МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

высшего образования

«КРЫМСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ им. В. И. ВЕРНАДСКОГО»

ФИЗИКО-ТЕХНИЧЕСКИЙ ИНСТИТУТ

Кафедра компьютерной инженерии и моделирования

**НАЗВАНИЕ**

Отчет по лабораторной работе № 1

по дисциплине «Базы данных»

студента 3 курса группы ПИ-212(1)

Кучеренко Александра Сергеевича

Направления подготовки 09.03.04 «Программная инженерия»

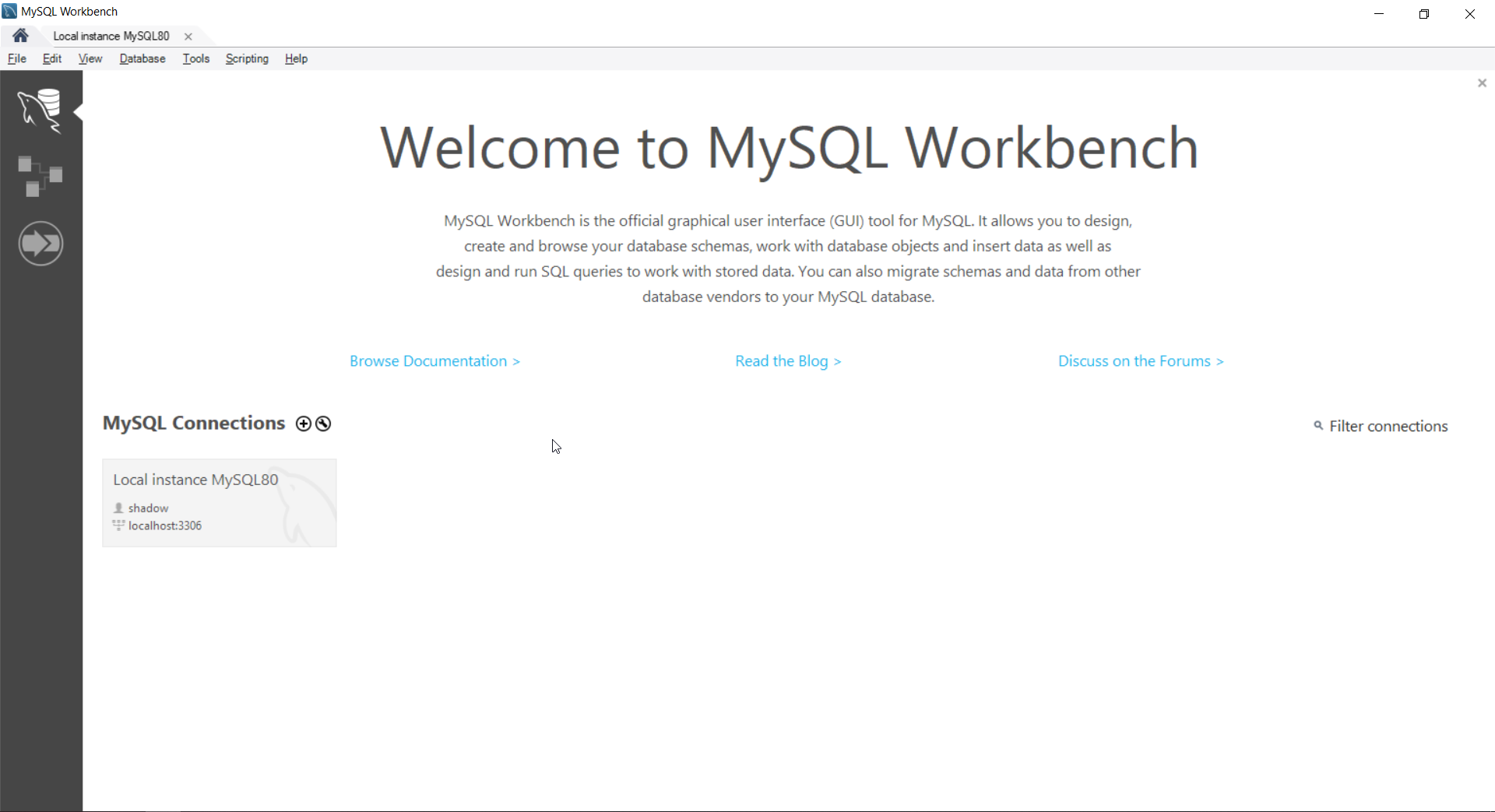
Симферополь, 2023

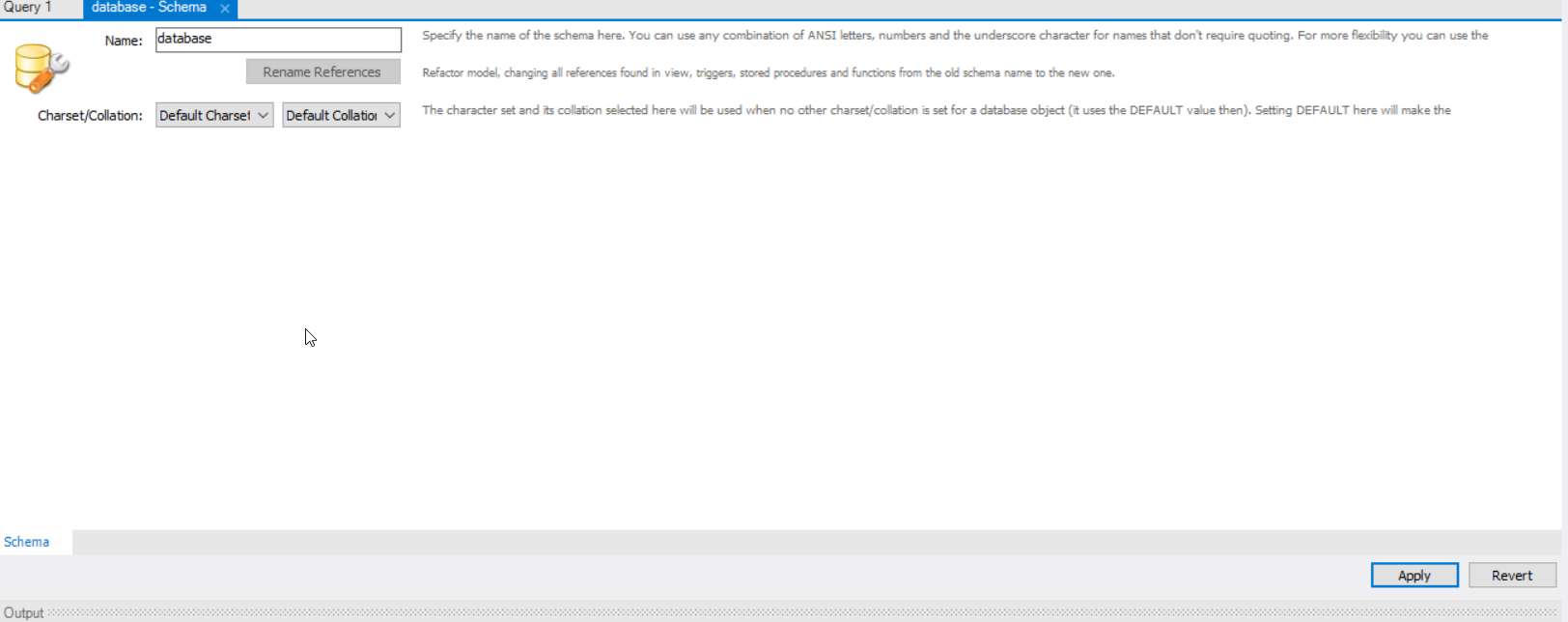
# Тема: "Создание простой базы данных"

**Цель:** ознакомиться с пользовательским интерфейсом СУБД. Научиться создавать таблицы реляционной СУБД в режиме конструктора

**Ход работы:**

**1.1.** Запустите программу для работы с базой данных. Войдите в режим конструктора таблиц

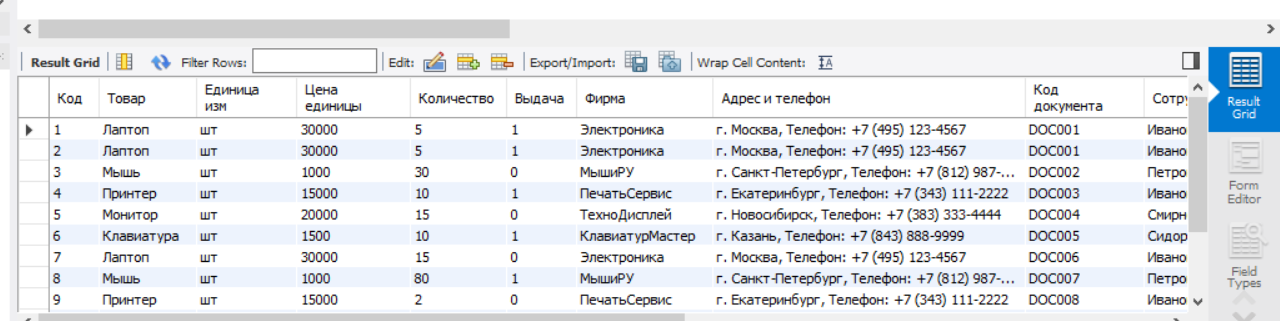
1. Открываю MYSQL Workbench
2. Создаю базу данных для дальнейшего создания таблиц



**1.2.** Создайте таблицу для хранения данных, необходимых для учета продукции, хранящейся на складе. Каждая запись таблицы должна отражать факт приема или выдачи определенного количества товара определенной стоимости фирме-партнеру от определенных сотрудников склада (не более трех человек), по определенному документу. Сохраните созданную таблицу.



**1.3.** Заполните созданную таблицу данными (не менее 10 записей). Данные в столбцах с названиями товаров, фирм, и фамилий сотрудников должны повторяться два-три раза.



**Контрольные вопросы**

1. Оптимально ли хранятся данные в созданной таблице? Обоснуйте ответ.

