Федеральное государственное образовательное бюджетное учреждение

высшего профессионального образования

«Нижегородский Государственный Университет им.

Н.И.Лобачевского» (ННГУ)

Национальный исследовательский Университет

Институт Информационных Технологий Математики и Механики

**Отчёт по лабораторной работе №3**

Выполнил:

студент группы 3821Б1ПМ3

Колганов Д. В.

Проверила:

доцент кафедры математического

обеспечения и суперкомпьютерных

технологий

Малкина Е.В

Нижний Новгород

2023 г.

Оглавление

[Введение 3](file:///C:\Users\kolgd\Downloads\Telegram%20Desktop\Еремеев_3.docx#_Toc132501044)

[Цели работы 4](file:///C:\Users\kolgd\Downloads\Telegram%20Desktop\Еремеев_3.docx#_Toc132501045)

[Задания 4](file:///C:\Users\kolgd\Downloads\Telegram%20Desktop\Еремеев_3.docx#_Toc132501046)

[Ход работы 6](file:///C:\Users\kolgd\Downloads\Telegram%20Desktop\Еремеев_3.docx#_Toc132501047)

[1. Простой выбор данных 8](file:///C:\Users\kolgd\Downloads\Telegram%20Desktop\Еремеев_3.docx#_Toc132501048)

[2. Внешнее объединение таблиц. 11](file:///C:\Users\kolgd\Downloads\Telegram%20Desktop\Еремеев_3.docx#_Toc132501049)

[3. Выбор данных с помощью группирующих запросов с условием 14](file:///C:\Users\kolgd\Downloads\Telegram%20Desktop\Еремеев_3.docx#_Toc132501050)

[4. Выбор данных с помощью подзапросов. 16](file:///C:\Users\kolgd\Downloads\Telegram%20Desktop\Еремеев_3.docx#_Toc132501051)

[Выводы 18](file:///C:\Users\kolgd\Downloads\Telegram%20Desktop\Еремеев_3.docx#_Toc132501052)

[Заключение 18](file:///C:\Users\kolgd\Downloads\Telegram%20Desktop\Еремеев_3.docx#_Toc132501053)

# Введение

Базы данных-это сложные, эффективные структуры, позволяющие хранить в себе разнообразную информацию о каком-то количестве объектов, а также связи между этими объектами, что позволяет удобно размещать информацию, а также находить и обновлять её. Предметами, описываемыми базами данных могут быть совершенно различные сущности, которые состоят из нескольких частей, значения которых содержатся в базах данных. При описании предметной области важно правильно определить, какие будут сущности для неё, а также как их описать и связать, чему мы научились в ходе первой лабораторной работы. В данной же лабораторной работе мы научимся работать с запросами, то есть выводить нужную нам информацию из базы данных при помощи соответствующего синтаксиса. Запросы будут рассмотрены самые разнообразные, для того, чтобы лучше разобраться в создании и исполнении запросов. Если наша база данных небольшая, она нужна для освоения навыков, то вот реальные базы данных огромные и запросы сильно облегчают поиск в них. Для этого будем использовать приложение «Microsoft Access».

Данная лабораторная работа направлена на получение базовых навыков по созданию запросов к базе данных, что поможет в дальнейшем лучше ориентироваться в базах данных, извлекать из них нужную информацию быстрым способом, а не просматривать всю базу данных вручную. Хоть и созданная в прошлой лабораторной работе база данных получилась небольшая, в реальности базы данных содержат миллионы записей и множество фигурирующих в них таблиц и запросы могут очень сильно облегчить поиски по базе данных.

