

**Лабораторные работы по курсу
Базы данных**

**Лабораторная работа 3
«Использование объединяющих и вложенных запросов языка SQL»**

Москва, 2023

Оглавление

1.	Теоретическая часть.....	3
1.1.	Соединение таблиц.....	3
1.1.1.	Неявное соединение таблиц.....	3
1.1.2.	Соединение с помощью JOIN.....	4
1.1.3.	Объединение, разность, пересечение таблиц.....	5
1.1.4.	Пример соединения и объединения таблиц.....	6
1.2.	Подзапросы.....	8
1.2.1.	Некоррелированный запрос.....	8
1.2.2.	Коррелированный запросы.....	9
2.	Практическая часть.....	10
2.1.	Задание 1.....	10
2.2.	Задание 2.....	10
2.3.	Задание 3.....	10
2.4.	Задание 4.....	10
2.5.	Задание 5.....	10
	Список литературы.....	10

1. Теоретическая часть

В предыдущей лабораторной работе были рассмотрены примеры запросов на выборку данных из одной таблицы. Однако, существует большое число задач, когда требуется проанализировать информацию из нескольких таблиц. Для этого существуют операции соединения.

1.1. Соединение таблиц

1.1.1. Неявное соединение таблиц

Самым простым способом является неявное соединение таблиц, когда таблицы объединяются перекрестно. Другими словами, каждой строке одной таблицы будет совмещаться с каждой строкой второй таблицы. В данном случае мы получаем прямое (декартово) произведение двух таблиц.

На языке SQL для неявного соединения необходимо указать требуемые таблицы через запятую после оператора FROM.

Предположим, что нам необходимо вывести ФИО всех преподавателей и полное название подразделения, в котором они трудоустроены. Для этого соединим таблицы Структурное подразделение и Трудоустройство. Для этого возможно выполнить следующий запрос:

```
SELECT abbreviated_title, professor_id FROM structural_unit, employment;
```

abbreviated_title	professor_id
Институт МПСУ	81001
Институт МПСУ	81002
Институт МПСУ	81003
Институт МПСУ	81004
Институт МПСУ	81005
Институт МПСУ	81006
Институт МПСУ	81007
Институт МПСУ	81008

В результате запроса будет выведены значения из двух таблиц, где каждому значению из второй будут соответствовать все значения из первой таблицы. Таким образом, будет выведено (число строк в первой таблице) * (число строк во второй таблице). В данном примере будет выведено 24 строки.

Аналогично добавим к данным двум таблицам ещё таблицу professor.

```
SELECT abbreviated_title, professor_id, surname FROM structural_unit, employment, professor;
```

abbreviated_title	surname	wage_rate
Институт МПСУ	Широков	0.25
Институт СПИНТех	Широков	0.25
	Широков	0.25
ВМ-1	Широков	0.25
	Широков	0.25
ЛПО	Широков	0.25
ПМТ	Широков	0.25
ЦД	Широков	0.25

Очевидно, что такой результат запроса нас не устроит. Для того, чтобы отобразить только преподавателей, прикрепленных к месту работы, необходимо добавить условие отбора.

Выберем только те строки, где значения номера подразделения в таблицах structural_unit и employment и значения номера преподавателя в таблицах employment и professor совпадают.

Итоговый запрос будет выглядеть следующим образом:

```
SELECT professor.surname, professor.name,
professor.patronymic ,structural_unit.abbreviated_title
FROM structural_unit, employment, professor
WHERE structural_unit.structural_unit_id = employment.structural_unit_number
AND employment.professor_id = professor.professor_id
```

surname	name	patronymic	abbreviated_title
Широков	Василий	Иванович	Институт МПСУ
Семенов	Андрей	Сергеевич	Институт МПСУ
Филатов	Илья	Иванович	Институт МПСУ
Ермаков	Виктор	Денисович	Институт МПСУ
Марков	Виктор	Дмитриевич	Институт МПСУ
Воронов	Николай	Михайлович	Институт МПСУ
Хахалин	Василий	Тихонович	Институт МПСУ
Козин	Николай	Дмитриевич	Институт МПСУ

Для того, чтобы обращаться к атрибуту конкретной таблицы, необходимо указывать название таблицы, отделив его от названия атрибута точкой.

