**Лабораторные работы по курсу**

**Базы данных**

**Лабораторная работа 3**

**«Использование объединяющих и вложенных запросов языка SQL»**

**Москва, 2024**

Оглавление

[1. Теоретическая часть 3](#_Toc157970428)

[1.1. Соединение таблиц 3](#_Toc157970429)

[1.1.1. Неявное соединение таблиц 3](#_Toc157970430)

[1.1.2. Соединение с помощью JOIN 4](#_Toc157970431)

[1.1.3. Объединение, разность, пересечение таблиц 6](#_Toc157970432)

[1.2. Подзапросы 7](#_Toc157970433)

[1.2.1. Некоррелированный запрос. 7](#_Toc157970434)

[1.2.2. Коррелированный запросы 8](#_Toc157970435)

[1.3. Общие табличные выражения 9](#_Toc157970436)

[1.4. Представления 9](#_Toc157970437)

[2. Практическая часть 10](#_Toc157970438)

[2.1. Задание 1. 10](#_Toc157970439)

[2.2. Задание 2. 10](#_Toc157970440)

[2.3. Задание 3. 10](#_Toc157970441)

[Контрольные вопросы 14](#_Toc157970442)

[Список использованной литературы 15](#_Toc157970443)

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. Соединение таблиц
     1. Неявное соединение таблиц

Самым простым способом является неявное соединение таблиц, когда таблицы объединяются перекрестно. Другими словами, каждой строка 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.

Рассмотрим для примера таблицы *Structural\_units, Employments, Professors* из учебной базы данных*.* Для упрощения сократим число строк в таблицах, оставив следующие значения.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Обратите внимание, что к третьему структурному подразделению не прикреплен ни один преподаватель и преподаватель по фамилии Быкова не прикреплена ни к одному подразделению.

Предположим, что нам необходимо вывести код всех преподавателей и название подразделения, в котором они трудоустроены. Для этого необходимо объединить таблицы *Structural\_units* и *Employment.*

Выполним следующий запрос:

**SELECT abbreviated\_title, professor\_id**

**FROM structural\_units, employments**

abbreviated\_title | professor\_id

-------------------+--------------

МПСУ | 810001

СПИНТех | 810001

ИЦОД | 810001

МПСУ | 820001

СПИНТех | 820001

ИЦОД | 820001

(6 rows)

В результате запроса будет выведены значения из двух таблиц, где каждому значению из первой будут соответствовать все значения из второй таблицы. Таким образом, будет выведено (число строк в первой таблице) \* (число строк во второй таблице). В данном примере будет выведено 6 строк.

В итоговой таблице будут присутствовать все преподаватели и все подразделения независимо от реального трудоустройства. Очевидно, что в этом нет никакого смысла.

Для того, чтобы выведенная информация связывала фамилию преподавателя и только то структурное подразделение, на котором он трудоустроен необходимо добавить условие отбора. Для вывода фамилии преподавателя вместо его кода используем таблицу *Professors.*

Выберем только те строки, где значения номера подразделения в таблицах *Structural\_units* и *Employments*, а также значения номера преподавателя в таблицах *Employments* и *Professors* совпадают.

Итоговый запрос будет выглядеть следующим образом:

**SELECT structural\_units.abbreviated\_title, professors.last\_name**

**FROM structural\_units, employments, professors**

**WHERE structural\_units.structural\_unit\_id = employments.structural\_unit\_id**

**AND employments.professor\_id = professors.professor\_id**

abbreviated\_title | last\_name

-------------------+-----------

МПСУ | Широков

СПИНТех | Воронов

(2 rows)

Этот результат несет реальную смысловую нагрузку. Как можно заметить из результата запроса, преподаватель Быкова и подразделение ИЦОД в итоговую выборку не попали, т. к. для них указанное условие не было выполнено.

Обратите внимание, что для того, чтобы обращаться к атрибуту конкретной таблицы, необходимо указывать название таблицы, отделив его от имени атрибута точкой (название\_таблицы.имя\_атрибута).

Для сокращения возможно использовать более короткую запись. Для этого после названия таблицы в выражении FROM нужно указать для неё псевдоним. Пример ниже аналогичен запросу, рассмотренному выше, однако выглядит более компактным.

