# введение

### Тема курсовой работы:

Разработать базу данных ателье, реализующую следующие процессы: прием заказов от клиентов и распределение их по мастерам с учетом их квалификации или выполняемых функций, учет услуг ателье и учет цен на услуги, хранение информации о сотрудниках и клиентах ателье. Разработать запросы, перечень которых указан в разделе исходных данных по курсовой. Разработать хранимые процедуры: внесение данных о новом сотруднике ателье, внесение новой услуги, проверка наличия заказа конкретного клиента, подсчет суммы заказа и общей суммы, если у конкретного клиента несколько заказов, изменение стоимости конкретной услуги, изменение даты заказа (например, отсрочка заказа на неделю при определенных условиях).

Цель курсовой работы:

Спроектировать базу данных склада аптечных товаров на основе концептуальной, логической и физической модели. В качестве инструмента проектирования выбрать CASE-средство. Разработать базу данных на основе СУБД MySQL.

# Задачи курсовой работы:

* проанализировать и описать предметную область БД;
* разработать концептуальную модель базы данных с помощью CASE инструментария;
* на основе концептуальной модели, сгенерировать логическую и физическую модель базы данных с помощью CASE-инструментария;
* сгенерировать SQL-скрипт описания физической модели базы данных в MySQL;
* проверить правильность сгенерированного скрипта, проиллюстрировать диаграмму созданной базы данных;
* разработать запросы базы данных;
* разработать хранимые процедуры базы данных с входными и выходными параметрами;
* разработать триггеры базы данных;
* заполнить таблицы данными и подготовить к тестированию.

# ПОСТАНОВКА ЗАДАЧИ

Курсовая работа является практической разработкой, которая позволит получить навык применения методологического и инструментального аппарата для построения информационных систем на основе реляционных баз данных для решения задач в области экономики и менеджмента.

Курсовая работа формирует системный подход в освоении информационных технологий и выполняется как разработка информационного продукта в соответствии с его жизненным циклом. Работа должна последовательно реализовывать следующие этапы жизненного цикла:

1. этап анализа и описания предметной области БД;
2. этап постановки задачи и формализованного описания требований к БД;
3. этап разработки модели данных БД;
4. этап проектирования базы данных (БД);
5. этап разработки хранимых процедур и триггеров для БД;
6. этап документирования БД (пояснительная записка к курсовой работе).

# МОДЕЛИРОВАНИЕ ДАННЫХ

### **Анализ предметной области. Создание объектов базы данных**.

Перед началом проектирование концептуальной модели базы данных, необходимо продумать объекты, из которых она будет состоять. Предметная область данной курсовой работы – разработка базы данных для склада аптечных товаров. При анализе предметной области формируем сущности необходимые нам для создания БД: материалы, тип\_одежды, мастера, клиенты, заказы, работы, размеры, скидки. Для разработки базы данных использовалась MySQL WORKBENCH 8 CE. При создании объектов базы данных необходимо задать корректно имена сущностей и атрибутов с учетом ограничений MySQL. Коды объектов (Code) соответствуют идентификаторам объектов в физической модели и поэтому задаются с учетом ограничений: набор символов, содержащих латинские буквы, цифры и знаки подчеркивания и не содержащий недопустимых символов, длиной не более 128.

Ниже, в таблицах 1-8, представлены все созданные сущности с его атрибутами, жирным шрифтом обозначен атрибут, который планируется сделать ключевым.

Таблица . Сущность материалы

|  |  |  |
| --- | --- | --- |
| метериалы | | |
| **код** | количество | качество |
| название | цена |  |

Таблица . Сущность Тип одежды

|  |  |  |
| --- | --- | --- |
| Тип\_одежды | | |
| **код** | тип | сложность |

Таблица . мастера

|  |  |  |
| --- | --- | --- |
| мастера | | |
| **код** | фамилия | имя |
| отчество | Уровень\_мастера | Дата\_начала |

Таблица . Сущность Клиенты

|  |  |  |
| --- | --- | --- |
| клиенты | | |
| **код** | фамилия | имя |
| отчество | Номер\_телефона |

