САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО

Институт компьютерных наук и кибербезопасности

Высшая школа технологий искусственного интеллекта

Направление: 02.03.01 Математика и компьютерные науки

Отчет по лабораторным работам
по дисциплине: «Методы проектирования баз данных»

Студент:
группы 5130201/20102
_____ Салимли А.

Преподаватель:
к.т.н, доц.
_____ Попов С.Г.

«____»_____2024 г.

Санкт-Петербург, 2024 г.

Содержание

Введение

В данном отчете, описаны результаты выполнения комплекса лабораторных работ, расширяющих функциональные возможности базы данных на тему «электронная подача заявления на поступление в ВУЗ». Которая была спроектирована в течении предыдущего семестрового курса «Теоретические основы баз данных».

В ходе выполнения лабораторных работ, изучены и реализованы в СУБД: Представления, событийная модель (триггеры), права доступа, функция, процедура и транзакции.

1 Постановка задачи

В ходе прохождения данного курса, необходимо выполнить пять лабораторных работ:

1. Создать представление, инкапсулирующее запрос. Написать запрос, использующий в себе представление. Представление отображает таблицу о количестве вступительных испытаний абитуриентом и количеством институтов в которые подал абитуриент.
2. Написать триггеры, автоматизирующие сбор статистической информации о численности абитуриентов подавших заявление в конкретный институт.
3. Создать двух пользователей. Первый должен иметь доступ только на просмотр ранее созданного представления в пером задании. Второй пользователь так же может просматривать представление, помимо этого, он должен уметь редактировать все таблицы, участвующие в запросе представления.
4. Создать функцию которая принимает ФИО абитуриента, и выводит конкатинированные инициалы имени, отчества и полную фамилию. Так же создать процедуру, в которую передаются все атрибуты института и направления, после рассматриваются четыре варианта событий - 1. Есть такой институт и такое направление. 2. Есть такой институт, нет такого направления. 3. Нет такого института, есть такое направление. 4. Нет такого института и нет такого направления.
5. Задать уровень изоляции транзакции как read commited и проверить, выполняет ли этот уровень защиту от феномена - неустойчивое чтение, (non-repeatable read).

2 Лабораторные работы

2.1 Работа 1: Создание представлений

Задача: Разработать представление для хранения запроса внутри СУБД.

Формулировка: Для каждого абитуриента, посчитать число его вступительных испытаний и число институтов в которые он подал документы.

Сделано представление (view), E_Stats1, которое хранит запрос, считающий число вступительных испытаний абитуриента и число институтов в которые подавал абитуриент.

```
CREATE VIEW E_Stats1 AS
SELECT
    e.enrollee_id,
    CONCAT(e.last_name, ' ', e.first_name, ' ', e.middle_name) AS enrollee_name,
    COALESCE(a.exam_count, 0) AS exam_count,
    COALESCE(d.institute_count, 0) AS institute_count
FROM
    Enrollee e
LEFT JOIN (
    SELECT enrollee_id, COUNT(achievment_id) AS exam_count
    FROM Achievment
    GROUP BY enrollee_id
) a ON e.enrollee_id = a.enrollee_id
LEFT JOIN (
    SELECT c.enrollee_id, COUNT(s.department_id) AS institute_count
    FROM Choice c
    JOIN Specialty s ON c.specialty_id = s.specialty_id
    GROUP BY c.enrollee_id
) d ON e.enrollee_id = d.enrollee_id;
```

Представление создает таблицу с колонками id абитуриента, количество вступительных испытаний и количество институтов. Результат выполнение первых 5 абитуриентов в таблице 1.

enrollee_id	enrollee_name	exam_count	institute_count
200083	Nepomnyashchikh Mikhail Mikhailovich	3	2
200084	Bakinets Irina Ivanovich	3	5
200085	Istov Ayzek Viktorovich	5	1
200086	Istov Viktor Viktorovich	5	4
200087	Kapustin Ayzek Petrovich	5	4

Таблица 1

Представление используется в следующем запросе, в котором добавляется информация о статусе подачи документов абитуриента.

```
SELECT
    es.enrollee_id,
    es.enrollee_name,
    es.exam_count,
    es.institute_count,
    e.submitted_docs_id
FROM
```

```
    E_Stats1 es
JOIN
    Enrollee e ON es.enrollee_id = e.enrollee_id
WHERE
    es.institute_count > 1;
```

Результат выполнения запроса для первых 5 абитуриентов, где, статус 6 - подал документы, 5 - не подавал, представлен в таблице 2.

enrollee_id	enrollee_name	exam_count	institute_count	submitted_docs_id
200083	Nepomnyashchikh Mikhail Mikhailovich	3	2	5
200084	Bakinets Irina Ivanovich	3	5	5
200086	Istov Viktor Viktorovich	5	4	5
200087	Kapustin Ayzek Petrovich	5	4	5
200091	Kapustin Ayzek Gennadyevich	4	5	6

Таблица 2

Ниже приведен «explain» запроса в рисунке 1.