Обратим внимание на то, что используя полное имя атрибута (название_таблицы.имя_атрибута) сильно удлинняется код запроса. Для сокращения возможно использовать более короткую запись

1.1.2. Соединение с помощью JOIN

Соединённая таблица — это таблица, полученная из двух других таблиц в соответствии с правилами соединения. Общий синтаксис описания соединённой таблицы:

T1 тип_соединения T2 [условие_соединения]

Кроме неявного соединения таблиц в языке SQL существует альтернативная форма записи операций соединения таблиц с помощью ключевого слова JOIN. Объединение происходит по столбцу, который есть в каждой из таблиц.

По типу соединения операторы JOIN подразделяются на внутренние и внешние — INNER JOIN и OUTER JOIN.

Внутреннее соединение - INNER JOIN используется для отбора строк из двух таблиц, в которых совпадают значения поля, по которому происходит объединение.

Например, объединим таблицы «Студент» и «Результат освоения дисциплины». Каждая из этих таблиц содержит поле студенческого билета. Произведем по этому полю соединение.

```
SELECT surname, name, mark
FROM Student
INNER JOIN Field_comprehension
ON Field_comprehension.student_id = Student.student_id;
```

В результате выполнения запроса на экран будет выведена информация об оценках всех студентов института.



Внешнее соединение - OUTER JOIN можно разделить на правое (RIGHT), левое (LEFT) и полное (FULL).

Операция левого внешнего соединения возвращает кроме обычного результата соединения, строки из левого операнда, для которых не нашлось парного значения строки в правом операнде. Вместо значений атрибутов правого операнда будет указано неопределенное значение NULL.

Аналогично, операция правого внешнего соединения возвращает все строки, для которых не нашлось пары в первом операнде.

Полное внешнее соединение включает в себя все пересекающиеся строки и все непарные строки из обеих таблиц.

Аналогично неявному соединению, возможно произвести декартово произведение таблиц с помощью оператора JOIN. Для этого существует ключевое слово CROSS.

С помощью операций соединения возможно соединять несколько таблиц. Рассмотрим это на примере более сложного запроса. Необходимо вывести всех преподавателей и структурное подразделение, к которым они прикреплены.

Данные о преподавателях хранятся в таблице professor. Данные об структурных подразделениях – в таблице structural_unit. Общих столбцов у данных таблиц нет, поэтому необходимо соединить их вместе с промежуточной таблицей – employment.

```
SELECT professor.surname, professor.name, professor.patronymic,
structural_unit.full_title
FROM professor
INNER JOIN employment ON
employment.professor_id = professor.professor_id
INNER JOIN structural_unit ON
structural_unit.structural_unit_id = employment.structural_unit_number
```

surname	name	patronymic	full_title
Александрова	Александра	Александровна	Кафедра высшей математики № 1
Аркадина	Ирина	Николаевна	Кафедра высшей математики № 1
Баранова	Виктория		Кафедра высшей математики № 1
Березина	Наталья	Владимировна	Институт высокотехнологичного права, социальных и гуманитарных наук
Быкова	Елена	Николаевна	Кафедра высшей математики № 1
Вернадский	Владимир	Иванович	Институт цифрового дизайна
Витгенштейн	Людвиг		Институт лингвистического и педагогического образования
Воронов	Николай	Михайлович	Институт микроприборов и систем управления имени Л.Н. Преснухина

1.1.3. Объединение, разность, пересечение таблиц

Кроме соединения таблиц, когда в результате операции атрибуты одной таблицы будут добавлены к атрибутам другой существуют операции **объединения**. В данном случае число атрибутов не изменяется, но в итоговой таблице будут содержаться значения из нескольких таблиц. При объединении таблиц необходимо соблюдать условие, что тип данных каждого столбца первой таблицы должен совпадать с типом данных соответствующего столбца во второй таблице. Имена столбцов в объединяемых таблицах не обязательно должны быть одинаковыми.