Таблица . Сущность заказ

|  |  |
| --- | --- |
| заказы | |
| **код** | трудоёмкость |
| Уровень\_качества |

Таблица . Сущность работа

|  |  |
| --- | --- |
| работа | |
| **код** | Дата\_начала |

Таблица . Сущность размеры

|  |  |
| --- | --- |
| размеры | |
| **код** | наименование |

Таблица . Сущность скидки

|  |  |  |
| --- | --- | --- |
| скидки | | |
| **код** | наименование | скидка |

Названия кодов и типов данных атрибутов модели “Ателья” приведена в таблице 9.

Таблица

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Атрибут | Тип | Длина |
| Код материала | код | int |  |
| количество материала | количество | int |  |
| качество материала | качество | Varchar(45) | 45 |
| название материала | название | Varchar(45) | 45 |
| Цена материала | цена | double |  |
| Код типа | Код | int |  |
| Наименование типа | тип | Varchar(45) | 45 |
| Сложность типа | сложность | Varchar(45) | 45 |
| код мастера | код | int |  |
| фамилия мастера | фамилия | Varchar(45) | 45 |
| имя мастера | имя | Varchar(45) | 45 |
| отчество мастера | отчество | Varchar(45) | 45 |
| Уровень мастера | Уровень\_мастера | int |  |
| Дата начала работы | Дата\_начала | Date |  |
| Код клиента | Код\_клиента | int |  |
| фамилия клиента | фамилия | Varchar(45) | 45 |
| имя клиента | имя | Varchar(45) | 45 |
| отчество клиента | отчество | Varchar(45) | 45 |
| Номер телефона клиента | Номер\_телефона | Varchar(11) | 11 |
| Код заказа | Код | int |  |
| Трудоемкость заказа | трудоёмкость | int |  |
| Код работы | код | int |  |
| Дата начала работы над заказом | Дата\_начала | date |  |
| Код размера одежды | код | int |  |
| Рамер одежды | размер | int |  |
| Код скидки | код | int |  |
| Название скидки | название | Varchar(45) | 45 |
| Скидка | скидка | double |  |

### Создание концептуальной модели

В ходе работы с web-приложением DRAW.IO была создана концептуальная модель будущей базы данных. Модель состоит из 8 сущностей и промежуточной таблицы. Концептуальная модель дает общее представление о предметной области. В концептуальной модели находят свое отражение все актуальные требования предметной области на стадии проектирования, также на этом этапе определяются задачи, которые будет решать пользователь с помощью базы данных. На рис. 1 представлена концептуальная модель. Концептуальная модель сохраняется в виде файла в формате .jpg.

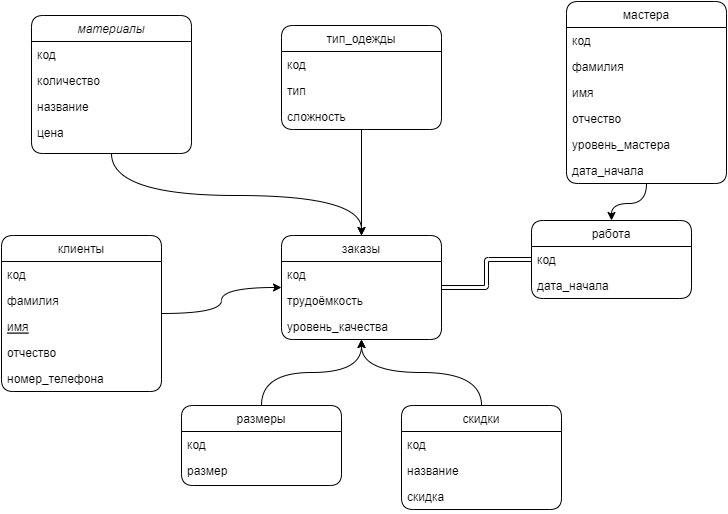


рис. . Концептуальная модель

### Физическая модель

Физическая модель БД определяет способ размещения данных в среде хранения и способы доступа к этим данным, которые поддерживаются на физическом уровне. Исторически первыми системами хранения и доступа были файловые структуры и системы управления файлами (СУФ), которые фактически являлись частью операционных систем. СУБД создавала над этими файловыми моделями свою надстройку, которая позволяла организовать всю совокупность файлов таким образом, чтобы она выглядела как единое целое и получала централизованное управление от СУБД. Однако непосредственный доступ осуществлялся на уровне файловых команд, которые СУБД использовала при манипуляции всеми файлами, составляющие хранимые данные одной или нескольких баз данных.