SELECT su.abbreviated\_title, p.last\_name

FROM structural\_units su, employments e, professors p

WHERE su.structural\_unit\_id = e.structural\_unit\_id

AND e.professor\_id = p.professor\_id

* + 1. Соединение с помощью JOIN

Кроме неявного соединения таблиц в языке SQL существует альтернативная форма записи операций соединение таблиц с помощью ключевого слова JOIN. Объединение происходит по столбцу, который есть в каждой из таблиц. В результате запроса формируется соединенная таблица. Соединённая таблица ­­­­­­— это таблица, полученная из двух других таблиц (*T1, T2)* в соответствии с условием соединения. Общий синтаксис описания соединённой таблицы:

T1 тип\_соединения T2 [ условие\_соединения ]

По типу соединения операторы JOIN подразделяются на внутренние и внешние — INNER JOIN и OUTER JOIN, а также CROSS JOIN.

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

Формат:

SELECT столбцы FROM таблица1

[INNER] JOIN таблица2    ON условие1

[[INNER] JOIN таблица3 ON условие2]

Составим запрос, аналогичный примеру выше, в котором необходимо было вывести фамилии трудоустроенных преподавателей.

SELECT su.abbreviated\_title, p.last\_name

FROM structural\_units su

INNER JOIN employments e ON su.structural\_unit\_id = e.structural\_unit\_id

INNER JOIN professors p ON e.professor\_id = p.professor\_id

abbreviated\_title | last\_name

-------------------+-----------

МПСУ | Широков

СПИНТех | Воронов

(2 rows)

Результат выполнения запроса совпал с примером выше.

**Внешнее соединение** - **OUTER JOIN** позволяет включить в вывод все строки из одной или обоих таблиц. Его можно разделить на правое (RIGHT), левое (LEFT) и полное (FULL).

Формат:

SELECT столбцы FROM таблица1

 { LEFT | RIGHT | FULL }  [OUTER] JOIN таблица2  ON условие1

   [  { LEFT | RIGHT | FULL }  [OUTER] JOIN таблица3  ON условие2 ]…

Операция левого внешнего соединения **(LEFT)** возвращает все строки левой (первой) таблицы, включая те, для которых не нашлось парного значения в правой (второй) таблице. Вместо ненайденных значений атрибутов правой таблицы будет указано неопределенное значение NULL. Аналогично, операция правого внешнего соединения **(RIGHT)** возвращает все строки правой (второй) таблицы, включая те, для которых не нашлось пары в левой. Обратите внимание, операции правого и левого соединения не коммутативны: A LEFT JOIN B ≠ B LEFT JOIN A. Вывод A LEFT JOIN B совпадает с B RIGHT JOIN A.

Полное внешнее соединение **(FULL)** включает в себя все пересекающиеся строки и все непарные строки из обеих таблиц.

Рассмотрим примеры:

Выберем из таблиц *Structural\_units* и *Employments* значения номеров телефонов всех структурных подразделений и ставку трудоустроенных профессоров.

SELECT abbreviated\_title, phone\_number, professor\_id, wage\_rate

FROM structural\_units su

LEFT JOIN employments e ON su.structural\_unit\_id = e.structural\_unit\_id;

abbreviated\_title | phone\_number | professor\_id | wage\_rate

-------------------+--------------+--------------+-----------

МПСУ | 25-13 | 810001 | 0.25

СПИНТех | 23-45 | 820001 | 0.35

ИЦОД | 66-36 | null | null

(3 rows)

В данном примере была взята вся информация из первой (левой) таблицы и к ней присоединены значения, удовлетворяющие указанному условию. В противном случае проставлено значение *null*.

Рассмотрим второй пример. Выберем фамилии всех преподавателей и ставки тех из них, кто трудоустроен.

SELECT wage\_rate, last\_name

FROM employments e

RIGHT JOIN professors p ON e.professor\_id = p.professor\_id

wage\_rate | last\_name

-----------+-----------

0.25 | Широков

0.35 | Воронов

null | Быкова

(3 rows)

В текущем примере к таблице трудоустройства была присоединена таблица, содержащая преподавателей с использованием RIGHT JOIN. Таким образом была получена вся вторая (правая) таблица и удовлетворяющие условию значения из левой. Аналогично в пустые ячейки проставлено значение *null*.

**Декартово произведение - CROSS JOIN**

Формат:

SELECT столбцы FROM таблица1

CROSS JOIN таблица2

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

SELECT abbreviated\_title, professor\_id

FROM structural\_units

CROSS JOIN employments

abbreviated\_title | professor\_id

-------------------+--------------

МПСУ | 810001

СПИНТех | 810001

ИЦОД | 810001

МПСУ | 820001

СПИНТех | 820001

ИЦОД | 820001

(6 rows)

Ключевые слова OUTER и INNER являются устаревшим и команды LEFT OUTER JOIN, RIGHT OUTER JOIN, INNER JOIN возможно сокращать до LEFT JOIN, RIGHT JOIN и просто JOIN.

