1. Проверка города: В таблице City находится город Санкт-Петербург (существует с id\_city = 2).
2. Проверка университета: Университет Санкт-Петербургский Политехнический Университет отсутствует в таблице University.
3. Действие: В таблицу University добавляется университет с id\_city = 2.

**Результат:**

Таблица City (без изменений):

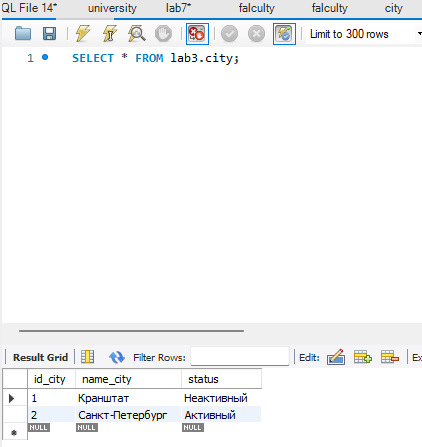
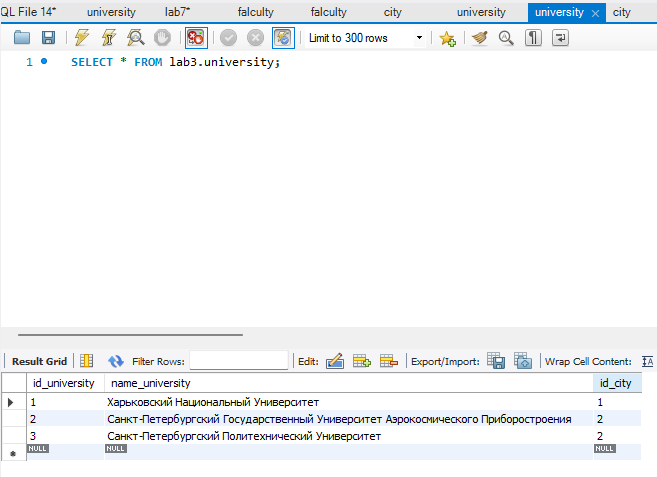


Таблица University (обновлена):



**Пример 2: Добавление университета для нового города**

**Ввод: CALL add\_university\_with\_city('Новосибирский Государственный Университет', 'Новосибирск');**

Процесс:

1. Проверка города: В таблице City отсутствует город Новосибирск.
2. Действие (город): В таблицу City добавляется город Новосибирск со статусом active.
3. Проверка университета: Университет Новосибирский Государственный Университет отсутствует в таблице University.
4. Действие (университет): В таблицу University добавляется университет с id\_city, полученным с помощью LAST\_INSERT\_ID().

**Результат:**

Таблица City (обновлена):

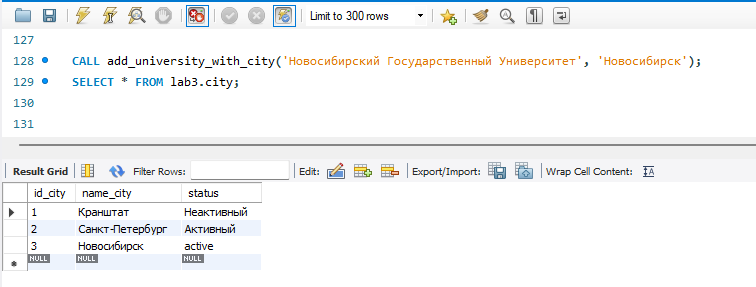
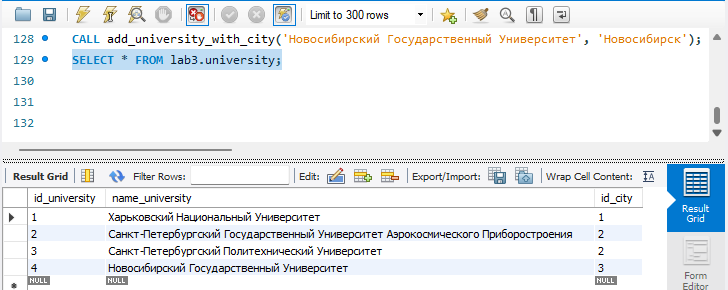


Таблица University (обновлена):