Физическую схему базы данных создаётся в MySQL Workbench. Физические модели определяют способы размещения данных в среде хранения и способы доступа к этим данным, которые поддерживаются на физическом уровне. На рис. 2 представлен скриншот внутренней среды разработки модели.

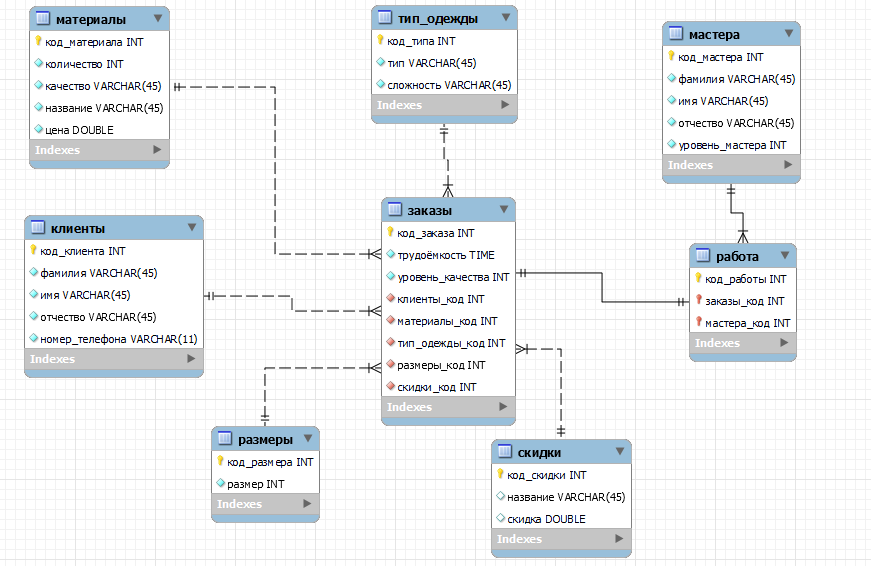


рис.

### Связи между таблицами

Все связи в физической модели идентифицирующие, т.к. экземпляр сущности-потомка однозначно определяется своей связью (отношением) с сущностью-родителем. В таком случае связь каждый экземпляр подчиненной сущности идентифицируется значениями атрибутов родительской сущности. Это означает, что экземпляр подчиненной сущности зависит от родительской сущности и не может существовать без экземпляра родительской сущности. В идентифицирующем отношении единственный экземпляр родительской сущности связан с множеством экземпляров подчиненной.

В таблице 10 представлены вся связи.

Для того чтобы создать связь «многие ко многим», создаётся промежуточная таблица.

Таблица

|  |  |
| --- | --- |
| **сущности** | **связь** |
| материалы - заказы | Один - ко многим |
| Тип\_одежды - заказы | Один - ко многим |
| клиенты - заказы | Один - ко многим |
| размеры - заказы | Один - ко многим |
| скидки - заказы | Один - ко многим |
| работа - заказы | Один – к одному |
| мастера - работа | Один - ко многим |

### Выгрузка на сервер

Для того, чтобы загрузить на сервер, поэтапно выполняются следующе действия, которые показаны на рисунках 3 - 6.

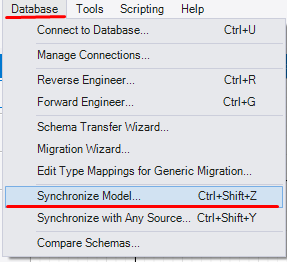


рис.