Если столбцы, по которым производится соединение, имеют одинаковое название, то вместо записи условия с помощью ключевого слова ON возможно использовать ключевое слово USING.

Формат:

SELECT столбцы FROM таблица1

{ [INNER] | { LEFT | RIGHT | FULL } [OUTER] } JOIN таблица2

USING ( список столбцов соединения )

В отличии от соединения с использованием ключевого слова ON, USING удаляет в итоговой таблице повторяющийся столбец, по которому происходит соединение. Например запрос, указанный ниже, аналогичен рассмотренному примеру на внутреннее соединение.

SELECT abbreviated\_title, last\_name

FROM structural\_units

INNER JOIN employments USING(structural\_unit\_id)

INNER JOIN professors USING(professor\_id)

* + 1. Объединение, разность, пересечение таблиц

Кроме соединения таблиц, когда в результате операции атрибуты (столбцы) 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, при котором тип данных каждого столбца первой таблицы должен совпадать с типом данных соответствующего столбца во второй таблице. Также должно совпадать количество выбранных столбцов из всех таблиц. Имена столбцов в объединяемых таблицах не обязательно должны быть одинаковыми.

В языке SQL для объединения таблиц используется оператор UNION.

Формат:

SELECT\_выражение1

UNION [ALL] SELECT\_выражение2

[UNION [ALL] SELECT\_выражениеN]

Исходные таблицы могут содержать идентичные строки, тогда при объединении по умолчанию повторяющиеся строки удаляются. Если необходимо чтобы общая таблица содержала все строки, включая повторяющиеся используют параметр ALL.

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 |

Рассмотрим еще один пример. Вычислим

|  |  |  |
| --- | --- | --- |
| SQL запрос | Возвращенные значения | Комментарий |
| SELECT professor\_id FROM professors  UNION  SELECT professor\_id FROM employments; | professor\_id  --------------  820001  840001  810001  (3 rows) | Объединение всех кодов преподавателя без повторяющихся значений. |
| SELECT professor\_id FROM professors  INTERSECT  SELECT professor\_id FROM employments; | professor\_id  --------------  810001  820001  (2 rows) | Вывод только трудоустроенных преподавателей |
| SELECT professor\_id FROM professors  EXCEPT  SELECT professor\_id FROM employments | professor\_id  --------------  840001  (1 row) | Вывод нетрудоустроенных преподавателей. | |

* 1. Подзапросы

Результатом выполнения запроса является набор кортежей, оформленный в виде таблицы. Данную таблицу возможно вывести на экран для просмотра или использовать в других запросах. 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.

Для создания подзапросов иногда бывают полезными ключевые слова ALL, ANY, EXISTS, IN.

|  |  |
| --- | --- |
| Ключевое слово | Применение |
| условие **ALL** (подзапрос) | Если все значения подзапроса удовлетворяет условию, то оператор возвращает *true* |
| условие **ANY** (подзапрос) | Если хотя бы одно значение подзапроса удовлетворяет условию, то оператор возвращает *true* |
| **EXISTS** (подзапрос) | Если подзапрос содержит хотя бы одну строку, то оператор возвращает *true* |
| Выражение [**NOT**] **IN** (подзапрос) | Если подзапрос [не] содержит строку, полученную в результате вычисления выражения, то он возвращает *true* |

* + 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 last\_name, first\_name, salary::numeric**

**FROM professors**

**WHERE salary::numeric >**

**(**

**SELECT AVG(salary::numeric)**

**FROM professors**

**)**

last\_name | first\_name | salary

---------------+------------+-----------

Широков | Василий | 98000.00

Семенов | Андрей | 210000.00

Филатов | Илья | 85000.00

Воронов | Николай | 98000.00

Серебряков | Александр | 150000.00

Воронов | Артём | 98000.00

Михайлова | Анастасия | 210000.00

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

)

* + 1. 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 *Students*, не указанной после ключевого слова FROM данного подзапроса. Если мы попытаемся запустить отдельно от основного данный подзапрос, то в результате будет получена ошибка.

**SELECT last\_name, first\_name**

**FROM students**

**WHERE 2 = ANY**

**(**

**SELECT mark**

**FROM field\_comprehensions**

**WHERE field\_comprehensions.student\_id = students.student\_id**

**)**

Создадим запрос, выводящий средний балл каждого из студентов. Для этого создадим подзапрос, вычисляющий средний балл и ссылающийся на атрибут **student\_id** из таблицы *Students*, используемой во внешнем запросе.

