

ГУАП

КАФЕДРА № 42

ОТЧЕТ  
ЗАЩИЩЕН С ОЦЕНКОЙ \_\_\_\_\_

ПРЕПОДАВАТЕЛЬ

Доцент				Бржезовский А. В.
должность, уч. степень, звание		подпись, дата		инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №5

**ЗАПРОСЫ НА ЯЗЫКЕ SQL: ПОДЗАПРОСЫ**

Вариант 5

по курсу: Методы и средства проектирования информационных систем  
и технологий

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. №	4128			Воробьев В.А.
			подпись, дата	инициалы, фамилия

## СОДЕРЖАНИЕ

<b>1</b>	<b>Постановка задачи</b>	<b>3</b>
1.1	Цель работы	3
1.2	Задание	3
1.3	Содержание отчета	4
<b>2</b>	<b>Выполнение работы</b>	<b>5</b>
2.1	Задание 7	5
2.2	Задание 8	6
2.3	Задание 9	7
2.4	Разница запросов	8
<b>3</b>	<b>Вывод</b>	<b>10</b>
	<b>ПРИЛОЖЕНИЕ</b>	<b>11</b>

## 1 Постановка задачи

### 1.1 Цель работы

Освоить подзапросы, экзистенциальные запросы, производные таблицы и представления, а также директивы `create view`, `drop view`, `exists`.

### 1.2 Задание

По аналогии с примерами, приведенными выше:

- реализовать запросы ж) .. и), указанные в варианте задания;
- самостоятельно предложить и реализовать запросы, демонстрирующие использование подзапросов в операторах манипулирования данными;
- с помощью `[not] exists` реализовать запросы, разработанные в п. 4 для иллюстрации использования теоретико-множественных операций, показать различие в выполнении запросов с `[not] exists` и теоретико-множественными операциями при наличии в таблицах `null`-значений (п. 5.3).

### 5 Вариант:

Создайте базу данных для хранения следующих сведений: ВУЗ, студент, группа, факультет, конференция, тема доклада, программа конференции. Составьте запросы, позволяющие выбрать:

- а) студентов первого факультета, выступавших на конференции Информатика;
- б) темы докладов студентов для заданной группы;
- в) выступления, подготовленные двумя студентами различных факультетов;
- г) количество докладов для каждой конференции;
- д) среднее количество докладов, сделанных студентами третьего факультета на конференциях;
- е) студентов, выступивших на трех или большем числе конференций;
- ж) студентов четвертого факультета, не выступавших на конференциях;**
- з) студентов, выступивших на всех конференциях;
- и) пары студентов, всегда выступающие вместе.**

### **1.3 Содержание отчета**

- текст запросов на SQL (с пояснениями/комментариями);
- наборы данных, возвращаемые запросами.

## 2 Выполнение работы

Исходные данные взяты из лабораторной работы №2, отчет для которой есть на GitHub (URI - [https://github.com/vladcto/suai-labs/blob/d8c7a508971967641d8638ebcd107539c8fd618e/6\\_semester/%D0%9C%D0%A1%D0%9F%D0%98%D0%A1%D0%A2/%D0%BC%D1%81%D0%B8%D0%BF%D0%B8%D1%81%D1%82\\_2.pdf](https://github.com/vladcto/suai-labs/blob/d8c7a508971967641d8638ebcd107539c8fd618e/6_semester/%D0%9C%D0%A1%D0%9F%D0%98%D0%A1%D0%A2/%D0%BC%D1%81%D0%B8%D0%BF%D0%B8%D1%81%D1%82_2.pdf)).

Исходный код доступен в приложении и репозитории GitHub (URI - WIP).

### 2.1 Задание 7

Запрос для задания 1 позволяет получить список имен студентов первого факультета, принимавших участие в конференции “Информатика”. Для этого используется несколько JOIN операторов для объединения таблиц и условия WHERE для фильтрации результатов по номеру факультета и названию конференции. Ключевое слово DISTINCT используется для вывода уникальных имен студентов.

#### Листинг 7 задания:

```
1  -- студентов четвертого факультета, не выступавших на
   конференциях
2  USE conference_db_lab1;
3
4  SELECT s.name
5         FROM student s
6              JOIN uni_group g ON s.group_id = g.id
7              JOIN faculty f ON g.faculty_id = f.id
8  WHERE f.number = 4
9         AND NOT EXISTS (SELECT 1
10                          FROM authorship a
11                          WHERE a.author_id = s.id);
```

	name
1	Клон 1

Рисунок 2.1 - Результат 1.sql

## 2.2 Задание 8

Запрос 2.sql возвращает уникальные темы докладов студентов группы с указанным именем. Используются JOIN операторы для объединения таблиц и условие WHERE для фильтрации результатов по имени группы. Ключевое слово DISTINCT применяется для вывода уникальных тем докладов.

### Листинг 8 задания:

```

1  -- студентов, выступивших на всех конференциях;
2  USE conference_db_lab1;
3
4  SELECT s.id AS student_id,
5         s.name AS student_name
6  FROM student s
7  WHERE NOT EXISTS (
8      SELECT 1 FROM conference c
9      WHERE NOT EXISTS (
10         SELECT 1 FROM authorship a
11         JOIN topic t ON a.topic_id = t.id
12         JOIN conference_session cs ON t.session_id = cs.id
13         WHERE a.author_id = s.id AND cs.conference_id = c.id
14     )
15 );

```

	topic
1	Topic 1
2	Topic 2
3	Topic 4
4	Topic 6
5	Topic 8

Рисунок 2.2 - Результат 2.sql

### 2.3 Задание 9

В запросе для 3 задания осуществляется поиск выступлений, подготовленных двумя студентами с различных факультетов. С использованием операторов JOIN объединяются таблицы, представляющие информацию о темах докладов, авторах, студентах, группах и факультетах. Условия WHERE фильтруют результаты, чтобы выбрать только те темы, для которых студенты принадлежат разным факультетам. Результат запроса включает название темы, имя первого и второго студента. Результаты сортируются в алфавитном порядке по имени первого студента.