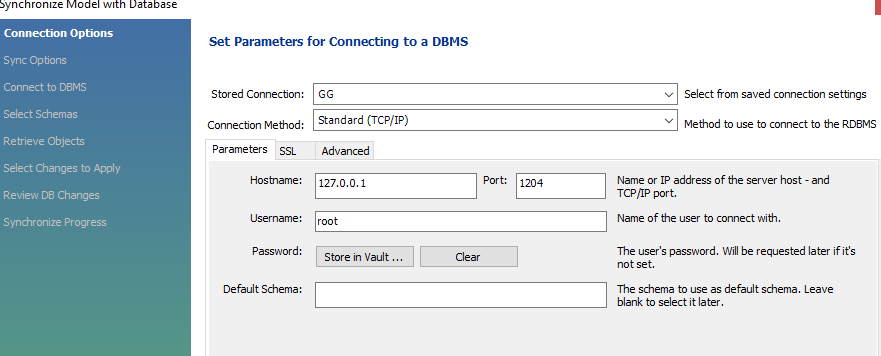


рис.

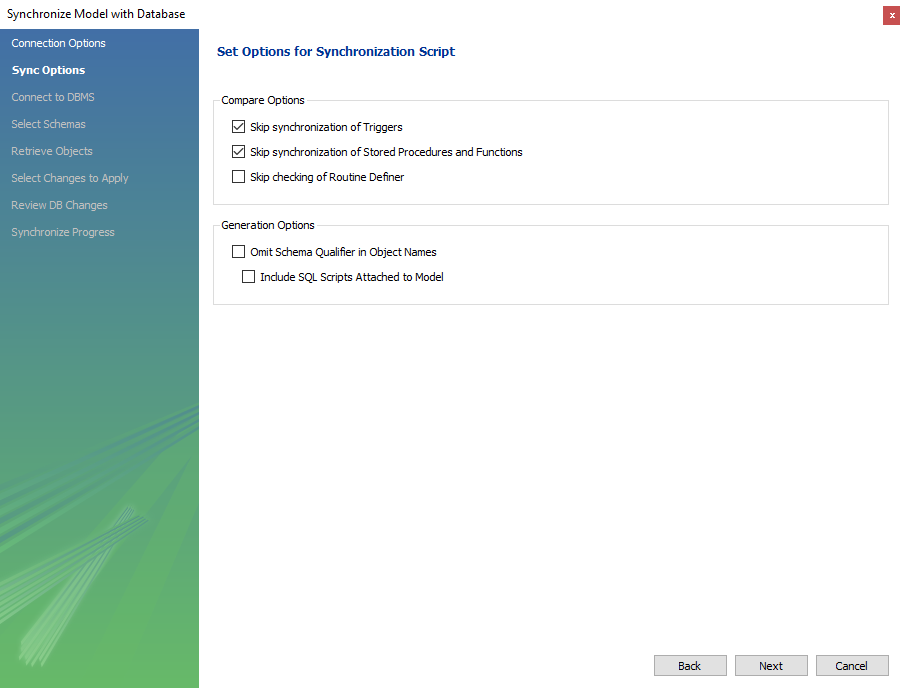


рис.

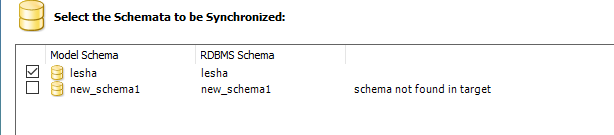


рис.

Далее проверяется на сервере рис.7 .

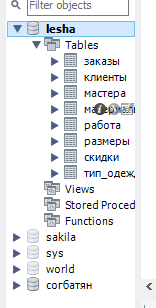


рис.

### Заполнение данных

На рисунках 8 -16 показаны все данные всех таблиц.