id	select_type	table	partitions	type	possible_keys
1	PRIMARY	e	NULL	ALL	PRIMARY
1	PRIMARY	e	NULL	eq_ref	PRIMARY
1	PRIMARY	<derived4>	NULL	ref	<auto_key0>
1	PRIMARY	<derived3>	NULL	ref	<auto_key0>
3	DERIVED	s	NULL	index	PRIMARY
3	DERIVED	c	NULL	ref	enrollee_id,specialty_id
4	DERIVED	achievment	NULL	index	enrollee_id

key	key_len	ref	rows	filtered	Extra
NULL	NULL	NULL	199682	100.00	NULL
PRIMARY	4	student_adm.e.enrollee_id	1	100.00	NULL
<auto_key0>	4	student_adm.e.enrollee_id	10	100.00	NULL
<auto_key0>	4	student_adm.e.enrollee_id	10	100.00	Using where
department_id	4	NULL	10	100.00	Using index; Using tempc
specialty_id	4	student_adm.s.specialty_id	66586	100.00	NULL
enrollee_id	4	NULL	797952	100.00	Using index

Explain запроса с view. Рис.1

Изменение и удаление данных из таблицы представления, должно приводить к ошибке, так как view, это исполнение запроса, а не реальная существующая таблица. Ниже приведены ошибки при попытке добавления удаления и изменения данных.

```
mysql> INSERT INTO E_Stats1 (enrollee_id, enrollee_name, exam_count, institute_count)
    -> VALUES (123, 'Ivanov Ivan Ivanovich', 3, 2);
ERROR 1471 (HY000): The target table E_Stats1 of the INSERT is not insertable-into


mysql> DELETE FROM E_Stats1 WHERE enrollee_id = 123;
ERROR 1288 (HY000): The target table E_Stats1 of the DELETE is not updatable


mysql> UPDATE E_Stats1
    -> SET exam_count = 5
```

```
    -> WHERE enrollee_id = 123;
ERROR 1288 (HY000): The target table E_Stats1 of the UPDATE is not updatable
```

2.2 Работа 2: Событийная модель, триггеры

Задача: Разработать триггер, производящий манипуляцию над БД, при добавлении, удалении и обновлении данных.

Формулировка задачи: Обновлять статистику о числе абитуриентов в каждом институте, при подачи документов или не поданных, и обновлении статуса подачи документов.

Скрипт ниже, создает таблицу, которая содержит колонку - институт и колонку - число абитуриентов.

```
CREATE TABLE de_count_tr (
    department_id INT NOT NULL,
    department_name VARCHAR(80) NOT NULL,
    enrollee_count INT UNSIGNED DEFAULT 0,
    PRIMARY KEY (department_id),
    FOREIGN KEY (department_id) REFERENCES Department(department_id)
        ON UPDATE RESTRICT ON DELETE RESTRICT
);
```

Далее нам необходимо заполнить новую таблицу данными, ниже приведен часть кода на Python, который с помощью библиотеки «mysql» заполняет таблицу de_count_tr готовыми данными из ранее существующих таблиц.

```
    with connection.cursor() as cursor:
        cursor.execute("TRUNCATE TABLE de_count_tr")
        query = """
            INSERT INTO de_count_tr (department_id, department_name, enrollee_count)
            SELECT d.department_id, d.name, COUNT(e.enrollee_id)
            FROM Department d
            LEFT JOIN Specialty s ON d.department_id = s.department_id
            LEFT JOIN Choice c ON s.specialty_id = c.specialty_id
            LEFT JOIN Enrollee e ON c.enrollee_id = e.enrollee_id
            WHERE e.submitted_docs_id = 6  -- Статус 'added'
            GROUP BY d.department_id, d.name
        """
        cursor.execute(query)

        connection.commit()
        print("Таблица de_count_tr успешно заполнена.")
```

Триггеры будут вызваны при добавлении, удалении и обновлении строк в таблице de_count_tr. При добавлении одного нового абитуриента в таблицу, к числу абитуриентов с статусом подал документы, будет прибавлена единица. При удалении одного абитуриента в колонке с числом абитуриентов отнимается единица. Так же при обновлении статуса абитуриента с «6» на «5» (с подал документы на не подавал), значение числа абитуриентов уменьшается на единицу, соответственно при обновлении статуса с «5» на «6» (с не подавал на подал документы), значение числа абитуриентов увеличивается на единицу. При добавлении нового института, в колонку

department_name будет добавлен новый институт (начальное количество абитуриентов в новом институте - ноль), при удалении нового института, будет удален из колонки выбранный институт.
Ниже приведены скрипты всех четырех триггеров:

Изменение статуса абитуриента на «подал документы»

```
DELIMITER //
CREATE TRIGGER update_enrollee_count_after_update
AFTER UPDATE ON Enrollee
FOR EACH ROW
BEGIN
    IF OLD.submitted_docs_id = 6 THEN
        UPDATE de_count_tr
        SET enrollee_count = enrollee_count - 1
        WHERE department_id IN (
            SELECT s.department_id
            FROM Specialty s
            JOIN Choice c ON s.specialty_id = c.specialty_id
            WHERE c.enrollee_id = OLD.enrollee_id
        );
    END IF;
    IF NEW.submitted_docs_id = 6 THEN
        UPDATE de_count_tr
        SET enrollee_count = enrollee_count + 1
        WHERE department_id IN (
            SELECT s.department_id
            FROM Specialty s
            JOIN Choice c ON s.specialty_id = c.specialty_id
            WHERE c.enrollee_id = NEW.enrollee_id
        );
    END IF;
END //
```

Добавление нового абитуриента

```
DELIMITER //
CREATE TRIGGER UECAF2
AFTER INSERT ON Choice
FOR EACH ROW
BEGIN
    DECLARE new_department_id INT;
    SELECT s.department_id INTO new_department_id
    FROM Specialty s
    WHERE s.specialty_id = NEW.specialty_id
    LIMIT 1;

    IF (SELECT submitted_docs_id FROM Enrollee WHERE enrollee_id =
NEW.enrollee_id) = 6 THEN
        UPDATE de_count_tr
        SET enrollee_count = enrollee_count + 1
        WHERE department_id = new_department_id;
    END IF;
END //
DELIMITER ;
```

Удаление абитуриента

```
DELIMITER //
CREATE TRIGGER UECAD2
AFTER DELETE ON Choice
FOR EACH ROW
BEGIN
    DECLARE old_department_id INT;
    SELECT s.department_id INTO old_department_id
    FROM Specialty s
    WHERE s.specialty_id = OLD.specialty_id
    LIMIT 1;
    IF (SELECT submitted_docs_id FROM Enrollee WHERE enrollee_id =
OLD.enrollee_id) = 6 THEN
        UPDATE de_count_tr
        SET enrollee_count = enrollee_count - 1
        WHERE department_id = old_department_id;
    END IF;
END //
DELIMITER ;
```

Добавление нового института

```
DELIMITER //
CREATE TRIGGER add_new_department
AFTER INSERT ON Department
FOR EACH ROW
BEGIN
    INSERT INTO de_count_tr (department_id, department_name,
enrollee_count)
    VALUES (NEW.department_id, NEW.name, 0);
END //
DELIMITER ;
```

Удаление института

```
DELIMITER //
DROP TRIGGER IF EXISTS delete_department;

CREATE TRIGGER delete_department
BEFORE DELETE ON Department
FOR EACH ROW
BEGIN
    DELETE FROM de_count_tr
    WHERE department_id = OLD.department_id;
END //
DELIMITER ;
```

Ниже приведены запросы для проверки работоспособности триггера, запросы на, изменение статуса абитуриента, вставка нового абитуриента, удаление абитуриента, вставка нового института, удаление института, соответсвенно.

```
UPDATE Enrollee
SET submitted_docs_id = 5 --или на 6--
WHERE enrollee_id = 200083;
```

Запрос на замену статуса абитуриента.

```
SELECT d.department_id, d.department_name, d.enrollee_count
```

```
FROM de_count_tr d
JOIN Specialty s ON s.department_id = d.department_id
JOIN Choice c ON s.specialty_id = c.specialty_id
WHERE c.enrollee_id = 200083;
```
Запрос-проверка после изменения статуса.

```
INSERT INTO Enrollee (first_name, last_name, middle_name,
birth_date, submitted_docs_id)
VALUES ('Alexey', 'Ivanov', 'Alekseevich', '1999-12-25', 6);
SELECT LAST_INSERT_ID();
INSERT INTO Choice (enrollee_id, specialty_id, priority_index)
VALUES (LAST_INSERT_ID(), 1, 1);
SELECT * FROM de_count_tr;
```
Запрос на добавление нового абитуриента.

```
DELETE FROM Choice WHERE enrollee_id = (SELECT enrollee_id FROM
Enrollee WHERE submitted_docs_id = 6 LIMIT 1);
DELETE FROM Enrollee WHERE submitted_docs_id = 6 LIMIT 1;
SELECT * FROM de_count_tr;
```
Запрос на удаление абитуриента.

```
INSERT INTO Department (name, phone_number, address, headmaster_name)
VALUES ('New Institute', '1234567890', '123 New Street, City', 'Dr.
New Headmaster');
SELECT * FROM de_count_tr WHERE department_name = 'New Institute';
```
Запрос на вставку нового института.

```
DELETE FROM Department WHERE department_id = LAST_INSERT_ID();
```
Запрос на удаление института.