# Цели работы

Приобретение навыков реализации запросов пользователя на выборку данных из таблиц, добавление, удаление и редактирование информации. Приобретение практических навыков использования языка SQL, внутри СУБД MS Access. Работа со сформированной в лабораторной работе 2 базой данных:

* Выборка данных из одной таблицы. Выбор отдельных полей таблицы. Квалифицированный выбор – предложение WHERE. Сложные условия (использование OR, AND, NOT).
* Выборка данных из связанных таблиц. Работа с главными и подчиненными таблицами (Master-Detail).
* Создание вычислимых полей.
* Сортировка результатов запроса.
* Проблема агрегирования данных. Изучение агрегатных функций (SUM, AVG, MAX, MIN, COUNT).
* Подсчет простых итогов.
* Организация группировки. Группировка по нескольким полям.
* Организация отбора при группировке.
* Создание перекрестных запросов.
* Создание новых таблиц.
* Создание запросов на добавление, редактирование, удаление.

# Задания

В ходе лабораторной работы необходимо выполнить следующие задания, связанные с реализацией запросов.

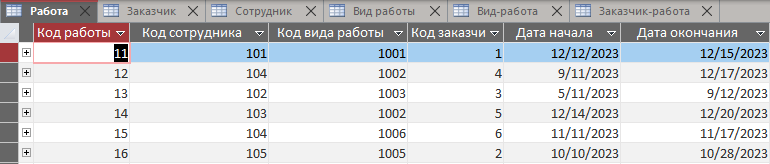
1. Простой выбор данных (SELECT, \*, FROM, WHERE, ORDER BY, вычислимые поля, DISTINCT).
   1. Создать простой запрос на выборку из одной таблицы. Включить несколько полей таблицы.
   2. Включить в запрос все поля с помощью знака «\*».
   3. Создать запрос на выборку данных из двух связанных таблиц.
   4. Выбрать несколько полей, по которым сортируется вывод.
   5. Определить условия отбора («И» и «ИЛИ»). Создать не менее двух запросов.
   6. Определить условия отбора с помощью параметра запроса.
   7. Создать вычислимые поля.
   8. Создать отсортированный по вычислимому полю запрос из нескольких таблиц, в котором определены условия «И» и «ИЛИ».
   9. Использовать предложение DISTINCT.
2. Внешнее объединение таблиц.
   1. Создать запрос на внешнее объединение таблиц.
   2. Продемонстрировать использование предложений Is null, Is not null.
   3. Использовать предложение LIKE.
   4. Использовать оператор UNION.
3. Выбор данных с помощью группирующих запросов с условием (GROUP BY, HAVING, MIN (), MAX (), SUM (), COUNT (), …).
   1. Создать итоговый запрос, содержащий несколько итоговых цифр.
   2. Создать простой группирующий запрос.
   3. Создать группирующий запрос с группировкой по нескольким полям.
   4. Создать группирующий запрос, в котором определяются условия, причем сначала выполняются вычисления, а затем происходит отбор.
   5. Создать группирующий запрос, в котором определяются условия, причем сначала происходит отбор, а затем выполняются вычисления.
   6. Создать группирующий запрос, в котором есть вычислимое выражение, содержащее несколько итоговых полей.
4. Выбор данных с помощью подзапросов.
   1. Создать запрос с выбором при помощи In.
   2. Использовать предложения ALL, ANY, EXISTS.

# Ход работы

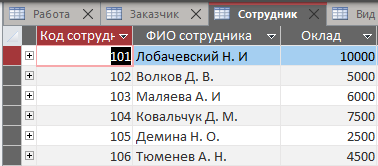
Итак, в ходе лабораторной работы будут постепенно выполнены поставленные задания, то есть постепенно созданы запросы, которые удовлетворяют заданиям. Состоять лабораторная будет из перечисления всех запросов, с пояснениями и скриншотами, чтобы было понимание, как работает какая-либо функция языка SQL. Далее по пунктам будут представлены запросы.

Для начала напомним содержание всех таблиц, для этого с помощью запросов выведем их все, для этого необходимо создать следующий запрос: SELECT \* FROM <table>, где под последним воспринимается название соответствующей таблицы.