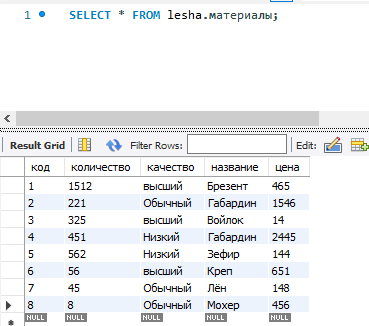


рис. Таблица материалы

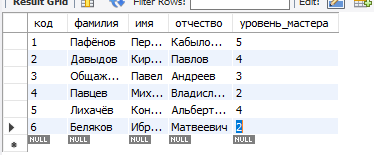


рис. Таблица мастера

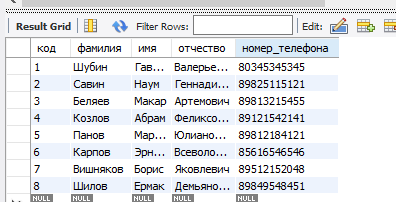


рис. Таблица клиенты

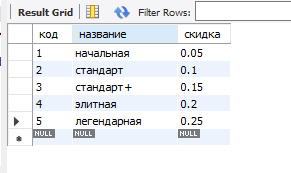


рис. Таблица скидки

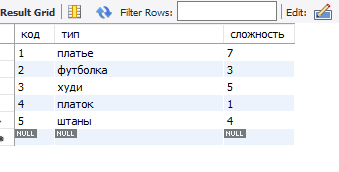


рис. Таблица тип\_одежды

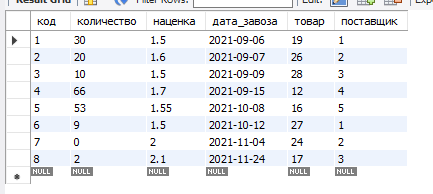


рис. Таблица склад

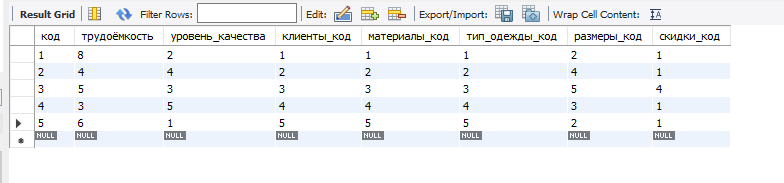


рис. Таблица сотрудник

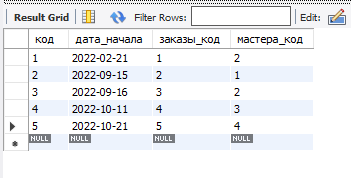


рис. Таблица работы

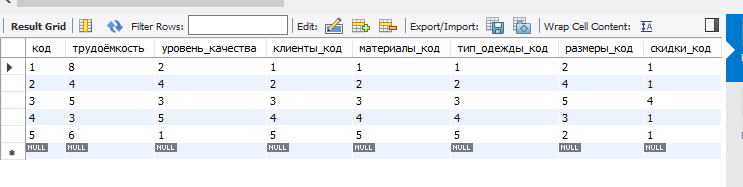


рис. Таблица товары

# запросы

На рисунках 19 - \_ представлены запросы на выборку.

1.Запрос показывает все заказы со всеми полями. Рис.17.

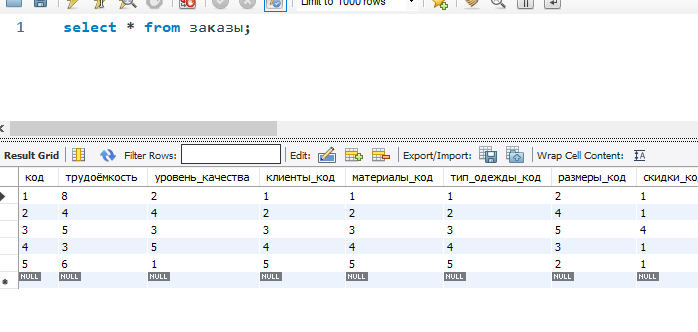


рис. все заказы

2. Запрос, показывающий фамилию с высокой заработной платой. Рис.18.