Нет, их надо разделять на маленькие таблички. С такими таблицами не удобно работать.

2. Сколько различных типов данных можно хранить в таблице используемой СУБД?

5 различных

Числовые, символьные, составные, дата/время, бинарные.

3. Почему набор данных в ключевых полях таблицы должен быть индивидуальным для каждой записи?

Потому что каждый объект в таблице надо как-то идентефицировать

**Отчет по лабораторной работе № 2**

# Тема: "НОРМАЛИЗАЦИЯ БАЗЫ ДАННЫХ"

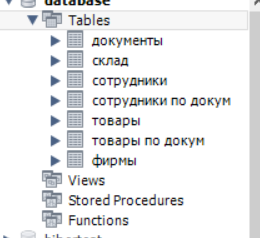
**Цель:** создать нормализованную базу данных, используя БД, созданную в ходе выполнения лабораторной работы №1.

**Ход работы:**

**2.1** Определите, какое количество разных сущностей содержит таблица, созданная на предыдущей лабораторной работе. Создайте в той же базе данных набор таблиц, позволяющих хранить ту же информацию.

Прошлая таблица содержит 6 разных сущностей: документы, сотрудники, сотрудники по документам, товары, товары по документам, фирмы

**2.2** Проверьте, нет ли в получившихся таблицах полей, не входящих в первичный ключ, но при этом содержащих одинаковую по смыслу информацию. Проверьте, нет в таблицах полей, содержащих (в отдельном поле) несколько разных свойств данного объекта. Внесите необходимые исправления.