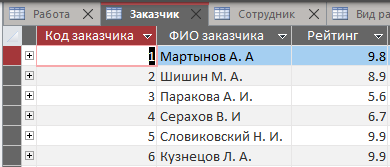
**Таблица Работа:**



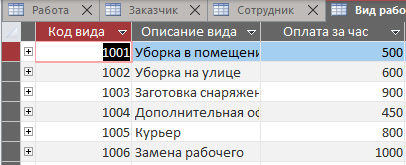
**Таблица Сотрудник:**



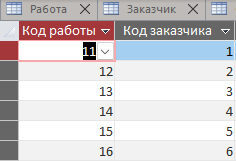
**Таблица Заказчик:**

****

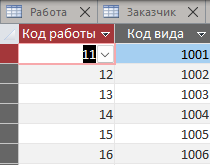
**Таблица Вид работы:**



**Таблица Заказчик-работа:**



**Таблица Вид-работа**



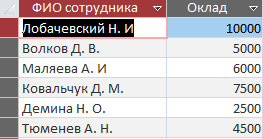
## 1. Простой выбор данных

1.1. Создать простой запрос на выборку из одной таблицы. Включить несколько полей таблицы.

Пусть нужно вывести только ФИО Клиентов и номера их телефонов, сделаем это с помощью следующего запроса:

SELECT Сотрудник.[ФИО сотрудника], Сотрудник.Оклад

FROM Сотрудник;

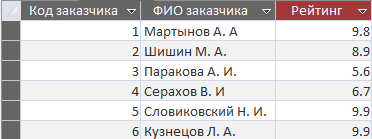


Оператор SELECT отвечает за то, чтобы указать СУБД что необходимо выбирать записи, далее идут названия столбцов, которые нас интересуют, которые нужно выбрать. Оператор FROM указывает СУБД откуда нужно выбрать, то есть из какой таблицы, ну а после этого оператора идёт название таблицы, из которой надо выбрать данные.

1.2. Включить в запрос все поля с помощью знака «\*».

Выведем с помощью запроса всю таблицу Заказчики

SELECT \* FROM Заказчики;

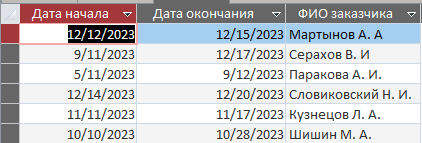


Оператор «\*» используется как раз для того, чтобы указать СУБД, что необходимо выбрать все из таблицы, он ставится вместо столбцов из предыдущего запроса.

1.3. Создать запрос на выборку данных из двух связанных таблиц.

SELECT Работа.[Дата начала], Работа.[Дата окончания], Заказчик.[ФИО заказчика]

FROM Заказчик INNER JOIN Работа ON Заказчик.[Код заказчика] = Работа.[Код заказчика];



Для выполнения данного задания создадим запрос, который выводит информацию о дате начала и окончания, а также ФИО Заказчика. Для этого теперь выбираем поля из разных таблиц при помощи символа «.», с указанием на таблицу, из которой его взять, а также делаем запрос из объединенной таблицы по ФИО поставщика с помощью INNER JOIN, и после слова ON указываем, что выбираем по совпадающему ФИО клиента.

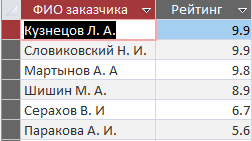
1.4. Выбрать несколько полей, по которым сортируется вывод.

Пусть нужно вывести таблицу Пост, отсортированную.

SELECT Заказчик.[ФИО заказчика], Заказчик.Рейтинг

FROM Заказчик

ORDER BY Заказчик.Рейтинг DESC;



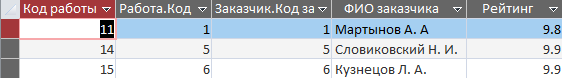
Приведём пример сортировки выбранных данных. Мы выбираем из таблицы все записи, а далее с помощью команды ORDER BY сортируем записи по значениям указанных атрибутов. При чём при многочисленной сортировке сначала сортируется по первому параметру, а потом по второму указанному.