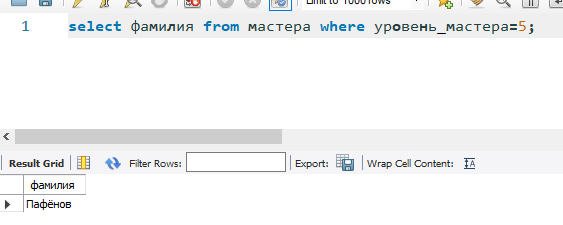


рис. мастера с определенным уровнем

3.Запрос показывает стаж мастеров. Рис.19.

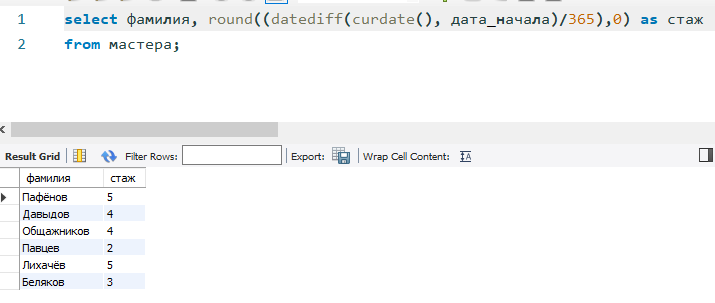


рис. стаж мастеров

4. Данный запрос ФИО клиентов. Рис.20.

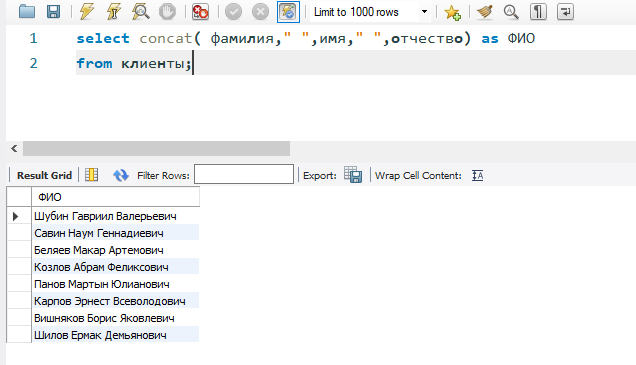


рис. ФИО клиентов

5.Вывод код работы, даты и код заказа у которых уровень мастера равен 3. Рис.21.

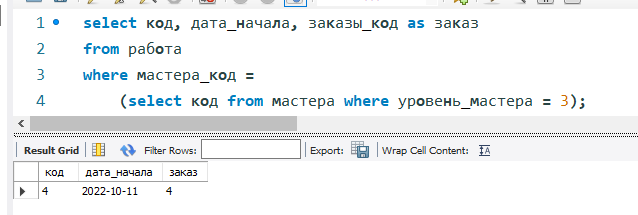


рис. код работы, даты и код заказа у которых уровень мастера равен 3

6. Итоговые цены заказов. Рис.22.

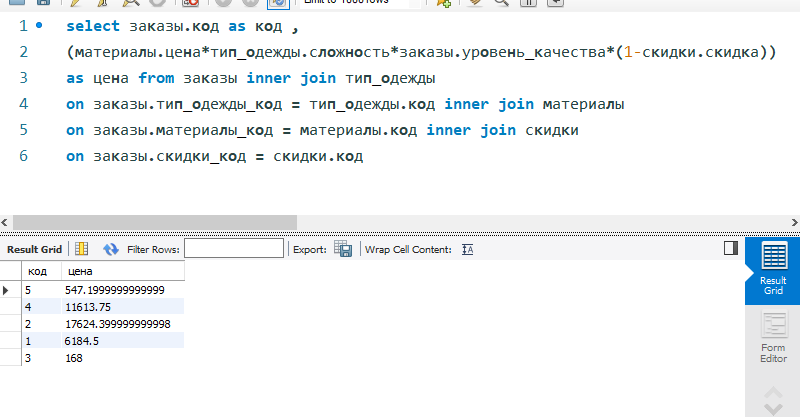


рис. Итоговые цены заказов

7. Запрос выводит количество всех заказов. Рис.23.