**2.3** Убедитесь, что данные в не ключевых полях таблиц зависят от всего первичного ключа, а не от его части. Если это не так, внесите необходимые исправления.

**2.4** Убедитесь, что новая база данных не содержит информации, которая может быть легко вычислена. Если это не так, внесите необходимые исправления

**Контрольные вопросы**

1. К каким результатам привела нормализация базы данных?

Удобный доступ и разделение данных

2. Какие существуют практические приемы нормализации? Какие из них можно реализовать в данной СУБД?

3. Какие нормальные формы (НФ) вы применили в ходе выполнения данной работы?

Нормализация по первой нормальной форме (1НФ) предполагает, что каждое поле таблицы БД должно быть неделимым и не содержать повторяющихся групп.

Вторая нормальная форма (2НФ) требует, чтобы все поля таблицы зависели от первичного ключа, то есть чтобы первичный ключ однозначно определял запись и не был избыточным.

Третья нормальная форма (3НФ) требует, чтобы значение любого поля, не входящего в первичный ключ, не зависело от значения другого поля, также не входящего в первичный ключ.

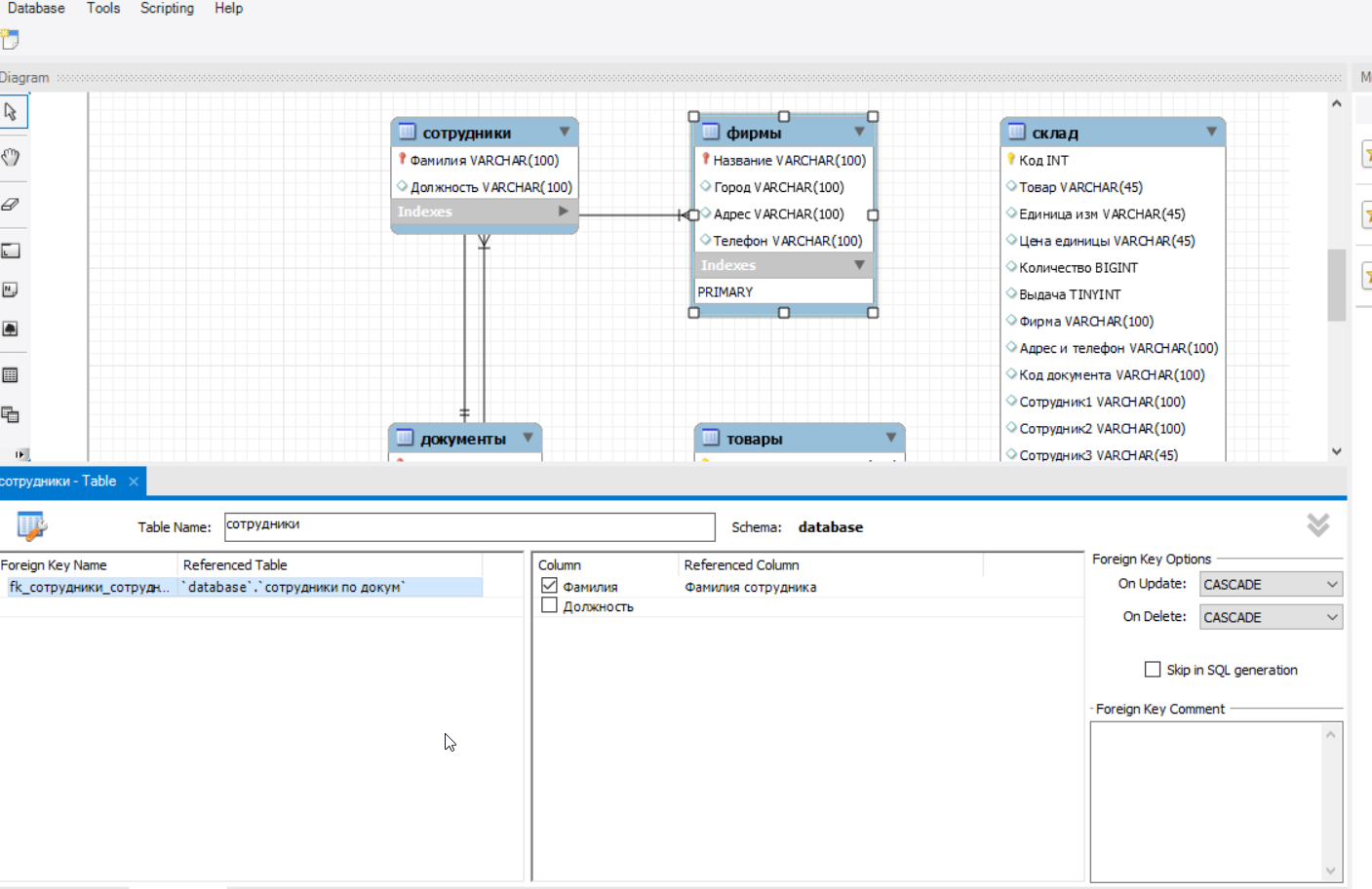
**Отчет по лабораторной работе № 3**

# Тема: "УСТАНОВКА ФОРМАЛЬНЫХ РЕЛЯЦИОННЫХ ОТНОШЕНИЙ МЕЖДУ ТАБЛИЦАМИ. ЗАПОЛНЕНИЕ БАЗЫ ДАННЫХ"

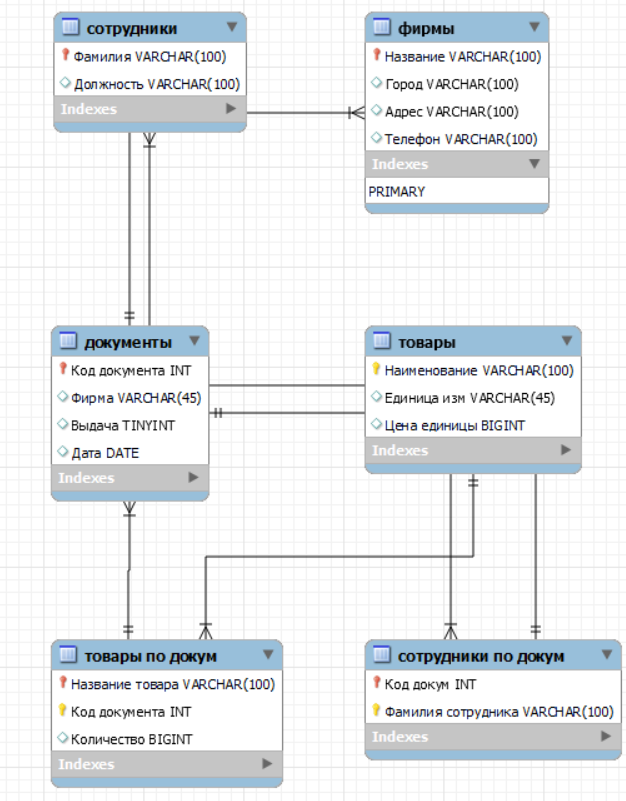
**Цель:** настроить в базе данных, созданной в ходе выполнения лабораторной работы №2, формальные связи между таблицами и соответствующие каскадные изменения данных.

**Ход работы:**

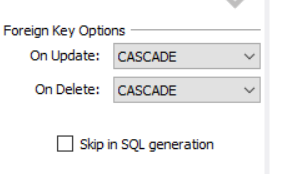
**3.1.** Откройте окно, предназначенное для создания связей между таблицами. Добавьте в это окно образы таблиц.

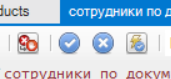