В языке SQL для объединения таблиц используется оператор UNION.

запрос1 UNION [ALL] запрос2

Т.к. оператор UNION объединяет строки из двух таблиц результатов, в объединенной таблице могут содержаться повторяющиеся строки. По умолчанию оператор UNION в процессе своего выполнения удаляет повторяющиеся строки.

Кроме операции объединения существуют операции **пересечения** (INTERSECT) и **разности** (EXCEPT). Операция пересечения оставляет только общие строки из двух таблиц, а операция разности удаляет из первой таблицы значения, содержащиеся во второй. Данные три операции в языке SQL тесно связаны с логическими операциями булевой алгебры.

Операция	SQL операция	Логическая операция
Объединение	UNION	A OR B
Пересечение	INTERSECT	A AND B
Разность	EXCEPT	A AND NOT B

1.1.4. Пример соединения и объединения таблиц

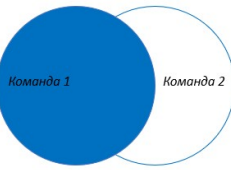

Для закрепления материала рассмотрим еще один пример. Предположим, что имеются две таблицы, содержащие игроков двух команд. Два игрока входят в составы обеих из них. Применим к ним операции соединения.

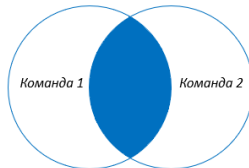
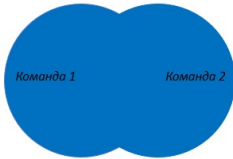
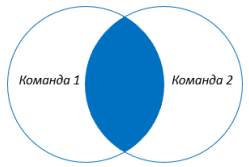