Ниже приведены результаты реагирования триггеров на обновление статуса(таб.4), добавление(таб.5) и удаление(таб. 6) абитуриента, добавление(таб.7) и удаление института(таб.8).

department_id	department_name	enrollee_count
12	IKNK	30237
15	IBSiB	30306
16	IMPP	120726
17	IMMiT	30098
19	ISI	29873
21	IFKiS	29965
22	IPMEiT	30119

Исходная таблица, таб.3

department_id	department_name	enrollee_count
12	IKNK	30237
15	IBSiB	30306
16	IMPP	120727
17	IMMiT	30098
19	ISI	29873
21	IFKiS	29966
22	IPMEiT	30119

Изменение статуса на 6(IMPP), таб. 4

department_id	department_name	enrollee_count
12	IKNK	30237
15	IBSiB	30306
16	IMPP	120728
17	IMMiT	30098
19	ISI	29873
21	IFKiS	29966
22	IPMEiT	30119

Вставка абитуриента (IMPP), таб. 5

department_id	department_name	enrollee_count
12	IKNK	30237
15	IBSiB	30306
16	IMPP	120727
17	IMMiT	30098
19	ISI	29873
21	IFKiS	29965
22	IPMEiT	30119

Удаление абитуриента (IMPP), таб. 6

department_id	department_name	enrollee_count
12	IKNK	30237
15	IBSiB	30306
16	IMPP	120727
17	IMMiT	30098
19	ISI	29873
21	IFKiS	29965
22	IPMEiT	30119
25	New Institute	0

Вставка института, таб. 7

department_id	department_name	enrollee_count
12	IKNK	30237
15	IBSiB	30306
16	IMPP	120727
17	IMMiT	30098
19	ISI	29873
21	IFKiS	29965
22	IPMEiT	30119

Удаление института, таб. 8

Все триггеры хранятся в информационной схеме (Information_schema). Просмотреть все имена триггеров можно следующим запросом:

```
SELECT TRIGGER_NAME
    -> FROM information_schema.TRIGGERS;
```

Результат запроса в таблице 9:

```
+-----------------------------------+
| TRIGGER_NAME                      |
+-----------------------------------+
| sys_config_insert_set_user        |
| sys_config_update_set_user        |
| update_enrollee_count_after_update |
| add_new_department                |
| UECAF2                            |
| UECAD2                            |
| delete_department                 |
+-----------------------------------+
```

Имена триггеров, таб. 9

2.3 Работа 3: Разграничение прав доступа

Первый пользователь должен уметь читать представление, а второй - редактировать таблицы, участвующие в представление.

Создаются два пользователя - viewSmotr и coolView. Первому пользователю даются права только на просмотр представления, а второму на просмотр представления и редактирование таблиц представления: Choice, Achievment, Enrollee и Specialty.

Ниже приведен скрипт создающий данных пользователей:

‘viewSmotr’@‘localhost’:

```
CREATE USER 'viewsmotr'@'localhost' IDENTIFIED BY 'Ayzek123321';
GRANT SELECT ON student_adm.E_Stats1 TO 'viewsmotr'@'localhost';
FLUSH PRIVILEGES;
```

‘coolView’@‘localhost’:

```
CREATE USER 'coolView'@'localhost' IDENTIFIED BY 'Ayzek123321';
GRANT SELECT ON student_adm.E_Stats1 TO 'coolView'@'localhost';
GRANT SELECT, INSERT, UPDATE, DELETE ON student_adm.Enrollee TO
'coolView'@'localhost';
GRANT SELECT, INSERT, UPDATE, DELETE ON student_adm.Achievment TO
'coolView'@'localhost';
GRANT SELECT, INSERT, UPDATE, DELETE ON student_adm.Choice TO
'coolView'@'localhost';
GRANT SELECT, INSERT, UPDATE, DELETE ON student_adm.Specialty TO
'coolView'@'localhost';
FLUSH PRIVILEGES;
```

В таблице 10, представлено сравнение реакций на различные действия пользователей viewSmotr и coolView.

Действие	Реакция coolView	Реакция viewSmotr
Просмотр таблицы View `SELECT * FROM E_Stats1;`	`200000 rows in set (1,15 sec)`	`200000 rows in set (1,15 sec)`
Вставка новой дисциплины `INSERT INTO Subject (name) VALUES ('Математика');`	`ERROR 1142 (42000): INSERT command denied to user 'coolView'@'localhost' for table 'subject'`	`ERROR 1142 (42000): INSERT command denied to user 'viewsmotr'@'localhost' for table 'subject'`
Удаление выбора специальности `DELETE FROM Choice WHERE choice_id = 1;`	`Query OK, 0 rows affected (0,00 sec)`	`ERROR 1142 (42000): DELETE command denied to user 'viewsmotr'@'localhost' for table 'choice'.`
Удаление таблицы Институтов `DROP TABLE Department;`	`ERROR 1142 (42000): DROP command denied to user 'coolView'@'localhost' for table 'Department'`	`ERROR 1142 (42000): DROP command denied to user 'viewsmotr'@'localhost' for table 'department'`
Вставка нового направления `INSERT INTO Specialty (department_id, code, name) VALUES (1, 'CS101', 'Computer Science');`	`Query OK, 1 row affected (0,00 sec)`	`ERROR 1142 (42000): INSERT command denied to user 'viewsmotr'@'localhost' for table 'specialty'`

Таб. 10. Реакции пользователей на запросы

2.4 Работа 4: Создание функции и процедуры

Задача: Реализовать функцию, принимающую на входе имя, фамилию, отчество абитуриента. Функция должна конкатинировать инициалы имени и отчества и полную фамилию. На выводе мы должны получить инициалы и фамилию. Так же рассмотреть случай при отсутствия отчества у абитуриента. Реализовать процедуру которая на входе получает все атрибуты таблицы института и таблицы направления, которая будет рассматривать все варианты добавления такого института / направления.