**SELECT last\_name, first\_name, (**

**SELECT CAST(AVG(mark) AS NUMERIC(2,1))**

**FROM Field\_comprehensions**

**WHERE Field\_comprehensions.student\_id = Students.student\_id**

**) AS "Средняя оценка"**

**FROM Students**

**ORDER BY "Средняя оценка" DESC;**

last\_name | first\_name | Средняя оценка

------------------+------------+----------------

Савельев | Максим | 4.3

Гаганов | Павел | 4.2

Лембке | Юлия | 4.1

Фадеева | Ксения | 4.1

Лобов | Александр | 4.1

Иртеньев | Николай | 4.1

Верховцев | Иван | 4.1

Ракитин | Михаил | 4.0

Ильин | Владимир | 4.0

* 1. Общие табличные выражения

Для упрощения сложного запроса возможно использовать конструкцию CTE (Common table expression), позволяющую разбить его на несколько частей. Эта конструкция   определяет временные таблицы, которые существуют только для одного запроса. CTE похожи на вложенные запросы, но более оптимизированы. Вложенный запрос в отличие от CTE повторяется для каждой строки, которую нашел основной запрос, что повышает ресурсоемкость и замедляет работу кода.

Формат:

WITH <название выражения> AS (

SELECT | INSERT | UPDATE |DELETE

), AS …

SELECT …

Например, следующий запрос вычисляет всех студентов, чьи оценки выше средней по всему вузу.

**WITH avg\_mark AS (**

**SELECT cast(avg(mark) AS numeric(2,1)) AS a\_mark FROM field\_comprehensions**

**)**

**SELECT s.student\_id,**

**last\_name,**

**first\_name,**

**cast(avg(mark) AS numeric(2,1)) AS "avg. mark"**

**FROM students s**

**INNER JOIN field\_comprehensions fc ON s.student\_id = fc.student\_id**

**GROUP BY s.student\_id**

**HAVING avg(mark) > (SELECT a\_mark FROM avg\_mark)**

**ORDER BY "avg. mark" DESC;**

student\_id | last\_name | first\_name | avg. mark

------------+------------------+------------+-----------

828170 | Савельев | Максим | 4.3

830114 | Гаганов | Павел | 4.2

892391 | Фадеева | Ксения | 4.1

834421 | Иртеньев | Николай | 4.1

816993 | Лембке | Юлия | 4.1

825852 | Лобов | Александр | 4.1

833207 | Верховцев | Иван | 4.1

895508 | Ракитин | Михаил | 4.0

894671 | Бессонов | Алексей | 4.0

* 1. Представления

При работе с базами данных очень часто приходится выполнять одинаковые сложные и объемные запросы. Каждый раз составлять длинный запрос оказывается весьма трудоемко. Для упрощения работы возможно сформировать из такого запроса представление, к которому далее возможно обращаться, как будто это обычная таблица какое угодно число раз. Представление состоит из строк и столбцов, которые могут формироваться из одной или нескольких таблиц. Сокращенный формат команды:

CREATE VIEW ***имя*** AS ***запрос***

Например, создадим представление на основе запроса, выводящего трудоемкость дисциплины в формате «Количество часов/ЗЕТ»

CREATE VIEW labor\_intensity AS

SELECT field\_name AS "Field name", (36\*zet::numeric)::varchar || '/' || zet as "Labor intensity"

FROM field

ORDER BY "Labor intensity"

Далее к созданному представлению возможно обратиться, как к таблице.

SELECT \* FROM labor\_intensity

Field name | Labor intensity

---------------------------------------------------------------------+-----------------

Риторика | 108/3

Основы управления проектами | 108/3

Основы рыночной экономики | 108/3

Безопасность жизнедеятельности | 108/3

Экология | 108/3

Иностранный язык | 108/3

История | 108/3

Правоведение | 108/3

1. Практическая часть
   1. Задание 1.

with count\_degree AS (

SELECT DISTINCT degree from professors

) SELECT count(degree) FROM count\_degree

Рассмотрите следующий запрос. Какой результат был получен после его выполнения. Измените запрос, поставив вместе значения degree в агрегатную функцию символ «\*». Сравните результаты выполнения запросов и объясните его.

* 1. Задание 2.