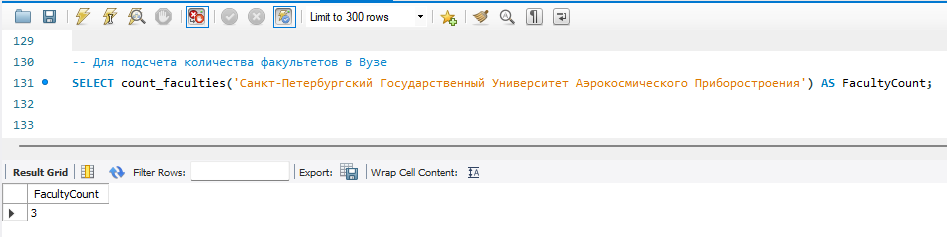
**Вызов функции**

Для подсчета количества факультетов, например, для

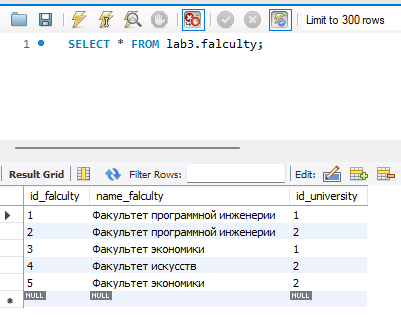
***“Санкт-Петербургский Государственный Университет Аэрокосмического Приборостроения”,*** выполните:

**SELECT count\_faculties('Санкт-Петербургский Государственный Университет Аэрокосмического Приборостроения') AS FacultyCount;**

Результат:



И это видно из таблиц Falculty так как этот вуз имеет id\_university =2



**Пример вызова процедуры**

**delete\_falculty\_and\_dependencies с демонстрацией изменений в таблицах**

**Исходные данные в таблицах**

Таблица Falculty:

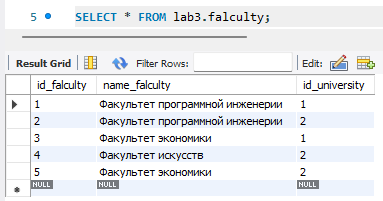


Таблица Department:

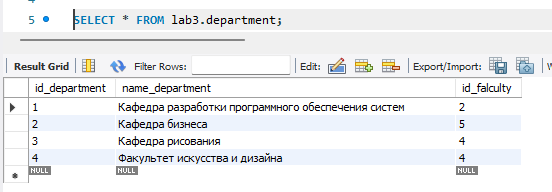
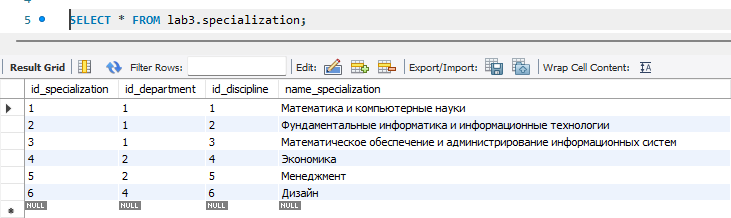


Таблица Specialization:



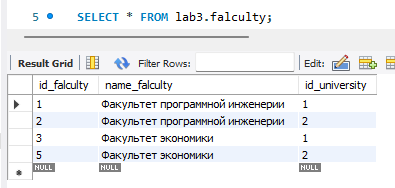
**Вызов процедуры**

Удалим факультет "Факультет искусств":

**CALL delete\_falculty\_and\_dependencies('Факультет искусств');**

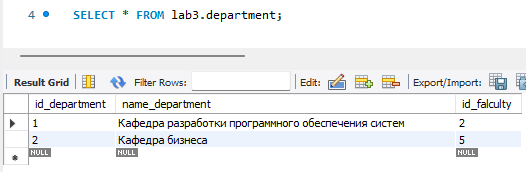
### **Изменения в таблицах**

#### Таблица Falculty после удаления:



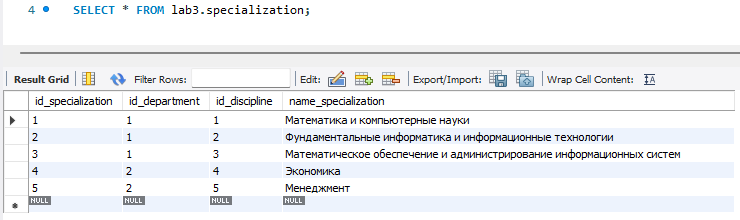
Факультет с id\_falculty = 4 удалён.