Формулировка задачи: Процедура должна добавлять новый институт при отсутствие института, должна добавлять новое направление при его отсутствие, должна добавлять и институт и направление при условии что такого института и направления не существует. Не добавлять существующие институты и направления. Функция должна выводить принимающие ФИО в формат: И.О. Фамилия.

Ниже приведен скрипт реализующий функцию giveIONS( ):

```
DELIMITER //
DROP FUNCTION IF exists giveIONS;
CREATE FUNCTION giveIONS(first_name VARCHAR(80), last_name VARCHAR(80),
middle_name VARCHAR(80))
RETURNS VARCHAR(100)
DETERMINISTIC
BEGIN
    RETURN IF(
        middle_name IS NULL OR middle_name = '',
        CONCAT(LEFT(first_name, 1), '. ', last_name),
        CONCAT(LEFT(first_name, 1), '.', LEFT(middle_name, 1), '. ',
last_name)
    );
END //

DELIMITER ;
```

Результат работы функции:

```
SELECT giveIONS(first_name, middle_name, last_name) AS ES
FROM Enrollee
WHERE enrollee_id = 199999;
```

```
+--------------------+
| ES                 |
+--------------------+
| M.S. Vladimirovich |
+--------------------+
```

Таб.11 Результат выполнения функции giveIONS()

Так же рассмотрен случай при отсутствии отчества у абитуриента:

```
SELECT giveIONS(first_name, middle_name, last_name) AS ES
FROM Enrollee
WHERE enrollee_id = 200001;
```

```
+------------------+
| ES               |
+------------------+
| A. Viktorovich   |
+------------------+
```

Таб.12 Результат giveIONS() при отсутствие отчества

Ошибка при передачи аргументов в giveIONS( ):

```
ERROR 1054 (42S22): Unknown column 'surname' in 'field list'
```

Ниже представлен скрипт реализующий процедуру AddIS( ):

```
DELIMITER //
DROP PROCEDURE IF exists AddIS;
CREATE PROCEDURE AddIS(
    IN p_department_name VARCHAR(80),
    IN p_phone_number VARCHAR(20),
    IN p_address TEXT,
    IN p_headmaster_name VARCHAR(250),
    IN p_specialty_code VARCHAR(50),
    IN p_specialty_name VARCHAR(50)
)
BEGIN
    DECLARE v_department_id INT;
    DECLARE v_specialty_id INT;
    SELECT department_id INTO v_department_id
    FROM Department
    WHERE name = p_department_name
    LIMIT 1;
    IF v_department_id IS NULL THEN
        INSERT INTO Department (name, phone_number, address, headmaster_name)
        VALUES (p_department_name, p_phone_number, p_address,
p_headmaster_name);
        SET v_department_id = LAST_INSERT_ID();
    END IF;
    SELECT specialty_id INTO v_specialty_id
    FROM Specialty
    WHERE code = p_specialty_code AND name = p_specialty_name
    LIMIT 1;
    IF v_specialty_id IS NULL THEN
        INSERT INTO Specialty (department_id, code, name)
        VALUES (v_department_id, p_specialty_code, p_specialty_name);
    ELSE
        UPDATE Specialty
        SET department_id = v_department_id
        WHERE specialty_id = v_specialty_id;
    END IF;
END //
DELIMITER ;
```

Процедура предназначена для добавления информации об институте и специальности в базу данных. Она принимает шесть параметров: название института, номер телефона, адрес, имя заведующего, код специальности и название специальности. Если институт с указанным именем не существует, процедура создаёт новую запись в таблице Department и получает его идентификатор. Затем она проверяет наличие специальности если она отсутствует, выполняется вставка в таблицу Specialty, и связывается с новым институтом. Если специальность уже существует, процедура обновляет ее, присваивая ей идентификатор института для связи между институтами и специальностями.