Напишите SQL запросы к учебной базе данных в соответствии с вариантом. Вариант к практической части выбирается по формуле: V = (N % 10) +1, где N – номер в списке группы, % - остаток от деления. Обратите внимание, что группировка заданий по разделам означает не столько подсказку, сколько **обязательное** использование данных параметров. Некоторые запросы возможно решить без использования указанных операторов, однако задача лабораторной работы познакомить вас с разнообразием возможностей PostgreSQL, поэтому их использование необходимо.

|  |  |
| --- | --- |
| № варианта | № запросов |
| 1 | 1, 11, 21, 31, 41, 51, 61, 71, 81 |
| 2 | 2, 12, 22, 32, 42, 52, 62, 72, 82 |
| 3 | 3, 13, 23, 33, 43, 53, 63, 73, 83 |
| 4 | 4, 14, 24, 34, 44, 54, 64, 74, 84 |
| 5 | 5, 15, 25, 35, 45, 55, 65, 75, 85 |
| 6 | 6, 16, 26, 36, 46, 56, 66, 76, 86 |
| 7 | 7, 17, 27, 37, 47, 57, 67, 77, 87 |
| 8 | 8, 18, 28, 38, 48, 58, 68, 78, 88 |
| 9 | 9, 19, 29, 39, 49, 59, 69, 79, 89 |
| 10 | 10, 20, 30, 40, 50, 60, 70, 80, 90 |

* 1. Задание 3.

Самостоятельно разработайте 3 **осмысленных** запроса к базе данных, используя приведенные в данной лабораторной работе материалы. Вариант выбирается в соответствии с номером по списку. Из созданных запросов создайте представления.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Вариант: | **1,2,9** | | | **3,4,10** | | | **5,6** | | | **7,8** | | |
| Номер запроса: | **1** | **2** | **3** | **1** | **2** | **3** | **1** | **2** | **3** | **1** | **2** | **3** |
| INNER JOIN | + |  |  | + |  |  | + |  |  | + |  |  |
| LEFT JOIN |  | + |  |  |  |  |  | + |  |  |  |  |
| RIGHT JOIN |  |  |  |  | + |  |  |  |  |  | + |  |
| UNION |  |  | + |  |  |  |  |  |  |  |  | + |
| EXCEPT |  |  |  |  |  | + |  |  |  |  |  |  |
| INTERSECT |  |  |  |  |  |  |  |  | + |  |  |  |

Сборник запросов к лабораторной работе

***INNER JOIN***

1. Вывести ФИО, должности, оклад, номер трудового договора всех преподавателей
2. Вывести ФИО, номера студенческих билетов, оценки по освоению дисциплин и названия дисциплин для всех студентов
3. Вывести наименование группы и наименование структурного подразделения, к которому она относится (для всех групп)
4. Вывести ФИО преподавателей, названия преподаваемых дисциплин и название групп, у которых они ведутся. Отсортировать по номеру группы.
5. Вывести ФИО преподавателя, названия преподаваемых дисциплин и все оценки ими выставленные. Исключить из вывода незаполненные поля оценок. Отсортировать по фамилии, имени и дисциплине.
6. Вывести номер трудового договора и название структурного подразделения, к которому он относится. Отсортировать по номеру трудового договора.
7. Вывести название структурного подразделения, номер трудового договора и ФИО преподавателя, который трудоустроен в этом структурном подразделении. Отсортировать по названию структурного подразделения и фамилии и имени преподавателя.
8. Вывести ФИО студента и освоенные им дисциплины с оценками. Отсортировать по фамилии и имени.
9. Вывести ФИО преподавателей, названия преподаваемых ими дисциплин. Отсортировать по фамилии и имени.
10. Вывести фамилию и имя преподавателей, у которых отсутствует отчество, а также названия преподаваемых ими дисциплин. Отсортировать по фамилии и имени.
11. Вывести названия групп, сокращенной название подразделения и ФИО руководителей структурного подразделения.
12. Вывести ФИО всех студентов, освоивших дисциплину Базы Данных на 5. Отсортировать по фамилии и имени.
13. Вывести все дисциплины, которые ведутся у группы ИТД-31
14. Вывести полные названия структурных подразделений и названия дисциплин, преподаваемых в них. Отсортировать по названию структурного подразделения.
15. Вывести ФИО и ставку всех трудоустроенных преподавателей. Отсортировать по ставке.
16. Вывести ФИО каждого студента и срок действия его студенческого билета.
17. Вывести ФИО всех студентов и форму их обучения. Включить в список только заочников.
18. Вывести ФИО преподавателей, их ставку и стаж. Включить в список преподавателей со стажем больше 5 лет.
19. Вывести ФИО всех студентов с оценками, освоивших дисциплину «Философия» меньше, чем на 4. Отсортировать по фамилии и имени.
20. Вывести ФИО, номер группы студентов имеющих оценку 5 по дисциплине 'Фотографика'.
21. Подсчитать количество отличников в каждой группе по каждой дисциплине, включающей в себя слово “Физика”.
22. Вывести названия групп, в которых есть хотя бы 1 студент, освоивший дисциплину “Физика“ на 5 (название дисциплины должно содержать слово “Физика“). Отсортировать по номеру группы.
23. Вывести руководителя структурного подразделения, номера групп входящие в его подразделение и количество студентов в каждой группе. Отсортировать по фамилии руководителя.
24. Вывести ФИО всех преподавателей и их зарплату, если зарплата больше средней по России (зарплата = оклад \* ставка)
25. Найти среднюю оценку студентов по каждой дисциплине. Вывести полное название дисциплины и среднюю оценку. Всем столбцам дать русское название. Отсортировать по оценке.
26. Подсчитать количество каждой оценки («2», «3», «4», «5»), которые выставили преподаватели. Вывести фамилию, имя, отчество преподавателя, наименование дисциплины, оценку и ее количество. Исключить из вывода незаполненные поля оценок. Отсортировать по фамилии, имени и дисциплине.
27. Вывести названия первых групп (включая вечерников) и названия их структурных подразделений.
28. Найти среднюю оценку каждого студента, родившегося в 2004 году по всем