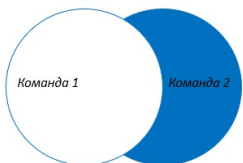
Команда1

ID	Фамилия
1	Герасимов
2	Полякова
3	Емельянов
4	Миронова

Команда2

ID	Фамилия
1	Лазарева
2	Емельянов
3	Новиков
4	Полякова

SQL запрос	Возвращенные значения	Комментарий																																																			
<pre>SELECT K1.Фамилия, K2.Фамилия FROM K1, K2; Или SELECT K1.Фамилия, K2.Фамилия FROM K1 CROSS JOIN K2</pre>	<table><thead><tr><th></th><th>Фамилия character varying (64)</th><th>Фамилия character varying (64)</th></tr></thead><tbody><tr><td>1</td><td>Герасимов</td><td>Лазарева</td></tr><tr><td>2</td><td>Герасимов</td><td>Емельянов</td></tr><tr><td>3</td><td>Герасимов</td><td>Новиков</td></tr><tr><td>4</td><td>Герасимов</td><td>Полякова</td></tr><tr><td>5</td><td>Полякова</td><td>Лазарева</td></tr><tr><td>6</td><td>Полякова</td><td>Емельянов</td></tr><tr><td>7</td><td>Полякова</td><td>Новиков</td></tr><tr><td>8</td><td>Полякова</td><td>Полякова</td></tr><tr><td>9</td><td>Емельянов</td><td>Лазарева</td></tr><tr><td>10</td><td>Емельянов</td><td>Емельянов</td></tr><tr><td>11</td><td>Емельянов</td><td>Новиков</td></tr><tr><td>12</td><td>Емельянов</td><td>Полякова</td></tr><tr><td>13</td><td>Миронова</td><td>Лазарева</td></tr><tr><td>14</td><td>Миронова</td><td>Емельянов</td></tr><tr><td>15</td><td>Миронова</td><td>Новиков</td></tr><tr><td>16</td><td>Миронова</td><td>Полякова</td></tr></tbody></table>		Фамилия character varying (64)	Фамилия character varying (64)	1	Герасимов	Лазарева	2	Герасимов	Емельянов	3	Герасимов	Новиков	4	Герасимов	Полякова	5	Полякова	Лазарева	6	Полякова	Емельянов	7	Полякова	Новиков	8	Полякова	Полякова	9	Емельянов	Лазарева	10	Емельянов	Емельянов	11	Емельянов	Новиков	12	Емельянов	Полякова	13	Миронова	Лазарева	14	Миронова	Емельянов	15	Миронова	Новиков	16	Миронова	Полякова	Декартово произведение двух таблиц. Каждой игроку из первой команды сопоставляется каждый игрок из второй команды.
	Фамилия character varying (64)	Фамилия character varying (64)																																																			
1	Герасимов	Лазарева																																																			
2	Герасимов	Емельянов																																																			
3	Герасимов	Новиков																																																			
4	Герасимов	Полякова																																																			
5	Полякова	Лазарева																																																			
6	Полякова	Емельянов																																																			
7	Полякова	Новиков																																																			
8	Полякова	Полякова																																																			
9	Емельянов	Лазарева																																																			
10	Емельянов	Емельянов																																																			
11	Емельянов	Новиков																																																			
12	Емельянов	Полякова																																																			
13	Миронова	Лазарева																																																			
14	Миронова	Емельянов																																																			
15	Миронова	Новиков																																																			
16	Миронова	Полякова																																																			
<pre>SELECT K1.Фамилия, K2.Фамилия FROM K1 FULL OUTER JOIN K2 ON K1.Фамилия = K2.Фамилия</pre>	<table><thead><tr><th></th><th>Фамилия character varying (64)</th><th>Фамилия character varying (64)</th></tr></thead><tbody><tr><td>1</td><td>Герасимов</td><td>[null]</td></tr><tr><td>2</td><td>Полякова</td><td>Полякова</td></tr><tr><td>3</td><td>Емельянов</td><td>Емельянов</td></tr><tr><td>4</td><td>Миронова</td><td>[null]</td></tr><tr><td>5</td><td>[null]</td><td>Новиков</td></tr><tr><td>6</td><td>[null]</td><td>Лазарева</td></tr></tbody></table>		Фамилия character varying (64)	Фамилия character varying (64)	1	Герасимов	[null]	2	Полякова	Полякова	3	Емельянов	Емельянов	4	Миронова	[null]	5	[null]	Новиков	6	[null]	Лазарева	Отбор всех игроков из двух команд с возможностью совпадений записей в левой и правой таблицах. Если таковых нет, на пустой стороне вставляется NULL.																														
	Фамилия character varying (64)	Фамилия character varying (64)																																																			
1	Герасимов	[null]																																																			
2	Полякова	Полякова																																																			
3	Емельянов	Емельянов																																																			
4	Миронова	[null]																																																			
5	[null]	Новиков																																																			
6	[null]	Лазарева																																																			
<pre>SELECT K1.Фамилия, K2.Фамилия FROM K1 LEFT OUTER JOIN K2 ON K1.Фамилия = K2.Фамилия</pre> 	<table><thead><tr><th></th><th>Фамилия character varying (64)</th><th>Фамилия character varying (64)</th></tr></thead><tbody><tr><td>1</td><td>Герасимов</td><td>[null]</td></tr><tr><td>2</td><td>Полякова</td><td>Полякова</td></tr><tr><td>3</td><td>Емельянов</td><td>Емельянов</td></tr><tr><td>4</td><td>Миронова</td><td>[null]</td></tr></tbody></table>		Фамилия character varying (64)	Фамилия character varying (64)	1	Герасимов	[null]	2	Полякова	Полякова	3	Емельянов	Емельянов	4	Миронова	[null]	Отбираются все игроки из первой команды и игроки из второй, которые также играют за первую команду.																																				
	Фамилия character varying (64)	Фамилия character varying (64)																																																			
1	Герасимов	[null]																																																			
2	Полякова	Полякова																																																			
3	Емельянов	Емельянов																																																			
4	Миронова	[null]																																																			
<pre>SELECT K1.Фамилия, K2.Фамилия FROM K1 RIGHT OUTER JOIN K2 ON K1.Фамилия = K2.Фамилия</pre> 	<table><thead><tr><th></th><th>Фамилия character varying (64)</th><th>Фамилия character varying (64)</th></tr></thead><tbody><tr><td>1</td><td>[null]</td><td>Лазарева</td></tr><tr><td>2</td><td>Емельянов</td><td>Емельянов</td></tr><tr><td>3</td><td>[null]</td><td>Новиков</td></tr><tr><td>4</td><td>Полякова</td><td>Полякова</td></tr></tbody></table>		Фамилия character varying (64)	Фамилия character varying (64)	1	[null]	Лазарева	2	Емельянов	Емельянов	3	[null]	Новиков	4	Полякова	Полякова	Отбираются все игроки из второй команды и игроки из первой, которые также играют за вторую команду.																																				
	Фамилия character varying (64)	Фамилия character varying (64)																																																			
1	[null]	Лазарева																																																			
2	Емельянов	Емельянов																																																			
3	[null]	Новиков																																																			
4	Полякова	Полякова																																																			