**3.2.** Установите связи между таблицами (определите внешние ключи) согласно логике работы базы данных. Расположите образы таблиц так, чтобы линии связей не перекрывали друг друга.



**3.3.** Задайте правила обработки данных при изменении, добавлении и удалении информации в связанных таблицах. Проверьте корректность работы БД с учетом заданных правил.



Для проверки правил выключаю автокоммит:  


До удаления Козлова.

Таблица "сотрудники по докум":

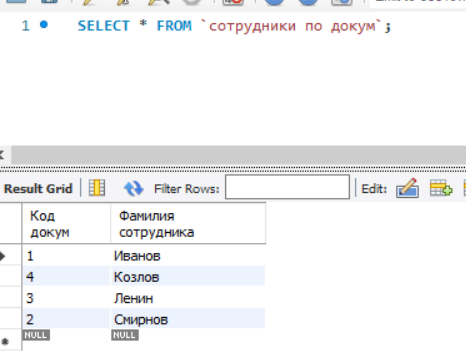
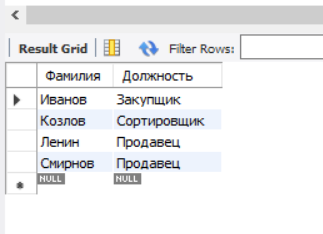


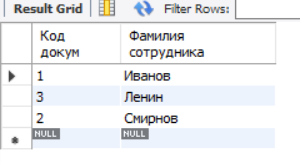
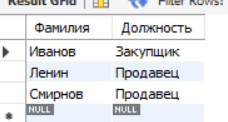
Таблица "сотрудники":



Удаляем Козлова из сотрудников:



Результат:



**Контрольные вопросы:**

1. Какие реляционные отношения между таблицами существуют в вашей базе данных? Сколько отношений каждого вида имеется?

- Один ко многим:

- Между таблицей "товары" и таблицей "товары по докум".

- Количество отношений такого типа: 1.

- Один к одному:

- Между таблицей "товары по докум" и таблицей "документы" (по полю "Код документа").

- Количество отношений такого типа: 1.

- Многие к одному:

- Между таблицей "документы" и таблицей "фирмы" (по полю "Фирма").

- Между таблицей "склад" и таблицей "фирмы" (по полю "Фирма").

- Количество отношений такого типа: 2.

- Многие ко многим:

- Между таблицей "сотрудники" и таблицей "сотрудники по докум".

- Количество отношений такого типа: 1.

2. В каких случаях при попытке изменить данные в дочерней таблице возникнет сообщение об ошибке?

- Если попытаться изменить значение поля, которое является частью первичного ключа в дочерней таблице.

- Если попытаться изменить значение поля, на которое ссылается внешний ключ из родительской таблицы и в дочерней таблице при этом уже есть связанные записи по этому значению.

Если попытаться изменить значение поля "Название товара", которое является частью первичного ключа в таблице "товары по докум".

Если попытаться изменить значение поля "Код документа", которое является частью первичного ключа в таблице "товары по докум".