дисциплинам, отсортировать, по средней оценке, в порядке убывания. Вывести фамилию, имя, номер студенческого и среднюю оценку.

1. Подсчитать количество всех оценок у студентов чьи номера студенческих билетов лежат в интервале 820000–850000. Вывести фамилию, имя, номер студенческого, оценку и ее количество. Исключить из подсчета незаполненные поля оценок. Отсортировать по номеру студенческого билета.
2. Вывести сокращенное название подразделения, которым руководит Соколова, фамилию и должность преподавателей, которые входят в это подразделение, а также дисциплины, которые ведут преподаватели этого подразделения.
3. Вывести ФИО студентов и преподавателей, у которых совпадают фамилии.
4. Вывести список студентов из групп ИВТ. Которые сдали на отлично более 10 предметов.
5. Создать пароль студентам по следующему правилу: первые пять символов почты + день рождения + месяц рождения +две цифры номера группы. Вывести фамилию, имя, дату рождения, почту, форму обучения, дату выдачи студенческого билета. Оставить только студентов очной формы обучения с датой выдачи студенческого билета> 01.01.2022.
6. Вывести список всех студентов, обучающихся заочно, отсортировать по фамилиям, в алфавитном порядке оставить только тех, кому больше 20 лет
7. Вывести всех студентов группы ИБ, отсортировать по числу 5 по дисциплине «Базы Данных». Оставить только тех, у кого 5к больше 5
8. Вывести фамилию, имя, номер студенческого и среднюю оценку студента по всем дисциплинам исключая ‘Философию’. Отсортировать, по средней оценке, в порядке убывания.
9. Вывести название предмета, фамилию и имя преподавателя, который ведет этот предмет и среднюю оценку у всех студентов за этот предмет. Оставить только те предметы, которые читаются в первом семестре.
10. Вывести фамилию, имя преподавателя, количество дисциплин, которые он преподает и его оклад. Оставить только тех, у кого количество дисциплин > 5. Отсортировать по фамилии.
11. Выведите полные названия структурных подразделений, название групп в него входящих и количество студентов в каждой группе. Оставьте только группы содержащие в своем названии буквы “В” и “Б” и оканчивающиеся цифрой “1”. Отсортируйте по номеру группы.
12. Вывести полное название структурного подразделения, его код, средний, минимальный и максимальный оклад преподавателей, которые работают в этом подразделении. Дать столбцам соответствующие русские названия. Отсортировать по коду подразделения.