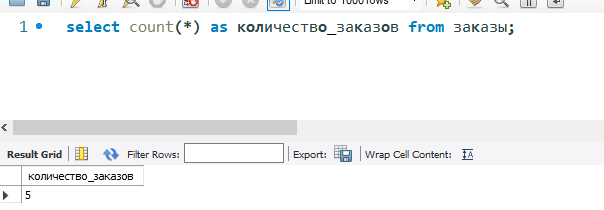


рис. количество всех заказов

8. Сумма всех заказов. Рис.24.

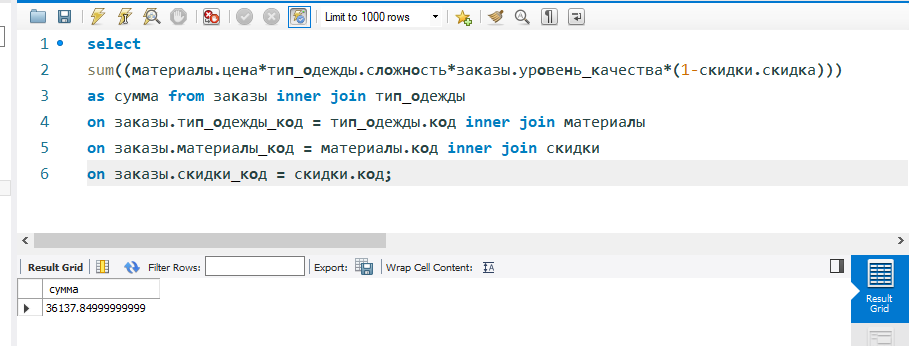


рис. Сумма всех заказов

9. Запрос выводит все мастеров, которые начинаются на «П». Рис.25.

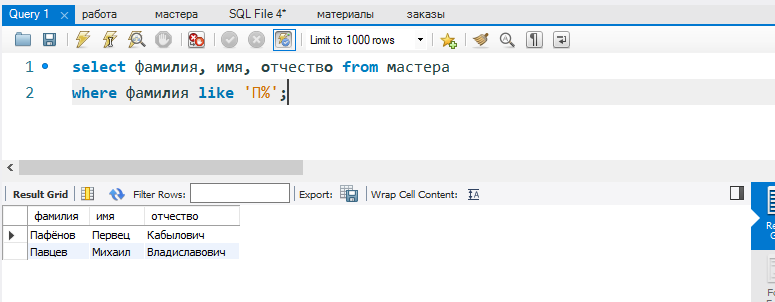


рис.

10. Вывод материалов в заказах Рис.26.

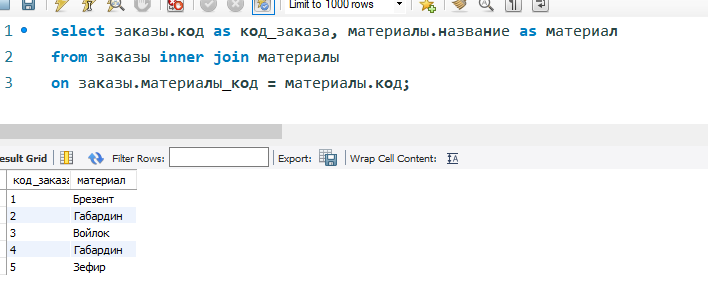


рис. Вывод материалов в заказах

11. Размер скидки. Рис.27.