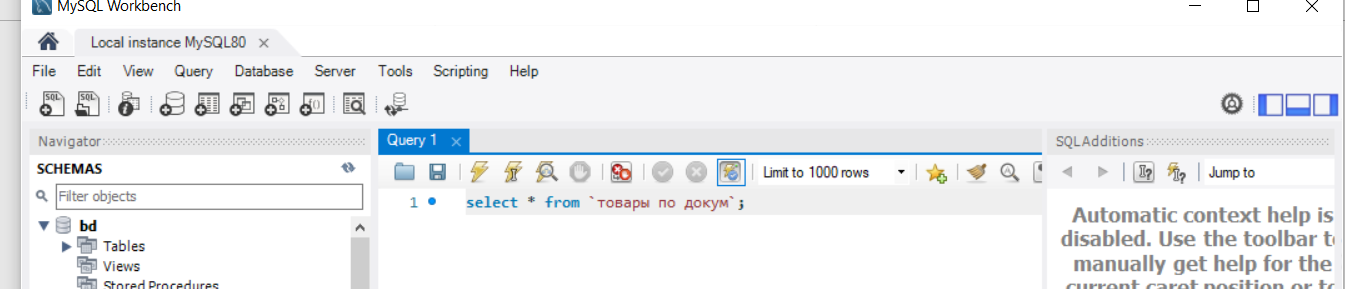
Оба этих поля являются составным первичным ключом таблицы "товары по докум" и связаны с таблицей "товары" по внешнему ключу.

**Отчет по лабораторной работе № 4**

СОЗДАНИЕ ПРОСТЫХ SQL-ЗАПРОСОВ К БАЗЕ ДАННЫХ. СОРТИРОВКА ЗАПИСЕЙ

Цель работы: Изучить среду для создания запросов на языке SQL. Научиться создавать элементарные запросы к таблице, созданной в ходе выполнения лабораторной работы №1. Во время работы должны быть решены задачи 4.1-4.3. Отчет о работе должен содержать тексты созданных SQL-запросов, а также ответы на контрольные вопросы.

4.1 Откройте окно для создания запроса в режиме SQL. Изучите назначения кнопок, имеющихся на панели инструментов. 13

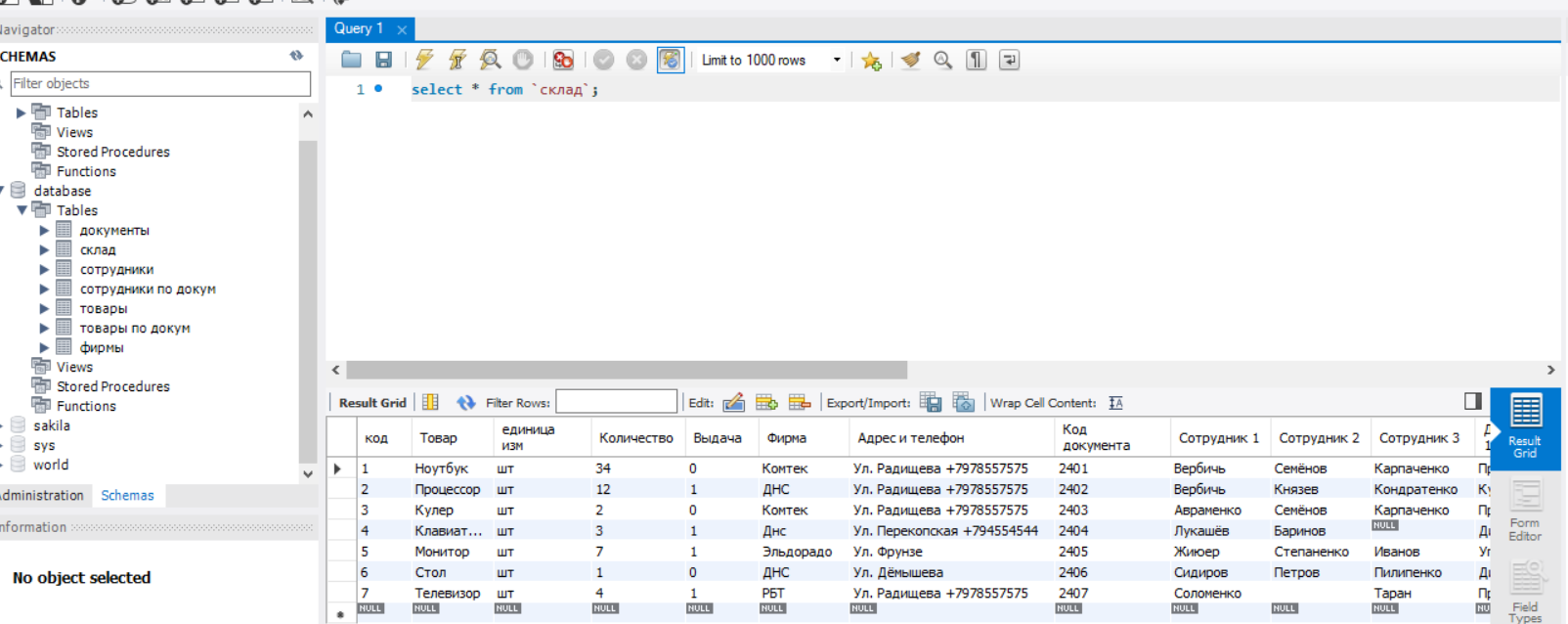


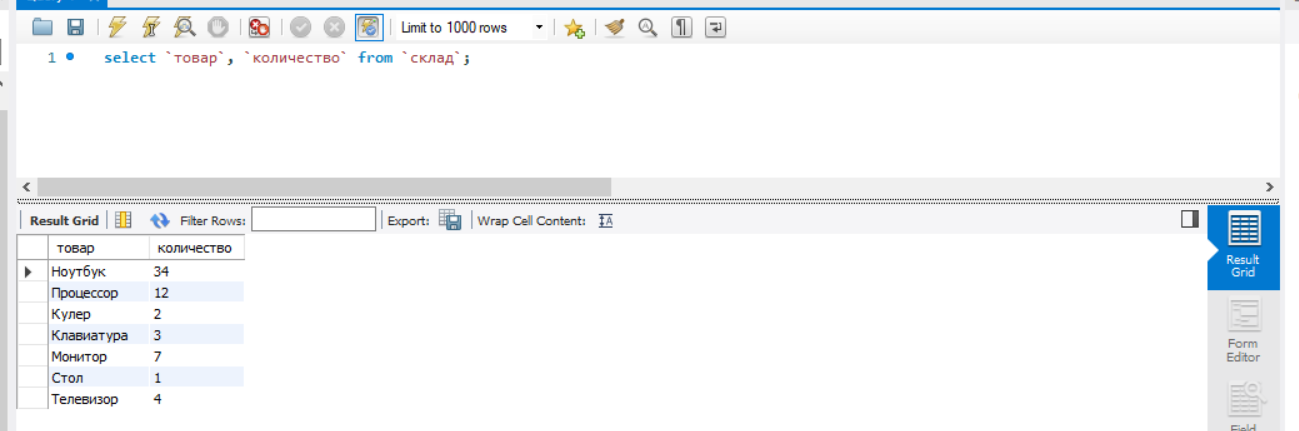
4.2 Создайте простейшей запрос для вывода всей таблицы «Склад», созданной в ходе выполнения лабораторной работы №1. Создайте 2-3 запроса для вывода отдельных полей данной таблицы в разном порядке.