1.5. Определить условия отбора («И» и «ИЛИ»). Создать не менее двух запросов.. Создадим 2 запроса, в которых продемонстрируем работы операторов «и» и «или». Чтобы отбирать соответствующие запросы понадобится ключевое слово WHERE, после которого идёт какое-либо условие.

SELECT Работа.[Код работы], Работа.[Код заказчика], Заказчик.[Код заказчика], Заказчик.[ФИО заказчика], Заказчик.Рейтинг

FROM Заказчик INNER JOIN Работа ON Заказчик.[Код заказчика] = Работа.[Код заказчика]

WHERE (((Заказчик.Рейтинг)>9));



SELECT Работа.[Код работы], [Вид работы].[Оплата за час]

FROM [Вид работы] INNER JOIN (Работа INNER JOIN [Вид-работа] ON Работа.[Код работы] = [Вид-работа].[Код работы]) ON [Вид работы].[Код вида] = [Вид-работа].[Код вида]

WHERE ((([Вид работы].[Оплата за час])>800 Or ([Вид работы].[Оплата за час])<400));

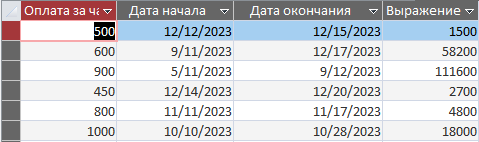


1.7. Создать вычислимые поля.

Пусть нужно вывести названия всех услуг, а также их цены в долларах. Создадим поля, которые будут вычисляться прямо в момент запроса, которые можно именовать удобным для нас и понятным образом, чтобы потом понимать, какие именно данные мы запрашивали. Чтобы именовать новое поле понадобится оператор AS.

SELECT [Вид работы].[Оплата за час], Работа.[Дата начала], Работа.[Дата окончания], ([Дата окончания]-[Дата начала])\*[Оплата за час] AS Выражение1

FROM Работа INNER JOIN ([Вид работы] INNER JOIN [Вид-работа] ON [Вид работы].[Код вида] = [Вид-работа].[Код вида]) ON Работа.[Код работы] = [Вид-работа].[Код работы];



1.8. Создать отсортированный по вычислимому полю запрос из нескольких таблиц,

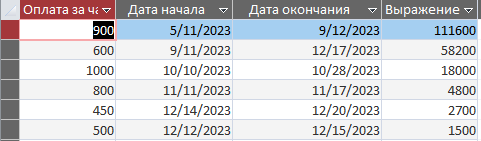
в котором определены условия «И» и «ИЛИ».

В принципе все необходимые команды для выполнения такого запроса мы разобрали, поэтому остаётся грамотно всё подставить и воспользоваться командами.

SELECT [Вид работы].[Оплата за час], Работа.[Дата начала], Работа.[Дата окончания], ([Дата окончания]-[Дата начала])\*[Оплата за час] AS Выражение1

FROM Работа INNER JOIN ([Вид работы] INNER JOIN [Вид-работа] ON [Вид работы].[Код вида] = [Вид-работа].[Код вида]) ON Работа.[Код работы] = [Вид-работа].[Код работы]

ORDER BY ([Дата окончания]-[Дата начала])\*[Оплата за час] DESC;

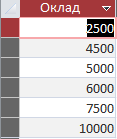


1.9. Использовать предложение DISTINCT.

Пусть нам зачем-либо необходимо знать сколько всего различных услуг.

SELECT DISTINCT Оклад

FROM Сотрудник;



Оператор DISTINCT, собственно, и указывает СУБД вывести все значения без повторений, то есть игнорировать вывод, если такой вывод уже есть, поэтому остаётся написать этот оператор перед атрибутом, по которому вы хотите посмотреть список уникальных значений.