Результат выполнения процедуры:

Начальная таблица Department:

department_id	name	phone_number	address	headmaster_name
1	IKNK	7913705422	Vernosti 4	Salimli
2	GI	1414019764	Vernosti 5	Kapustin
3	PhysMech	4324874250	Novorossiyskaya 50	Vaganova
4	IBSiB	6152356738	Novorossiyskaya 50	Bakinets
5	IMPP	8929767323	Polytechnicheskaya 21	Vaganova
6	IMMiT	6766163777	Vernosti 4	Vaganova
7	IE	8928472624	Grazhdansky pr. 7	Kuznetsov
8	ISI	7932140390	Nepokorennykh 6	Grigoryev
9	YA	7565582073	Obruchevykh 1	Malikov
10	IFKiS	6498919645	Obruchevykh 1	Vaganova
11	IPMEiT	9319424990	Obruchevykh 1	Kapustin

Таб.13 Начальная таблица Department

Начальная таблица Specialty:

specialty_id	department_id	code	name
1	2	Code_1	MKN
2	9	Code_2	PI
3	2	Code_3	PMiF
4	6	Code_4	BF
5	9	Code_5	BI
6	1	Code_6	RiOS
7	2	Code_7	ZR
8	10	Code_8	MS
9	5	Code_9	YA
10	11	Code_10	Eco

Таб.14 Начальная таблица Specialty

1. Нет такого института и нет такого направления.

Вызов:

```
call addis(
'imop',
```

```
'8921948932',
'gjatskaya 9',
'alber einschtein',
'1121',
'international relations');
```

Результаты таблицы Department:

department_id	name	phone_number	address	headmaster_name
1	IKNK	7913705422	Vernosti 4	Salimli
2	GI	1414019764	Vernosti 5	Kapustin
3	PhysMech	4324874250	Novorossiyskaya 50	Vaganova
4	IBSiB	6152356738	Novorossiyskaya 50	Bakinets
5	IMPP	8929767323	Polytechnicheskaya 21	Vaganova
6	IMMiT	6766163777	Vernosti 4	Vaganova
7	IE	8928472624	Grazhdansky pr. 7	Kuznetsov
8	ISI	7932140390	Nepokorennykh 6	Grigoryev
9	YA	7565582073	Obruchevykh 1	Malikov
10	IFKiS	6498919645	Obruchevykh 1	Vaganova
11	IPMEiT	9319424990	Obruchevykh 1	Kapustin
21	imop	8921948932	gjatskaya 9	alber einschtein

Таб.15 Результат addIS(). Добавлен imop в таблицу Department.

Результат таблицы Specialty:

specialty_id	department_id	code	name
1	2	Code_1	MKN
2	9	Code_2	PI
3	2	Code_3	PMiF
4	6	Code_4	BF
5	9	Code_5	BI
6	1	Code_6	RiOS
7	2	Code_7	ZR
8	10	Code_8	MS
9	5	Code_9	YA
10	11	Code_10	Eco
17	21	1121	international relations

Таб.16 Результат addIS(). Добавлена international relations в таблицу Specialty.

2. Есть такой институт и нет такого направления.

Вызов: (GI - уже существующий институт в таб.13)

```
call addis(
'gi',
'89219456703',
'gjatskaya 4',
'albert moysha izyakov',
'1113',
'flugmaschine rekorder physic'
);
```

Результат таблицы Specialty:

specialty_id	department_id	code	name
1	2	Code_1	MKN
2	9	Code_2	PI
3	2	Code_3	PMiF
4	6	Code_4	BF
5	9	Code_5	BI
6	1	Code_6	RiOS
7	2	Code_7	ZR
8	10	Code_8	MS
9	5	Code_9	YA
10	11	Code_10	Eco
17	21	1121	international relations
18	2	1113	flugmaschine rekorder physic

Таб.17 Результат addIS(). Добавлена flugmaschine rekorder physic в таблицу Specialty.

Результат таблицы Department:

department_id	name	phone_number	address	headmaster_name
1	IKNK	7913705422	Vernosti 4	Salimli
2	GI	1414019764	Vernosti 5	Kapustin
3	PhysMech	4324874250	Novorossiyskaya 50	Vaganova
4	IBSiB	6152356738	Novorossiyskaya 50	Bakinets
5	IMPP	8929767323	Polytechnicheskaya 21	Vaganova
6	IMMiT	6766163777	Vernosti 4	Vaganova
7	IE	8928472624	Grazhdansky pr. 7	Kuznetsov
8	ISI	7932140390	Nepokorennykh 6	Grigoryev
9	YA	7565582073	Obruchevykh 1	Malikov
10	IFKiS	6498919645	Obruchevykh 1	Vaganova
11	IPMEiT	9319424990	Obruchevykh 1	Kapustin
21	imop	8921948932	gjatskaya 9	alber einschtein

Таб.18 Результат addIS(). В Department не добавлен лишний GI.

3. Нет такого института и есть такое направление.