#### Листинг 9 задания:

```

1  -- пары студентов , всегда выступающие вместе .
2  USE conference_db_lab1 ;
3
4  SELECT a1.author_id AS student1 , a2.author_id AS student2
5     FROM authorship a1
6          JOIN authorship a2 ON a1.topic_id = a2.topic_id AND
7                               a1.author_id < a2.author_id
8  WHERE NOT EXISTS (SELECT 1

```

```

8          FROM authorship
9          WHERE author_id = a2.author_id
10             AND topic_id != a1.topic_id)
11 AND NOT EXISTS (SELECT 1
12                  FROM authorship
13                  WHERE author_id = a1.author_id
14                     AND topic_id != a2.topic_id)
15 GROUP BY a1.author_id , a2.author_id;

```

	topic	student1	student2
1	Topic 1	Клон 1	Клон 2

Рисунок 2.3 - Результат 3.sql

## 2.4 Разница запросов

В ниже представленных SQL-запросах представлены ранее не использованные команды. Первый запрос демонстрирует использование оператора BETWEEN для выбора тем докладов с идентификаторами от 1 до 3.

Второй запрос использует оператор IS NOT NULL для извлечения идентификаторов групп, у которых имя не равно NULL. Третий запрос иллюстрирует использование оператора LIKE с символом %, что позволяет выбрать все идентификаторы групп, где имя содержит любые символы.

### Листинг diff.sql:

```

1  USE conference_db_lab1;
2
3  DROP TEMPORARY TABLE IF EXISTS temp_students;
4  CREATE TEMPORARY TABLE temp_students AS
5  SELECT s.id , s.name
6     FROM student s
7     WHERE s.id <= 5
8  UNION
9  SELECT NULL, 'Аноним';
10
11 SELECT * FROM temp_students;
12
13 SELECT s.id , s.name
14     FROM student s
15     WHERE NOT EXISTS (SELECT NULL FROM temp_students ts WHERE
16                       ts.id = s.id);

```



```
16
17 SELECT s.id, s.name
18     FROM student s
19     WHERE s.id NOT IN (SELECT ts.id FROM temp_students ts);
```

### **3 Вывод**

В результате выполнения лабораторной работы были получены навыки работы с SQL-запросами.

Каждый запрос был разработан с учетом поставленных задач, а также внедрены самостоятельно предложенные запросы, демонстрирующие использование различных директив SQL. В процессе выполнения работы были охвачены такие аспекты, как фильтрация данных, сортировка результатов, объединение таблиц и использование различных условий для точного извлечения необходимой информации из базы данных.

Полученные знания и навыки будут полезны в будущих проектах и задачах, связанных с обработкой данных в среде SQL.

## ПРИЛОЖЕНИЕ

### Листинг 7.sql задания:

```
1  -- студентов четвертого факультета, не выступавших на
   конференциях
2  USE conference_db_lab1;
3
4  SELECT s.name
5         FROM student s
6             JOIN uni_group g ON s.group_id = g.id
7             JOIN faculty f ON g.faculty_id = f.id
8         WHERE f.number = 4
9             AND NOT EXISTS (SELECT 1
10                             FROM authorship a
11                             WHERE a.author_id = s.id);
```

### Листинг 8.sql задания:

```
1  -- студентов, выступивших на всех конференциях;
2  USE conference_db_lab1;
3
4  SELECT s.id AS student_id,
5         s.name AS student_name
6  FROM student s
7  WHERE NOT EXISTS (
8      SELECT 1 FROM conference c
9      WHERE NOT EXISTS (
10         SELECT 1 FROM authorship a
11         JOIN topic t ON a.topic_id = t.id
12         JOIN conference_session cs ON t.session_id = cs.id
13         WHERE a.author_id = s.id AND cs.conference_id = c.id
14     )
15 );
```

### Листинг 9.sql задания:

```
1  -- пары студентов, всегда выступающие вместе.
2  USE conference_db_lab1;
3
4  SELECT a1.author_id AS student1, a2.author_id AS student2
5         FROM authorship a1
6             JOIN authorship a2 ON a1.topic_id = a2.topic_id AND
7                                 a1.author_id < a2.author_id
8         WHERE NOT EXISTS (SELECT 1
```

```

8          FROM authorship
9          WHERE author_id = a2.author_id
10             AND topic_id != a1.topic_id)
11     AND NOT EXISTS (SELECT 1
12                     FROM authorship
13                     WHERE author_id = a1.author_id
14                        AND topic_id != a2.topic_id)
15     GROUP BY a1.author_id , a2.author_id;

```

### **Листинг diff.sql:**

```

1  USE conference_db_lab1;
2
3  DROP TEMPORARY TABLE IF EXISTS temp_students;
4  CREATE TEMPORARY TABLE temp_students AS
5  SELECT s.id , s.name
6     FROM student s
7     WHERE s.id <= 5
8  UNION
9  SELECT NULL, 'Аноним';
10
11 SELECT * FROM temp_students;
12
13 SELECT s.id , s.name
14     FROM student s
15     WHERE NOT EXISTS (SELECT NULL FROM temp_students ts WHERE
16                       ts.id = s.id);
16
17 SELECT s.id , s.name
18     FROM student s
19     WHERE s.id NOT IN (SELECT ts.id FROM temp_students ts);

```