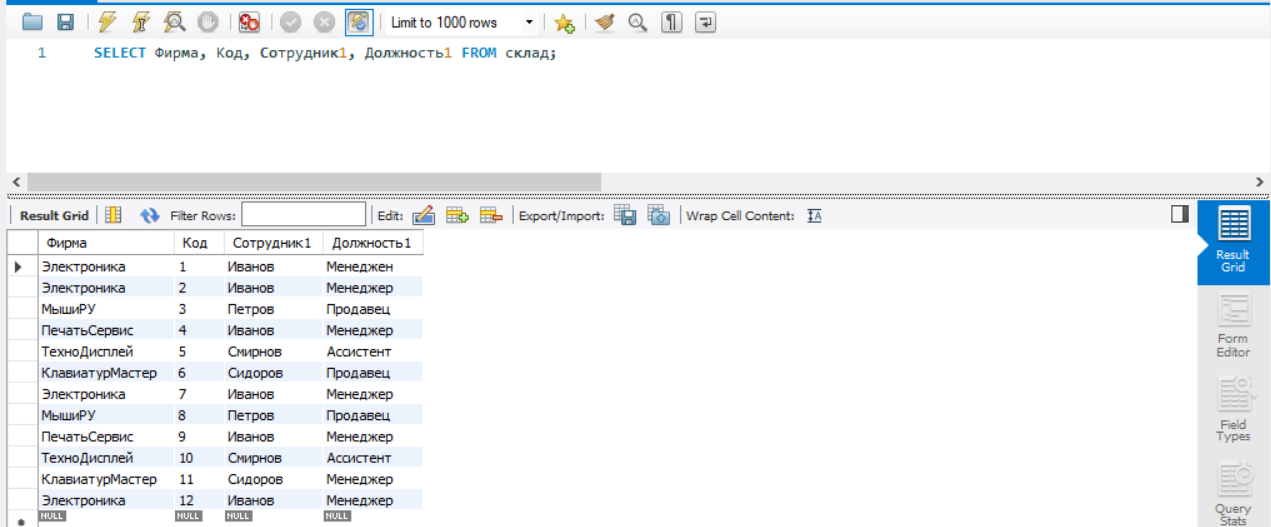
SELECT \* FROM склад;

SELECT Товар, Количество FROM склад;

SELECT Фирма, Код, Сотрудник1, Должность1 FROM склад;







4.3 Создайте 4 запроса с сортировкой данных по возрастанию, по убыванию, вложенной сортировкой по возрастанию и смешанной вложенной сортировкой (убыванию и возрастанию).

1. Запрос с сортировкой данных по возрастанию:

SELECT \* FROM товары ORDER BY `Цена единицы` ASC;

2. Запрос с сортировкой данных по убыванию:

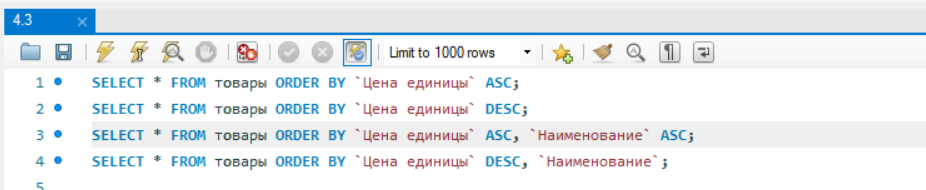
SELECT \* FROM товары ORDER BY `Цена единицы` DESC;

3. Запрос с вложенной сортировкой по возрастанию:

SELECT \* FROM товары ORDER BY `Цена единицы` DESC, `Наименование` DESC;

4. Запрос с смешанной вложенной сортировкой (убыванию и возрастанию):

SELECT \* FROM товары ORDER BY `Цена единицы` DESC, `Наименование`;



**Контрольные вопросы**

**1.** Изменится ли результат запроса типа «SELECT \* FROM Таблица», если в таблице при помощи конструктора изменить порядок следования полей?

Результат запроса «SELECT \* FROM Таблица» не изменится. SQL возвращает значения в соответствии с указанными в запросе именами столбцов, а не с их физическим порядком в таблице.

**2.** Можно ли вывести при помощи запроса одно и то же поле исходной таблицы в двух разных столбцах?

Да, можно вывести одно и то же поле исходной таблицы в двух разных столбцах при помощи запроса. Например:

SELECT `Наименование` as firstcol, `Наименование` as duplicatecol FROM товары;

**3.** Как работает вложенная сортировка в SQL-запросах?

Вложенная сортировка в SQL-запросах работает путем использования подзапросов. Внутренний подзапрос выполняется первым и возвращает отсортированные данные, а затем внешний запрос сортирует эти данные еще раз. **Таким образом**, вложенная сортировка позволяет получить двухуровневую сортировку в результате запроса.