Вызов: (flugmaschine rekorder physic - уже существующее направление в таб. 15)

```
call addis(
'aa',
'8921948999',
'gjatskaya 10',
'vladimir ilyich lenin',
'1113',
'flugmaschine rekorder physic'
);
```

Результат таблицы Department:

department_id	name	phone_number	address	headmaster_name
1	IKNK	7913705422	Vernosti 4	Salimli
2	GI	1414019764	Vernosti 5	Kapustin
3	PhysMech	4324874250	Novorossiyskaya 50	Vaganova
4	IBSiB	6152356738	Novorossiyskaya 50	Bakinets
5	IMPP	8929767323	Polytechnicheskaya 21	Vaganova
6	IMMiT	6766163777	Vernosti 4	Vaganova
7	IE	8928472624	Grazhdansky pr. 7	Kuznetsov
8	ISI	7932140390	Nepokorennykh 6	Grigoryev
9	YA	7565582073	Obruchevykh 1	Malikov
10	IFKiS	6498919645	Obruchevykh 1	Vaganova
11	IPMEiT	9319424990	Obruchevykh 1	Kapustin
21	imop	8921948932	gjatskaya 9	alber einschtein
22	aa	8921948999	gjatskaya 10	vladimir ilyich lenin

Таб.19 Результат addIS(). Добавлен aa в таблицу Department.

Результат таблицы Specialty:

specialty_id	department_id	code	name
1	2	Code_1	MKN
2	9	Code_2	PI
3	2	Code_3	PMiF
4	6	Code_4	BF
5	9	Code_5	BI
6	1	Code_6	RiOS
7	2	Code_7	ZR
8	10	Code_8	MS
9	5	Code_9	YA
10	11	Code_10	Eco
17	21	1121	international relations
18	2	1113	flugmaschine rekorder physic

Таб.20 Результат addIS(). В таблицу Specialty не добавлено лишнее flugmaschine rekorder physic.

4. Есть такой институт и есть такое направление.

Вызов: (aa - институт, flugmaschine rekorder physic - направление, уже существуют в таблицах Department, Specialty соответственно)

```
call addis(
'aa',
'8921948999',
'gjatskaya 10',
'vladimir ilyich lenin',
'1113',
'flugmaschine rekorder physic'
);
Query OK, 0 rows affected (0,00 sec) -- ничего не изменилось
```

Результат таблицы Department:

department_id	name	phone_number	address	headmaster_name
1	IKNK	7913705422	Vernosti 4	Salimli
2	GI	1414019764	Vernosti 5	Kapustin
3	PhysMech	4324874250	Novorossiyskaya 50	Vaganova
4	IBSiB	6152356738	Novorossiyskaya 50	Bakinets
5	IMPP	8929767323	Polytechnicheskaya 21	Vaganova
6	IMMiT	6766163777	Vernosti 4	Vaganova
7	IE	8928472624	Grazhdansky pr. 7	Kuznetsov
8	ISI	7932140390	Nepokorennykh 6	Grigoryev
9	YA	7565582073	Obruchevykh 1	Malikov
10	IFKiS	6498919645	Obruchevykh 1	Vaganova
11	IPMEiT	9319424990	Obruchevykh 1	Kapustin
21	imop	8921948932	gjatskaya 9	alber einschtein
22	aa	8921948999	gjatskaya 10	vladimir ilyich lenin

Таб.21 Результат addIS(). Ничего не изменилось.

Результат таблицы Specialty:

specialty_id	department_id	code	name
1	2	Code_1	MKN
2	9	Code_2	PI
3	2	Code_3	PMiF
4	6	Code_4	BF
5	9	Code_5	BI
6	1	Code_6	RiOS
7	2	Code_7	ZR
8	10	Code_8	MS
9	5	Code_9	YA
10	11	Code_10	Eco
17	21	1121	international relations
18	2	1113	flugmaschine rekorder physic

Таб.22 Результат addIS(). Ничего не изменилось.

Все функции и процедуры хранятся в информационной схеме, в ROUTINES.
Что бы просмотреть нашу функцию и процедуру нужно выполнить запрос:

```
select routine_name from information_schema.routines
```

Результат:

```
...
| giveIONS                            |
| AddIS                               |
+-------------------------------------+
50 rows in set (0,00 sec)
```

Таб.23 Список функций и процедур.

Наша функция и процедура находятся в этом списке. Если мы захотим удалить новые направления и их институты, начинать придется с таблицы Specialty, так как внешний ключ соединяет таблицу Specialty и Department, как видно из Рис.1. (Приложение А).

2.5 Работа 5: Транзакционная модель

Задача: Задать уровень изоляции транзакции на выбор.

Формулировка задачи: Необходимо проверить наличие феномена «неустойчивого чтения» при работе с таблицами с уровнем изоляции read committed. Уровень изоляции - запрещает другим транзакциям изменять строки, которые были считаны незавершенной транзакцией. Однако другие транзакции могут вставлять новые строки, содержащихся в текущей транзакции. При повторном запуске инструкции текущей транзакции будут извлечены новые строки, что приведет к «неустойчивому чтению». По итогу, использование read committed, действительно не разрешает «неустойчивое чтение».