Таблица Department после удаления:



Департаменты, связанные с удалённым факультетом (id\_falculty = 4), также удалены.

Таблица Specialization после удаления:



Специализации, связанные с удалёнными департаментами (id\_department = 3 и id\_department = 4), также удалены.

**Итог**

После вызова процедуры удалены:

1. Факультет с названием "Факультет искусств".
2. Все департаменты, связанные с этим факультетом.
3. Все специализации, связанные с департаментами удалённого факультета.

Если нужно удалить другой факультет, процедура будет работать аналогично.

**SQL операторы и скриншоты наборов данных, отображающие права пользователя и демонстрирующие его работу с процедурами работами.**

-- Доступ к таблицам:

SELECT

    grantee AS user,

    privilege\_type,

    table\_name AS name,

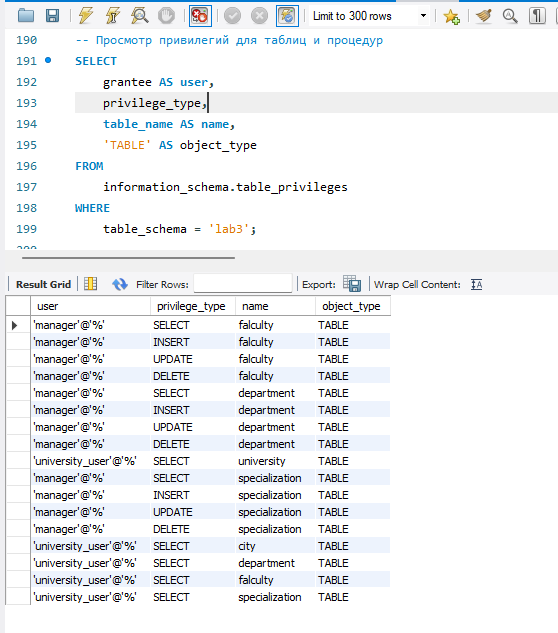
    'TABLE' AS object\_type

FROM

    information\_schema.table\_privileges

WHERE

    table\_schema = 'lab3';



-- Проверка доступа к процедурам

SELECT

    CONCAT(User, '@', Host) AS role\_name,

    Db AS procedure\_schema,

    Routine\_name AS procedure\_name,

    Routine\_type AS procedure\_type,

    Proc\_priv AS privileges,

    CASE

        WHEN FIND\_IN\_SET('Execute', Proc\_priv) > 0 THEN 'Yes'

        ELSE 'No'

    END AS can\_execute

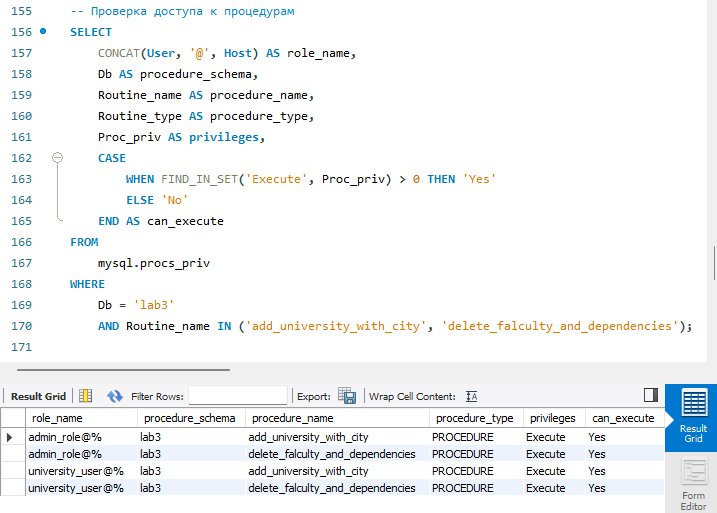
FROM

    mysql.procs\_priv

WHERE

    Db = 'lab3'

    AND Routine\_name IN ('add\_university\_with\_city', 'delete\_falculty\_and\_dependencies');



-- Проверка доступа к функций

SELECT

    CONCAT(User, '@', Host) AS role\_name,

    Db AS function\_schema,

    Routine\_name AS function\_name,

    Routine\_type AS object\_type,

    Proc\_priv AS privileges,

    CASE

        WHEN FIND\_IN\_SET('Execute', Proc\_priv) > 0 THEN 'Yes'