## 2. Внешнее объединение таблиц.

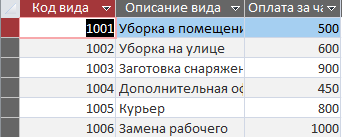
2.1. Продемонстрировать использование предложений Is null, Is not null.

Для этого на время разрешим нулевые значения для ФИО клиента в таблице . Мы добавим запись из предыдущего задания и еще пару таких записей в таблицу с нулевыми значениями клиентов, чтобы посмотреть, как работают данные операторы. По сути IS NULL и IS NOT NULL выводят соответственно записи, у которых значение указанного атрибута пустое или не пустое.

SELECT [Вид работы].[Код вида], [Вид работы].[Описание вида], [Вид работы].[Оплата за час]

FROM [Вид работы]

WHERE ((([Вид работы].[Оплата за час]) Is Not Null));



2.2. Использовать предложение LIKE.

Данный оператор позволяет отобрать записи, со значениями атрибутов, похожими на то, что указано после ключевого слова LIKE. Это может использоваться для выполнения различных задач. Пусть нужно найти услуги, у которых цена составляет 1000 рублей

SELECT Сотрудник.[ФИО сотрудника], Сотрудник.Оклад

FROM Сотрудник

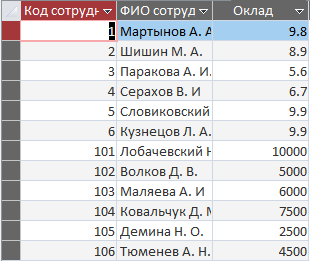
WHERE (((Сотрудник.Оклад) Like 5000));



2.3. Использовать оператор UNION.

Данный оператор применяется для того, чтобы объединить выборку данных из разных таблиц, для этого пишется 2 оператора SELECT, с последующим выбором того, что нужно, а также между ними ставится кодовое слово UNION. Например, нам нужно вывести список абсолютно всех людей с адресами и телефонами, фигурирующих в базе данных

SELECT \* FROM Сотрудник UNION SELECT \* FROM Заказчик;



## 3. Выбор данных с помощью группирующих запросов с условием

3.1. Создать итоговый запрос, содержащий несколько итоговых цифр.

Пусть нам нужно вывести количество деталей с ценой 1000 рублей. Для этого воспользуемся кодовым словом COUNT(\*), который необходимо поименовать некоторым образом и указать из какой таблицы берутся записи и с каким критерием. В нашем случае из таблицы деталей со значением поля Цена в 1000 рублей

SELECT COUNT(\*) AS Количество1000

FROM Сотрудник

WHERE Оклад =10000;



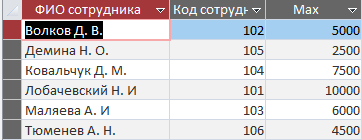
3.2. Создать простой группирующий запрос.

Для этого воспользуемся оператором MAX() внутри скобок нужно указать по какому атрибуту будем искать максимумы, также его нужно переименовать с помощью AS и после мы сгруппируем все записи по заказанным услугам, чтобы всё было читаемо.

SELECT Сотрудник.[ФИО сотрудника], Сотрудник.[Код сотрудника], MAX([Оклад]) AS Mаx

FROM Сотрудник

GROUP BY Сотрудник.[ФИО сотрудника], Сотрудник.[Код сотрудника];



## 4. Выбор данных с помощью подзапросов.

4.1. Создать запрос с выбором при помощи In.

SELECT \*

FROM Сотрудник

WHERE [Оклад] IN (10000, 5000);



При помощи использования подзапроса выведем всю информацию о услугах, с ценой 10000, 5000 , для этого используется ключевое слово IN, после чего в скобках указывается, из каких значений атрибута мы отбираем.