T	Транзакция 1. User1	Транзакция 2. User2
	Комиссия хочет узнать, средний балл результатов ЕГЭ, который сдавали абитуриенты. User1 просматривает средний балл, который дал результат п. После поданной апелляции, User2 повышает балл одному из абитуриентов. После повторного просмотра, user1 видит другой средний балл. Возникает артефакт "неустойчивое чтение, (non-repeatable read)".	
T_1	SET TRANSACTION ISOLATION LEVEL READ COMMITTED;	
T_2	START TRANSACTION;	START TRANSACTION;
T_3	SELECT AVG('ege_result') AS avg_ege_score FROM 'Achievement'; User1 просматривает средний результат ЕГЭ абитуриентов. +----------------+ | avg_ege_score | +----------------+ | 50.5210 | +----------------+	
T_4		UPDATE `Achievement` set`ege_result` = 95 WHERE `enrollee_id` = 199999; COMMIT; User2 Изменяет баллы полученные абитуриентом 199999 +----------+----------+----------+----------+----------+ | achievment_id | enrollee_id | subject_id | ege_result | inner_result | +----------+----------+----------+----------+----------+ | 800252 | 199998 | 14 | 95 | 58 | +----------+----------+----------+----------+----------+
T_5	SELECT AVG('ege_result') AS avg_ege_score FROM 'Achievement'; User1 Повторно просматривает средний результат ЕГЭ и получает неустойчивое чтение. +----------------+ | avg_ege_score | +----------------+ | 50.5212 | +----------------+	

Таб.24 Проведение транзакций

Заключение

В ходе прохождения данного курса было выполнено пять лабораторных работ.

1. Создано представление с запросом, подсчитывающее количество пройденных испытаний абитуриентом и количество поданных заявлений абитуриентом в конкретный институт.
2. Написаны триггеры, автоматизирующие информацию о количестве абитуриентов подавших документы в конкретный институт.
3. Созданы два пользователя. Первый имеет доступ только на просмотр представления (из пункта 1). Второй имеет права на редактирование таблиц использующиеся в запросе представления
4. Реализована функция, конкатинирующая инициалы имени, отчества, а так же полную фамилию абитуриента. Так же реализована процедура, добавляющее институт и направление связанное с этим институтом.
5. В последней работе проведено исследование уровня изоляции транзакции read committed и того, защищает ли этот уровень от неустойчивого чтения. Полученный результат полностью совпадает с теоретической информацией. Уровень read committed не может защитить от неустойчивого чтения-(non-repeatable read).

Приложение А

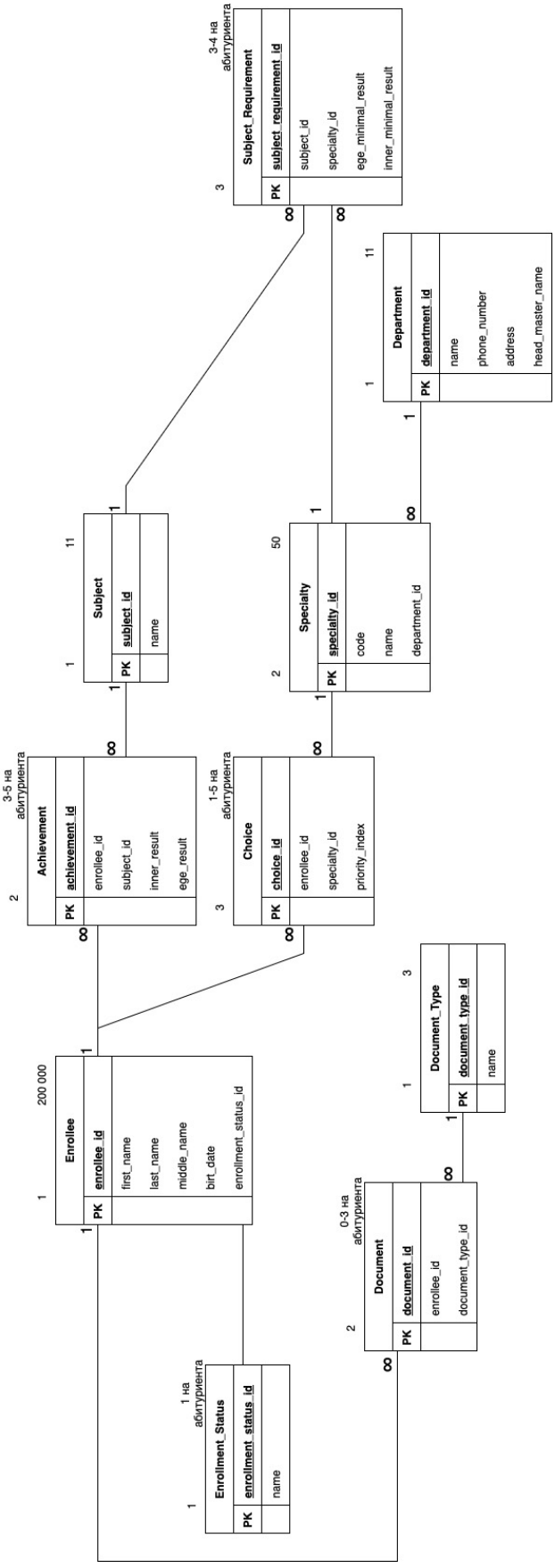

Рис. 1 Схема базы данных.