        ELSE 'No'

    END AS can\_execute

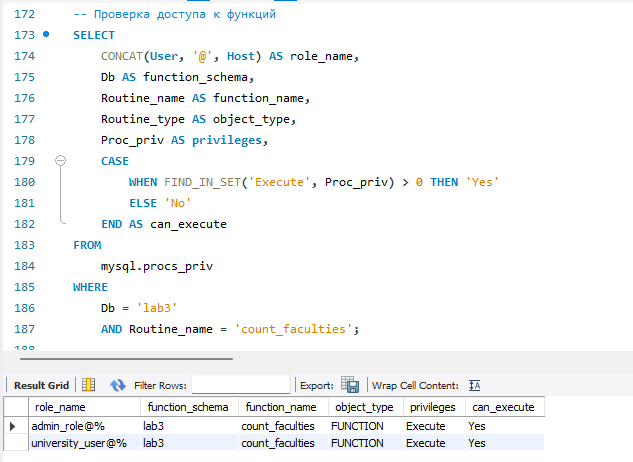
FROM

    mysql.procs\_priv

WHERE

    Db = 'lab3'

    AND Routine\_name = 'count\_faculties';



**Используя имя роли university\_user@%**

**Исходные данные в таблицах**

Таблица City:

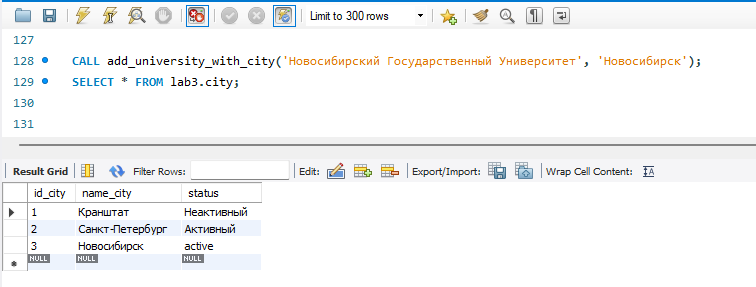
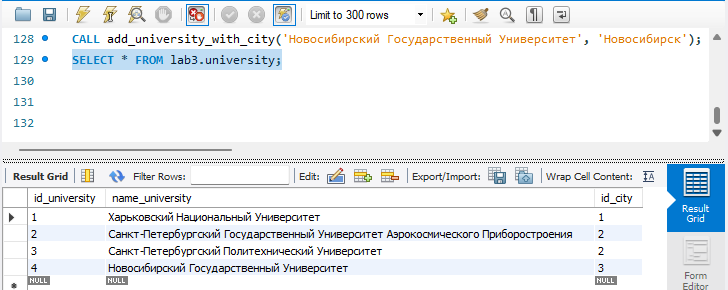
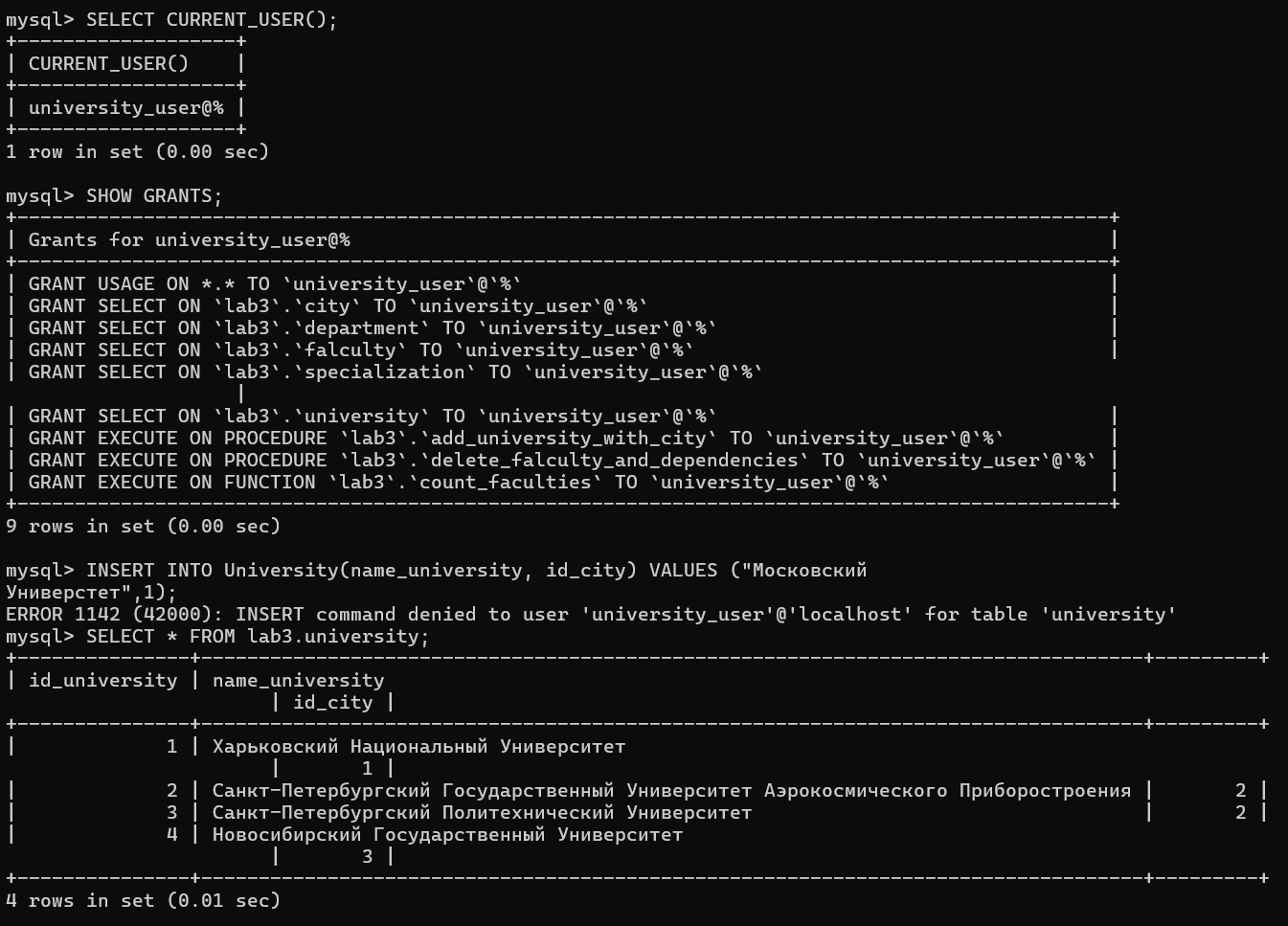
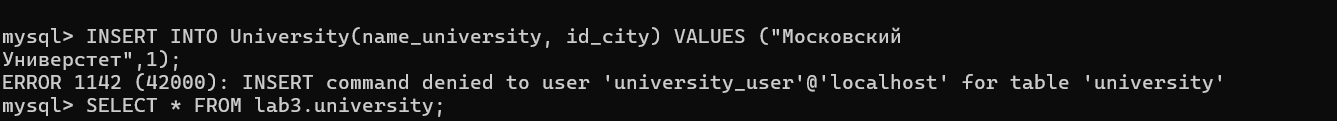