4.2. Использовать предложение ALL

Выведем всю информацию для услуг, цена у которых больше чем все цены из некоторого интервала кода. Для этого в параметре отбора пишем ключевое слово ALL а потом делаем подзапрос, то есть запрос внутри запроса, позволяющий выбрать необходимые записи в таблице и уже потом сравнивать с ними. ALL указывает на то, что сравниваются абсолютно со всеми записями, а не с какой-то конкретной.

SELECT \*

FROM Сотрудник

WHERE Оклад > ALL (SELECT Оклад FROM Сотрудник WHERE Оклад > 1000 AND Оклад < 7000);



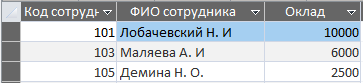
4.3. Использовать предложение ANY.

А теперь давайте выведем записи, похожие на предыдущий запрос, но в которых цена д равна цене хотя бы одной услуги из некоторого списка кодов, так мы, например, можем найти совпадающие цены. Для этого понадобится кодовое слово ANY, указывающее на то, что мы ищем хотя бы одно совпадение, а так, структура запроса останется неизменной, по сравнению с предыдущим.

SELECT \*

FROM Сотрудник

WHERE [Оклад] = ANY (SELECT [Оклад] FROM [Сотрудник] WHERE [Код сотрудника] IN (101,103,105));



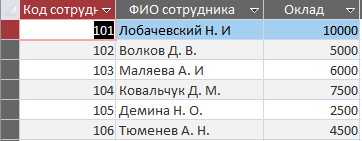
4.4. Использовать предложения EXISTS

Вообще, данное кодовое слово используется для того, чтобы проверить, возвращает ли подзапрос что-нибудь или нет (хотя бы одну строчку), по сути это является логическим типом в базе данных, проверяющий на наличие выбранных записей в ходе запроса. Чтобы показать, как это работает на практике, сначала в подзапросе укажем значение существующего в базе адреса в виде города Омск, чтобы при проверке подзапрос вернул какое-то количество записей и, следовательно, оператор EXISTS вернет истину и основной запрос выполнится, выведя всю информацию о поставщиках в таблице. Во второй же раз мы поставим в проверке записей в подзапросе не существующее в нашей таблице значение города Владивосток. Так как таких городов адресов в таблице нет, оператор не найдёт ни одной записи в ходе подзапроса и следовательно вернёт ложь, после чего основной запрос не будет дальше выполняться.SELECT FROM Клиенты

SELECT \*

FROM Сотрудник

WHERE EXISTS (SELECT \* FROM Сотрудник WHERE [Оклад]= 10000);



# Выводы

Итак, в ходе лабораторной работы к уже построенной базе данных во второй лабораторной работе по модели, созданной в третьей лабораторной работе, был создан список всевозможных запросов, которые применимы к данной базе данных и которые выводят разную полезную информацию. Допустим можно выводить суммы по заказанным деталям, телефоны всех фигурирующих в таблице людей и многое другое. Собственно, полный список был представлен в самой лабораторной работе. Система запросов очень удобна и полезна. Были изучены все основные типы запросов, а также кодовые слова, которые используются при создании этих запросах. При помощи данной работы были получены базовые навыки по созданию запросов, которые пригодятся в дальнейшем при работе с базами данных и запросами, ведь как уже было сказано, это очень удобный функционал. Были попутно выполнены все поставленные задания и осуществлены все цели работы, которые были перечислены в начале работы. Стоит отметить, что возможно запросы, составленные в ходе работы, какие-то из них, могут быть абсурдными и бесполезными, но они нужны для того, чтобы понять весь принцип построения и исполнения запросов, чтобы в дальнейшем удобно ими оперировать.

# Заключение

Базы данных-действительно очень интересный и важный инструмент, позволяющий хранить информацию о множестве сущностей, а также связях между ними. В дальнейшем следует развивать свои знания в этой области и выполнять другие задачи, постепенно создавая свою базу данных.