<div>SELECT K1.Фамилия, K2.Фамилия FROM K1 INNER JOIN K2 ON K1.Фамилия = K2.Фамилия</div> <div></div>	<table><thead><tr><th></th><th>Фамилия character varying (64)</th><th>Фамилия character varying (64)</th></tr></thead><tbody><tr><td>1</td><td>Полякова</td><td>Полякова</td></tr><tr><td>2</td><td>Емельянов</td><td>Емельянов</td></tr></tbody></table>		Фамилия character varying (64)	Фамилия character varying (64)	1	Полякова	Полякова	2	Емельянов	Емельянов	Отбираются участники, играющие за обе команды					
	Фамилия character varying (64)	Фамилия character varying (64)														
1	Полякова	Полякова														
2	Емельянов	Емельянов														
<div>SELECT Фамилия FROM K1 UNION SELECT Фамилия FROM K2</div> <div></div>	<table><thead><tr><th></th><th>Фамилия character varying (64)</th></tr></thead><tbody><tr><td>1</td><td>Лазарева</td></tr><tr><td>2</td><td>Герасимов</td></tr><tr><td>3</td><td>Полякова</td></tr><tr><td>4</td><td>Миронова</td></tr><tr><td>5</td><td>Новиков</td></tr><tr><td>6</td><td>Емельянов</td></tr></tbody></table>		Фамилия character varying (64)	1	Лазарева	2	Герасимов	3	Полякова	4	Миронова	5	Новиков	6	Емельянов	Объединение всех участников команд без повторяющихся значений.
	Фамилия character varying (64)															
1	Лазарева															
2	Герасимов															
3	Полякова															
4	Миронова															
5	Новиков															
6	Емельянов															
<div>SELECT Фамилия FROM K1 INTERSECT SELECT Фамилия FROM K2</div> <div></div>	<table><thead><tr><th></th><th>Фамилия character varying (64)</th></tr></thead><tbody><tr><td>1</td><td>Полякова</td></tr><tr><td>2</td><td>Емельянов</td></tr></tbody></table>		Фамилия character varying (64)	1	Полякова	2	Емельянов	Объединение всех участников команд.								
	Фамилия character varying (64)															
1	Полякова															
2	Емельянов															
<div>SELECT Фамилия FROM K1 EXCEPT SELECT Фамилия FROM K2</div> <div></div>	<table><thead><tr><th></th><th>Фамилия character varying (64)</th></tr></thead><tbody><tr><td>1</td><td>Герасимов</td></tr><tr><td>2</td><td>Миронова</td></tr></tbody></table>		Фамилия character varying (64)	1	Герасимов	2	Миронова	Выбор игроков, играющих за первую команду, но не играющих за вторую.								
	Фамилия character varying (64)															
1	Герасимов															
2	Миронова															
<div>SELECT Фамилия FROM K2 EXCEPT SELECT Фамилия FROM K1</div> <div></div>	<table><thead><tr><th></th><th>Фамилия character varying (64)</th></tr></thead><tbody><tr><td>1</td><td>Лазарева</td></tr><tr><td>2</td><td>Новиков</td></tr></tbody></table>		Фамилия character varying (64)	1	Лазарева	2	Новиков	Выбор игроков, играющих за вторую команду, но не играющих за первую.								
	Фамилия character varying (64)															
1	Лазарева															
2	Новиков															