Таблица University (обновлена):

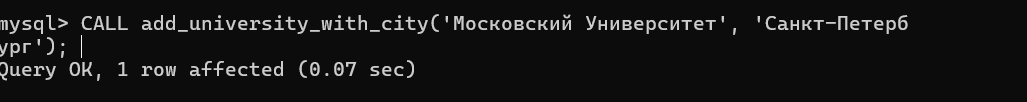




Попытался напрямую добавить университет, используя метод insert, но ничего не вышло, так как у меня нет на это прав



Затем я использовал процедуру add\_university\_with\_city, к которой мне предоставлен доступ, и это сработало. Я добавил к уже существующему городу



**Результат:**

Таблица City (без изменений):

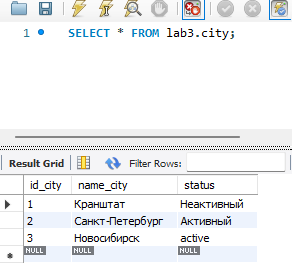
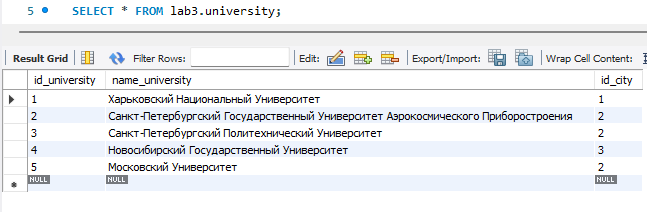
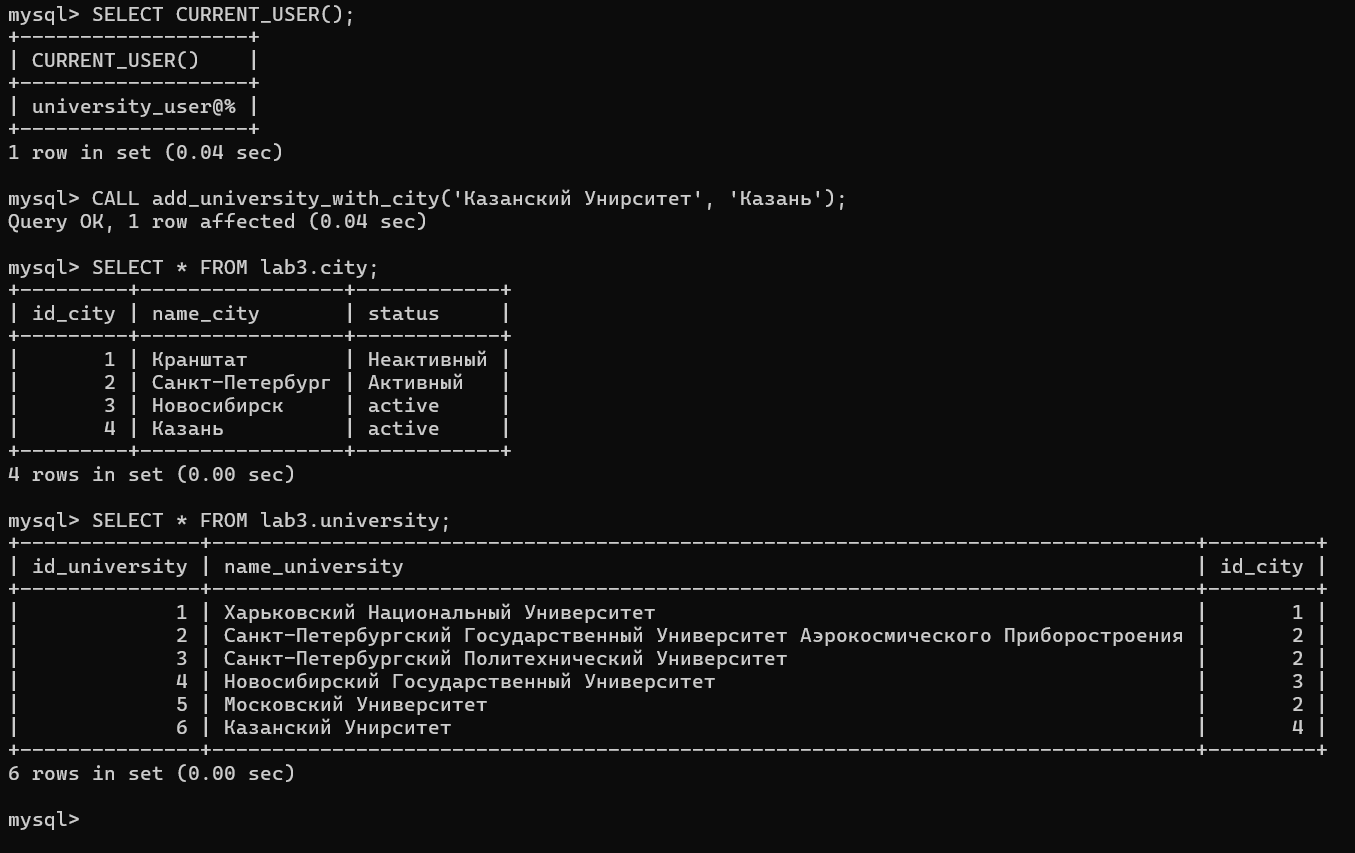


Таблица University (обновлена):



Теперь давайте воспользуемся процедурой add\_university\_with\_city для города, который не существует