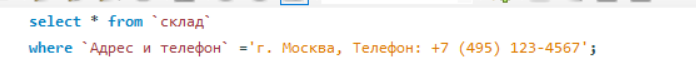
Лабораторная работа № 5

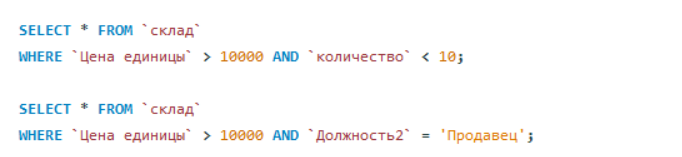
ИСПОЛЬЗОВАНИЕ ПРЕДЛОЖЕНИЯ WHERE, АРИФМЕТИЧЕСКИХ И ЛОГИЧЕСКИХ ОПЕРАТОРОВ

Цель работы: Освоить основные способы фильтрования записей, содержащихся в таблице БД.

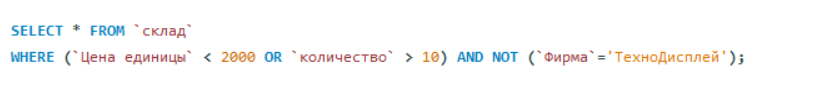
Во время работы должны быть решены задачи 5.1-5.3. Отчет о работе должен содержать тексты созданных SQL-запросов, а также ответы на контрольные вопросы.

5.1 Создайте простейший запрос, содержащий предложение WHERE.



5.2 Создайте не менее двух запросов, содержащих логическое «И», логическое «ИЛИ» и операторы сравнения: =, >, =, <=, < >. Запросы должны включать в себя хотя бы одно сравнение значения поля с константой и хотя бы одно сравнение значения поля со значением другого поля. 

5.3 Создайте запрос, содержащий одновременно логическое «И», логическое «ИЛИ» и логическое «НЕ».



**Контрольные вопросы**

1. Какие типы операторов используются в предложении WHERE?

В предложении WHERE используются следующие типы операторов:

Операторы сравнения, такие как "=", ">", "<", ">=", "<=", "<>", которые используются для сравнения значений полей с константами или другими полями.

Логические операторы, такие как "AND", "OR" и "NOT", которые используются для объединения и инвертирования условий.

2. Что можно сравнивать в предложении WHERE? Можно ли сравнивать значения разных типов данных?

предложении WHERE можно сравнивать значения полей с константами, значения полей между собой (сравнивать значения разных полей), а также применять различные операторы сравнения к ним. В большинстве СУБД можно сравнивать значения разных типов данных, но это может привести к неожиданным результатам, и необходимо быть осторожным при сравнении разных типов данных, так как они могут быть несовместимыми.

3. Можно ли использовать предложение ORDER BY в запросах с WHERE?

Да, предложение ORDER BY можно использовать в запросах с WHERE. Предложение ORDER BY используется для упорядочивания результатов запроса по определенным полям. Вы можете указать поле, по которому нужно упорядочить результаты, и порядок сортировки (ASC - по возрастанию, DESC - по убыванию). Это позволяет управлять порядком, в котором строки будут возвращены в результате запроса.

4. Функционируют ли в вашем интерпретаторе языка SQL операторы !=, !. Если нет, то какое сообщение выводится на экран в случае попытки выполнения запроса?

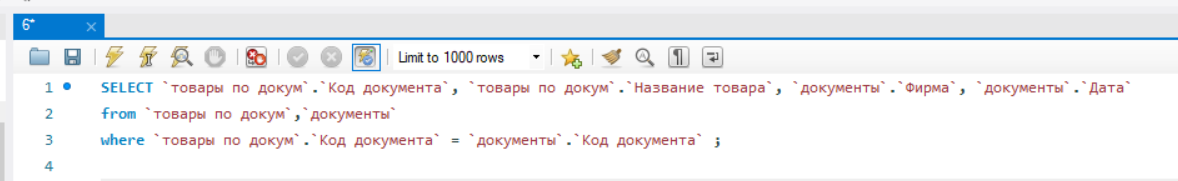
В языке SQL, обычно используется оператор "<>" для проверки неравенства (не равно). Оператор "!" в контексте сравнения обычно не используется. Если в запросе используется несуществующий оператор или синтаксическая ошибка, то большинство СУБД выдаст сообщение об ошибке, указывающее на проблему в запросе. Вот пример сообщения об ошибке, если использовать несуществующий оператор:

Лабораторная работа № 6

ВНУТРЕННИЕ СОЕДИНЕНИЯ ТАБЛИЦ

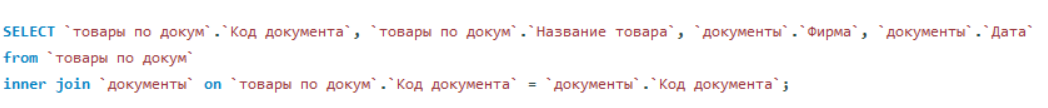
Цель работы: Изучить способы и приемы формирования SQLзапросов, получающих данные одновременно из нескольких таблиц. Во время работы должны быть решены задачи 6.1-6.2. Отчет о работе должен содержать тексты созданных SQL-запросов, а также ответы на контрольные вопросы.

6.1 Построить запрос к двум таблицам, соединенным при помощи предложения WHERE.



6.2 Построить запрос, аналогичный предыдущему, соединяющей таблицы при помощи оператора

INNER JOIN.



Контрольные вопросы

1. Как будет осуществляться внутреннее соединение таблиц, если одной записи таблицы А соответствует, согласно условию отбора, несколько записей таблица Б, а одной записи таблицы Б – несколько записей таблицы А?