1.2. Подзапросы

Результатом выполнения запроса является набор кортежей, оформленный в виде таблицы. Данную таблицу возможно вывести на экран для просмотра или использовать в других запросах. В таком случае, запрос, используемый внутри другого запроса называют вложенным запросом или подзапросом.

Существуют два типа подзапросов:

Некоррелированный подзапрос – оператор SELECT вложенный в другой запрос SQL, не связанный с внешним запросом (он может быть выполнен отдельно от него).

Коррелированный подзапрос – оператор SELECT вложенный в другой запрос SQL, и ссылающийся на один или несколько столбцов внешнего запроса.

Рассмотрим типы запросов на примерах.

1.2.1. Некоррелированный запрос.

Выведем всех преподавателей, чей оклад больше среднего. Для этого создадим подзапрос, вычисляющий среднюю зарплату всех преподавателей вуза. Далее, используя скалярный результат этого подзапроса найдем все большие значения.

```
SELECT professor.surname, professor.name ,professor.current_position
FROM professor
WHERE professor.salary::numeric >
(
    SELECT AVG(professor.salary::numeric)
    FROM professor
)
```

surname	name	current_position
Широков	Василий	доцент
Семенов	Андрей	профессор
Филатов	Илья	доцент
Воронов	Николай	доцент
Орлов	Даниил	профессор
Овсянников	Даниил	доцент
Воронов	Артём	доцент
Михайлова	Анастасия	профессор

Выведем всех студентов, которые имеют хотя бы одну оценку 2 за один из экзаменов. Для этого создадим подзапрос, выбирающих всех двоечников и используя их номера студенческого билета найдем их фамилию и имя.

```
SELECT surname, name
FROM student
WHERE student.student_id IN(
    SELECT field_comprehension.student_id
    FROM field_comprehension
    WHERE field_comprehension.mark = 2
)
```

surname	name
Алехин	Андрей
Коровина	Мария
Кузьмин	Мирослав
Гаврилов	Александр
Самойлов	Артём
Новиков	Кирилл
Маркин	Даниэль
Беляков	Иван

1.2.2. Коррелированный запросы

Создадим аналогичный запрос с поиском всех студентов, имеющих хотя бы одну оценку 2, но с помощью коррелированного запроса. Обратите внимание, что во вложенном запросе происходит обращение к таблице «Студент», не указанной после ключевого слова FROM данного подзапроса. Если мы попытаемся запустить отдельно от основного данный подзапрос, то в результате будет получена ошибка.

```
SELECT surname, name
FROM student
WHERE 2 = ANY
(
    SELECT field_comprehension.mark
    FROM field_comprehension
    WHERE field_comprehension.student_id = student.student_id
)
```


surname	name
Алехин	Андрей
Коровина	Мария
Кузьмин	Мирослав
Гаврилов	Александр
Самойлов	Артём
Новиков	Кирилл
Маркин	Даниэль
Беляков	Иван

Создадим запрос, выводящий средний балл каждого из студентов. Для этого создадим подзапрос, вычисляющий средний балл и ссылающийся на атрибут "Студент"."Номер студенческого билета" из внешнего запроса.

```
SELECT surname, name, (
    SELECT CAST(AVG(mark) AS NUMERIC(2,1))
    FROM Field_comprehension
    WHERE Field_comprehension.student_id = Student.student_id
) AS "Средняя оценка"
FROM Student
ORDER BY "Средняя оценка" DESC;
```

surname	name	Средняя оценка
Князева	Есения	4.4
Хомяков	Илья	4.3
Андреева	Ева	4.3
Григорьев	Даниил	4.3
Селезнев	Максим	4.3
Казаков	Николай	4.3
Ершова	Валерия	4.3
Морозов	Илья	4.2

2. Практическая часть

Напишите SQL запросы к учебной базе данных в соответствии с вариантом.