***LEFT JOIN, RIGHT JOIN, FULL JOIN***

1. Вывести руководителей всех структурных подразделений и название дисциплины, относящееся к этим структурным подразделениям (если они есть, иначе - null). Отсортировать по дисциплине в обратном порядке.
2. Вывести фамилии преподавателей и фамилии их однофамильцев среди студентов (если есть, иначе - null). Использовать *LEFT JOIN или RIGHT JOIN.*
3. Вывести названия структурных подразделений и группы, которые им принадлежат (если они есть, иначе - null)
4. Вывести ФИО преподавателя и название дисциплины, которую он ведёт (если нет - null)
5. Вывести список преподавателей и наименование структурного подразделения, в котором они трудоустроены (если не трудоустроены - null)
6. Вывести ФИО всех студентов и форму их обучения
7. Вывести все пары фамилий преподавателей, у которых одинаковые имена
8. Вывести ФИО всех студентов и предмет, по которым у них долг (если долга нет - null), отсортировать по ФИО.
9. Вывести всех руководителей структурных подразделений, их номера телефонов и ФИ и электронные почты всех студентов
10. Вывести все пары фамилий студентов родившихся в один день

**UNION/EXCEPT/INTERSECT**

1. Вывести фамилии всех студентов и преподавателей в (дубли оставить). Отсортировать по фамилии. Добавить столбец, в котором обозначить преподаватель это или студент.
2. Вывести название всех групп и всех структурных подразделений. Отсортировать по названию.
3. Вывести ФИО всех преподавателей и руководителей структурных подразделений в одной колонке. Добавить колонку, где указать – “руководитель” для соответствующей записи.
4. Вывести в одном столбце: максимальный оклад среди преподавателей, средний оклад среди преподавателей, минимальный оклад среди преподавателей. В дополнительном столбце сделать соответствующие пояснения.
5. Вывести однофамильцев среди студентов и преподавателей (фамилию). Использовать *UNION/EXCEPT/INTERSECT.*
6. Вывести названия всех групп, в последней строчке вывести: ‘Итого студентов' и количество всех студентов в институте.
7. Выведите фамилии всех преподавателей и их зарплату. В первой строчке вывести: ‘Средняя зарплата' (можно начать с латинской буквы “C” ) и значение средней зарплаты у преподавателей.
8. Вывести фамилии преподавателей, у которых нет однофамильцев среди студентов.
9. Вывести ФИО, оклад и должность всех преподавателей, кроме преподавателей МПСУ. Сортировать по убыванию оклада. Использовать EXCEPT.
10. Вывести ФИО студентов и сокращенное название подразделения, к которому относится их группа. В список включить всех студентов, кроме тех, у кого группа принадлежит МПСУ. Использовать EXCEPT.
11. Вывести название всех групп, количество студентов в них, код всех структурных подразделений, к которым привязаны группы и количество групп в них.
12. Выведите в два столбца название семестра (например, “Семестр 1”) и количество читаемых в этом семестре дисциплин. Внизу в эти же два столбца добавите фамилии преподавателей, которые читают дисциплины в 8 семестре и количество этих дисциплин.
13. Вывести название всех групп и число студентов в них, в последней строчке вывести: ‘Итого студентов' и количество всех студентов в институте.
14. Вывести всех преподавателей и студентов, у которых совпадают первые буквы имени и фамилии. Добавить столбец, в котором обозначить преподаватель это или студент.
15. Вывести ФИО всех студентов вторых групп (включая вечерников), их номера групп и преподавателей-ассистентов, читающих дисциплины в 5 семестре. У преподавателей в столбец номера группы записать 'преподаватель’.
16. Вывести имена студентов из первой и второй групп, количество этих имен, столбец, содержащий значение 'первая' или ‘вторая' в зависимости от номера группы.
17. Вывести ФИО всех мужчин в институте (среди преподавателей и студентов). Учесть очень большое, в отличии от женщин, разнообразие окончаний фамилий и частое отсутствие отчеств. Добавить столбец, в котором обозначить преподаватель это или студент.
18. Вывести ФИО всех женщин в институте (среди преподавателей и студентов). Пол определить по окончанию фамилии или отчества. Добавить столбец, в котором обозначить преподаватель это или студент.
19. Найдите ФИО всех девушек среди студентов (пол определить по окончанию фамилии или отчества). Используя найденную информацию выведите всех юношей среди студентов.
20. Вывести фамилии всех студентов, чьи фамилии начинаются на букву “Ф” и преподавателей-ассистентов, чьи фамилии начинаются на буквы “К” и “И”. Добавить столбец, в котором обозначить преподаватель это или студент.