Если одной записи таблицы A соответствует несколько записей таблицы B, и одной записи таблицы B соответствует несколько записей таблицы A, то при выполнении внутреннего соединения (INNER JOIN) между этими таблицами будут возвращены все возможные комбинации строк, удовлетворяющие условию соединения. Это означает, что для каждой записи из таблицы A, которая соответствует нескольким записям из таблицы B, будет создана строка результата для каждой из этих соответствующих записей таблицы B, и наоборот. В результате может получиться довольно большой объем данных, и это следует учитывать при выполнении запроса.

2. Как будет осуществляться внутреннее соединение таблиц, если одной записи таблицы А соответствует, согласно условию отбора, не соответствует ни одной записи таблицы Б?

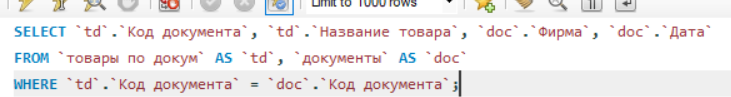
Если одной записи таблицы A не соответствует ни одной записи таблицы B согласно условию отбора (нулевое соответствие), то в результате внутреннего соединения не будет включено ни одной строки, связанной с этой записью из таблицы A. Внутреннее соединение возвращает только те строки, которые имеют соответствие в обоих таблицах согласно условию соединения. Если запись из одной таблицы не имеет соответствия в другой таблице, она исключается из результата.

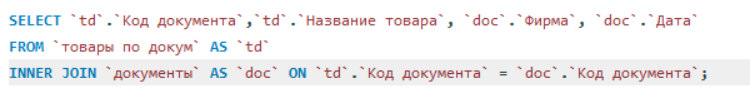
Лабораторная работа № 7

ИСПОЛЬЗОВАНИЕ ПСЕВДОНИМОВ ПОЛЕЙ И ТАБЛИЦ

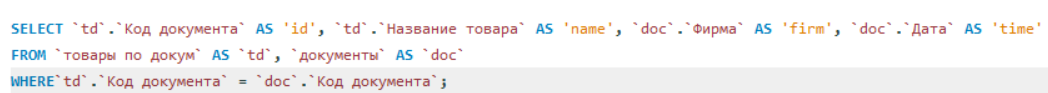
Цель работы: Научиться использовать на практике псевдонимы полей и таблиц в языке SQL. Во время работы должны быть решены задачи 7.1-7.3. Отчет о работе должен содержать тексты созданных SQL-запросов, а также ответы на контрольные вопросы.

7.1 Создайте два запроса с внутренним соединением таблиц, аналогичные запросам из предыдущей лабораторной работы, но использующие вместо имен таблиц их псевдонимы.





7.2 Модифицируйте запросы так, чтобы вместо названий полей в выводимых наборах данных содержались их псевдонимы.



7.3 Напишите запрос, который производит внутреннее соединение таблицы с этой же таблицей.

Контрольные вопросы

1. Какие преимущества дают псевдонимы полей и таблиц?

Улучшение читаемости: Псевдонимы позволяют задать более понятные и информативные имена полям и таблицам, что делает SQL-запросы более понятными для разработчиков и обслуживающего персонала.

Устранение неоднозначности: Если в SQL-запросе используются несколько таблиц с одинаковыми именами полей, псевдонимы позволяют явно указать, к какой таблице относится то или иное поле, и таким образом устранить неоднозначность.

Улучшение безопасности: Использование псевдонимов может помочь скрыть фактические имена таблиц и полей от конечных пользователей или посторонних лиц, улучшая безопасность базы данных.

2. Возможны ли SQL-запросы, в которых нельзя обойтись без использования псевдонима таблицы?

SQL-запросы, в которых нельзя обойтись без использования псевдонима таблицы:

Самосоединение: Когда вы выполняете самосоединение таблицы, то есть объединение таблицы с самой собой, псевдонимы обязательно нужны, чтобы различить две копии таблицы. Например, для поиска пар сотрудников, работающих в одном и том же отделе, вы используете два разных псевдонима для одной и той же таблицы.

Сложные подзапросы: В некоторых случаях, когда вы используете подзапросы в запросе, псевдонимы таблиц могут сделать запрос более ясным. Например, при использовании вложенных подзапросов, псевдонимы могут помочь указать, к какой таблице относятся поля во вложенных подзапросах.

Извлечение данных из одной таблицы в несколько раз: Если вам нужно извлекать данные из одной таблицы несколько раз, используя разные условия, псевдонимы таблиц позволяют создать несколько копий таблицы и ссылаться на них отдельно.

Лабораторная работа № 8

ВНЕШНИЕ СОЕДИНЕНИЯ ТАБЛИЦ

Цель работы: Освоить способы внешнего соединения таблиц. Улучшить понимание принципов внутреннего и внешнего соединения. Во время работы должны быть решены задачи 8.1-8.3. Отчет о работе должен содержать тексты созданных SQL-запросов, а также ответы на контрольные вопросы.

8.1 Создайте запросы к вашей базе данных, использующие левое (LEFT JOIN) и правое (RIGHT JOIN) соединения таблиц.

8.2 Создайте запрос, использующий внешнее соединение трех разных таблиц. Все таблицы должны быть заданы псевдонимами.

8.3 Создайте запрос к двум таблицам без указания способа соединения.

Контрольные вопросы

1. Предположим, что мы создали запрос: SELECT A.\*, B.\* FROM A LEFT JOIN В ON... В таблице A – сто записей. Количество записей в таблице B – неизвестно. Может ли наборе данных, полученном в результате выполнения запроса быть: а) 0 записей; б) 90 записей; с) 100 записей; д) 120 записей.

2. По какому принципу происходит объединение данных в таблицах, если способ соединения не задан?