2.1. Задание 1.

№ варианта	
1	Выведите фамилии и имена всех людей, кто причастен к вузу.
2	Выведите список имен студентов и преподавателей, которые совпадают.
3	Выведите имена студентов, которые не совпадают с именами преподавателей.
4	Сделайте проверку, все ли преподаватели трудоустроены.
5	Выведите номер студенческого билета студента, который получил 2 по любым дисциплинам.
6	Вывести ФИО преподавателей, которые работают на четверть ставки.

2.2. Задание 2.

№ варианта	
1	Выведите информацию о том, к каким институтам относятся группы.
2	Выведите ФИО преподавателей, их предметы и в каких группах они ведут.
3	Выполните запрос, выводящий группу, ее институт/кафедру, руководителя.
4	Выведите Фамилии и Имена студентов, кто получил 5 по Базам данных.
5	Вывести ФИО преподавателей, у которых зарплата больше 20 000.
6	Отсортировать результат предыдущего задания по алфавиту и исключить повторения после выполнения запроса.

2.3. Задание 3.

№ варианта	
1	Выведите все номера преподавателей и проверьте, есть ли совпадения с номерами студентов. В случае совпадения-вывести номер в столбцах студента и преподавателя.
2	Сделайте запрос на проверку трудоустройства преподавателей.
3	Выведите весь список студентов и проверьте оценки по всем дисциплинам только у ИБ-21.
4	Выведите студента, его группу и преподавателя, у которого фамилия = фамилия студента.
5	Выведите список принадлежности учебных групп к структурным подразделениям.
6	Выведите почты студентов и ФИО, у которых пароль начинается с 2002.

2.4. Задание 4.

№ варианта	
1	Выведите самое часто повторяемое имя у студентов.
2	Выведите полные названия структурных подразделений, который обучают очно.
3	Выведите номера студенческих билетов и когда истечет их срок у студентов группы ИБ-21.
4	Выведите фамилию и имя всех студентов, кто учится заочно.
5	Выведите преподавателей, среднюю оценку их студентов, которая выше средней по институту.
6	Сгруппировать предыдущее задание по группам. Посчитать, сколько человек из каждой группы с таким паролем и вывести.

2.5. Задание 5.

Самостоятельно разработайте 5 **осмысленных** запросов к базе данных, используя приведенные в данной лабораторной работе материалы. Вариант выбирается в соответствии с номером по списку.

	Вариант 1,5					Вариант 2,6					Вариант 3					Вариант 4				
Ключевое слово	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
INNER JOIN	+	+				+	+				+	+				+	+			
LEFT OUTER JOIN			+					+												
RIGHT OUTER JOIN													+					+		
UNION				+					+					+					+	
EXCEPT					+					+										
INTERSECT															+					+

Список литературы

- [1] Документация к PostgreSQL 15.1, 2022.
- [2] «Исходный код СУБД postgres,» [В Интернете]. Available: <https://github.com/postgres/postgres>. [Дата обращения: 30 01 2023].
- [3] Е. Рогов, PostgreSQL изнутри, 1-е ред., Москва: ДМК Пресс, 2023, р. 662 .
- [4] Б. А. Новиков, Е. А. Горшкова и Н. Г. Графеева, Основы технологии баз данных, 2-е ред., Москва: ДМК пресс, 2020, р. 582.
- [5] Е. П. Моргунов, PostgreSQL. Основы языка SQL, 1-е ред., Санкт-Петербург:

БХВ-Петербург, 2018, р. 336.