**В*ложенные запросы***

1. Вывести фамилию, имя студентов и их оценку по дисциплине ‘ Базы Данных’, если она больше средней оценки всех студентов за эту дисциплину.
2. Вывести номера трудовых договоров и ставки тех преподавателей, у которых ее значение выше средней. В конце таблицы добавить строку - ‘Средняя ставка’ и вписать ее значение.
3. Вывести фамилию, имя и оценку студентов, у которых оценка хотя бы по одной из освоенных дисциплин выше, чем средняя по ВУЗу.
4. Вывести фамилию, имя и оценку студентов группы ИБ-21, у которых хотя бы одна оценка выше средней по группе.
5. Вывести всех отличников (студентов, у которых за все дисциплины оценка 5)
6. Вывести средний возраст студентов каждого структурного подразделения
7. Вывести фамилию, имя, группу и возраст студента, у которого возраст меньше среднего по всем студентам.
8. Вывести максимальное количество студентов в группе по всему ВУЗу. Использовать вложенный запрос.
9. Вывести фамилия, имя и оклад всех преподавателей, чей оклад меньше среднего. Отсортируйте по окладу в порядке убывания. Первой строчкой выведите средний оклад.
10. Вывести фамилию и имя преподавателя с максимальным стажем работы.
11. Вывести фамилию, имя, группу и возраст тех студентов, у которых возраст больше среднего по группе. Первой строчкой в каждой группе при выводе должна быть запись – ‘Средний возраст’(для сортировки можно использовать первую букву латинскую ‘C’) + номер группы + средний возраст по группе. Далее должны быть выведены студенты этой группы, чей возраст больше среднего.
12. Найдите название дисциплины, по которой получено наибольшее количество пятерок. Выведите фамилию, имя, отчество студентов, получивших пять по этой дисциплине.
13. Вывести фамилию, имя преподавателей и максимальную трудоемкость среди преподаваемых ими дисциплин. В выборку включить только тех преподавателей, у которых максимальная трудоемкость меньше средней. Первой строкой вывести среднюю трудоемкость.
14. Вывести фамилию, имя, номер группы студентов, которые учатся в группе с максимальным количеством человек.
15. Вывести преподавателя, у которого наибольшее количество выставленных двоек за определенную дисциплину. Вывести его фамилию, имя, дисциплину и количество двоек.
16. Вывести фамилию, имя и группу студентов, у которых троек больше, чем четверок.
17. Выведите все дисциплины семестра, в котором читаются наибольшее количество дисциплин.
18. Вывести фамилию, имя стаж и зарплату преподавателей всех преподавателей, у которых стаж работы больше среднего, а зарплата ниже средней. Отдельной строкой вывести средний стаж и зарплату всех преподавателей.
19. Вывести фамилию, имя всех студентов, чей средний балл выше среднего балла в ИТД-21
20. Выведите фамилию, имя и группу студентов, в которой наибольший средний возраст

***Дополнительные запросы***

1. Вывести номера студенческих билетов студентов, у которых нет ни одной двойки.
2. Вывести номера всех преподавателей, которые не являются преподавателями дисциплины ‘Базы Данных’ (использовать EXCEPT).
3. Выведите всех преподавателей, которые преподают в первом семестре, но не преподают во втором. Используйте INTERSECT.
4. Вывести номера нетрудоустроенных преподавателей (без трудового договора)
5. Вывести ФИ и номера студенческих билетов студентов, а также дисциплины по которым у них нет двоек
6. Вывести в одном столбце: - самое популярное имя среди студентов; - фамилии всех студентов с этим именем.
7. Вывести в одном столбце: - email-адреса всех студентов; - число уникальных доменных адресов (gmail.com, yandex.ru)
8. Выведите фамилию, имя и группу студентов, у которых троек больше, чем четверок. Отдельными столбцами выведите количество этих оценок.
9. Вывести ФИО и дату рождения самого молодого студента
10. Выведите 10 лучших студентов 3-го курса института МПСУ (имеющих самую высокую среднюю оценку), окончивших учебу без троек и двоек. Отсортируйте ее по убыванию среднего балла. Таблица должна содержать ФИО, номер студенческого билета и средний балл.

Контрольные вопросы

1. Существует ли отличие между использованием ключевых слов INNER JOIN или перечисления таблиц через запятую в предложении FROM?
2. Для чего предназначено соединение LEFT JOIN?
3. В чем отличие коррелированного от некоррелированного подзапроса?
4. Назовите отличие соединения таблиц от их объединения.
5. В каких случаях используется ключевое слово INNER и OUTER?
6. В чем разница использования конструкции CTE и вложенного запроса?
7. В чем отличие представления от обычного запроса?

Список использованной 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, p. 662 . |
| [4] | Б. А. Новиков, Е. А. Горшкова и Н. Г. Графеева, Основы технологии баз данных, 2-е ред., Москва: ДМК пресс, 2020, p. 582. |
| [5] | Е. П. Моргунов, PostgreSQL. Основы языка SQL, 1-е ред., Санкт-Петербург: БХВ-Петербург, 2018, p. 336. |