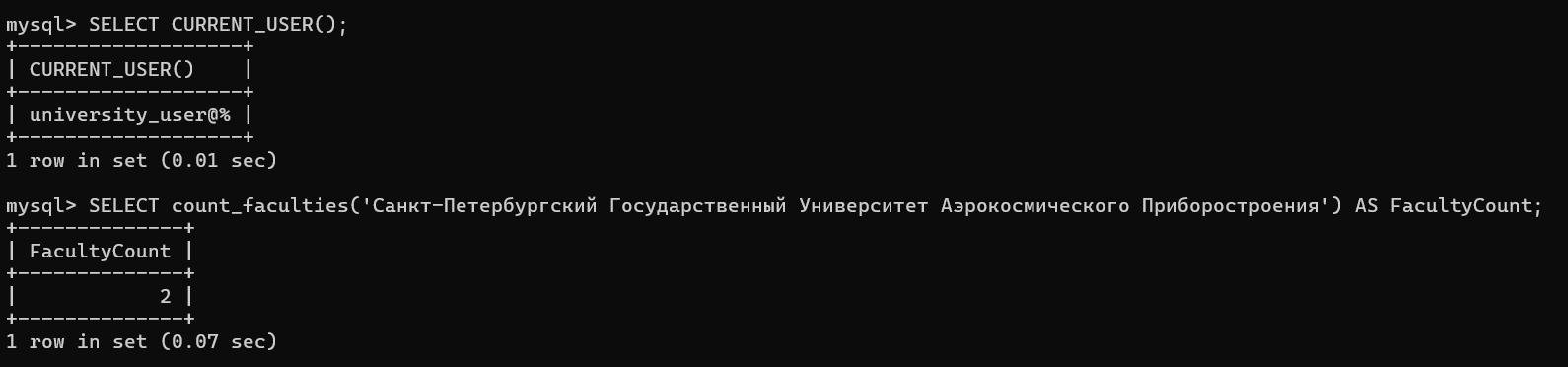


Мы видим, что добавлен новый город и успешно добавлен университет

**Вызов функции пользователем university\_user**

Для подсчета количества факультетов, например, для

***“Санкт-Петербургский Государственный Университет Аэрокосмического Приборостроения”,*** выполните:



**Пример вызова процедуры**

**delete\_falculty\_and\_dependencies с демонстрацией изменений в таблицах пользователем university\_user**

**Исходные данные в таблицах**

Таблица Falculty:

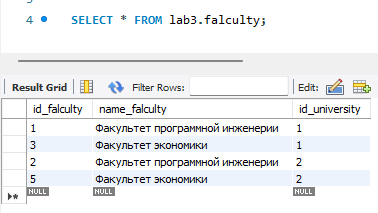


Таблица Department:

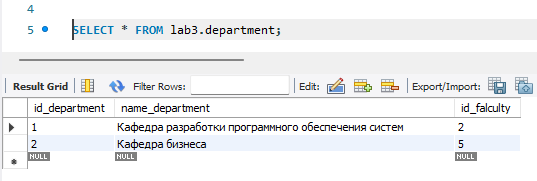
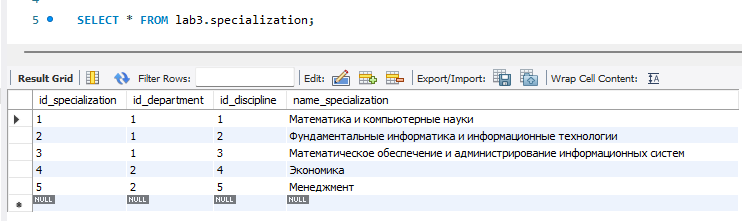


Таблица Specialization:

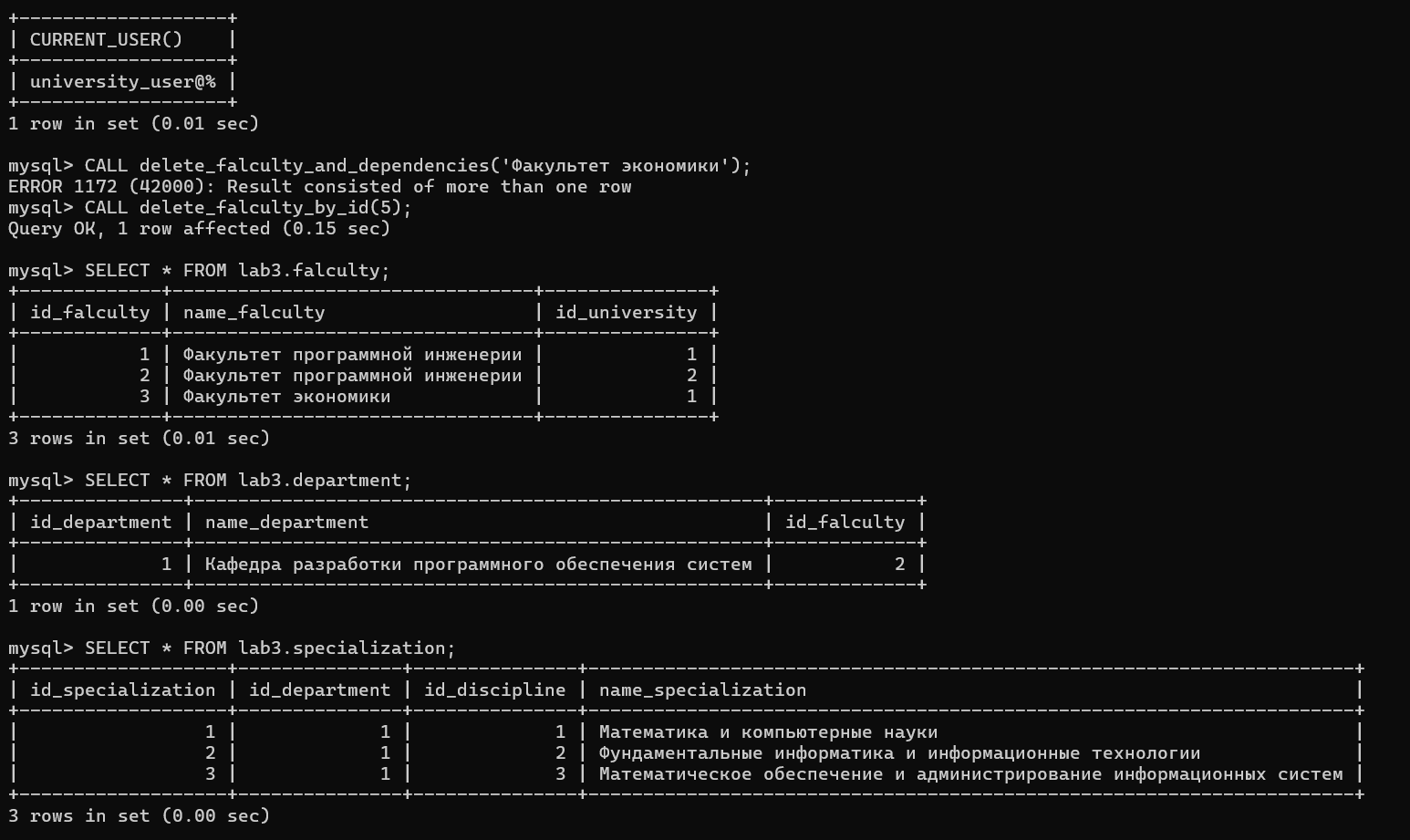


**Вызов процедуры**

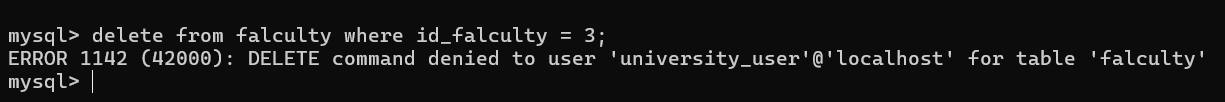
Удалим факультет "Факультет искусств":

**CALL delete\_falculty\_by\_id(5);**

### **Изменения в таблицах**



Если я попытаюсь удалить файл без использования этой процедуры, мне будет отказано в доступе



**Выводы об использовании процедур и управление доступом в разработанной мной базе данных**

В ходе работы с разработанной базой данных были изучены аспекты использования хранимых процедур и управление правами доступа для пользователей. Созданные процедуры облегчают выполнение операций по вставке, обновлению и удалению данных, что способствует упрощению взаимодействия с базой данных и повышению ее безопасности. Ограничив доступ к таблицам и предоставив пользователям права на выполнение только необходимых процедур, мы обеспечили защиту данных от несанкционированного доступа, позволяя пользователям выполнять только те действия, которые необходимы для их работы. Это подчёркивает важность правильной настройки управления доступом для эффективного взаимодействия с базами данных.