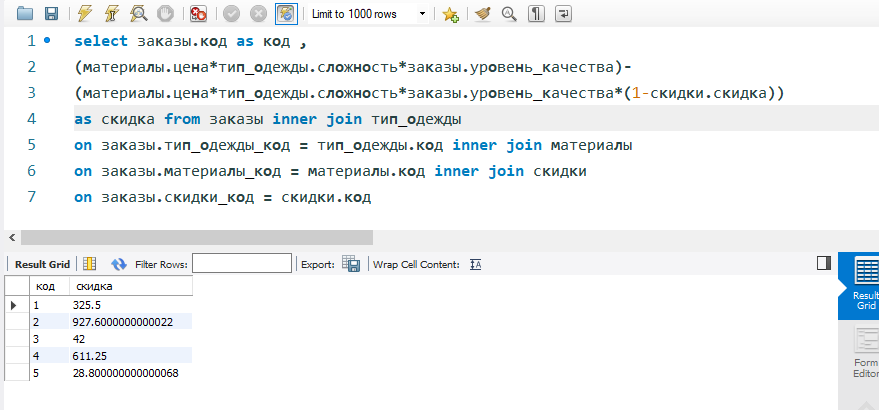


рис. Размер скидки

# 4. ПРОЦЕДУРЫ

Итак, хранимые процедуры в SQL — это аналог функций в других языках программирования. Хранимые процедуры могут выполнять действия над данными автоматически: вывод данных, удаление, изменение — то, что мы делали вручную.

Хранимые процедуры в СУБД MySQL можно создать несколькими способами. В данной работе все процедуры создаются с sql консоли, Также как и все запросы SELECT, INSERT и т.д.

1.Создание процедуры, которая вносит нового мастераРис.28.

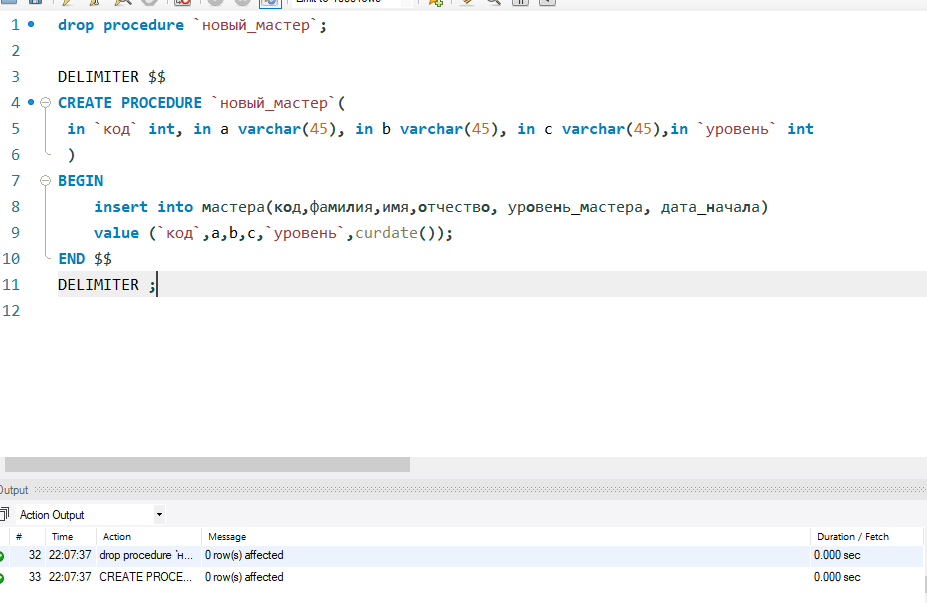


рис. новый мастер

Проверка процедуры Рисунки 29-31

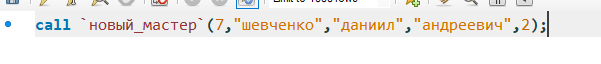


рис.



рис.

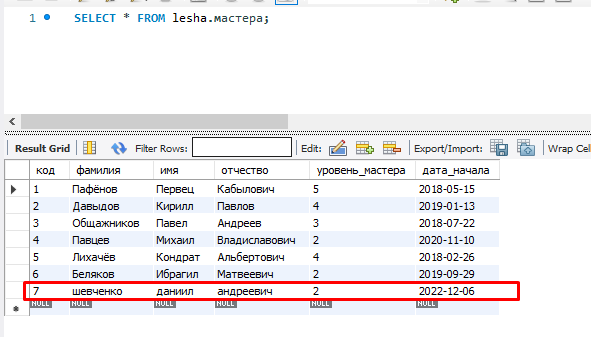


рис.

1. Процедура на создание внесение новых данных о типе одежде. Рис.32

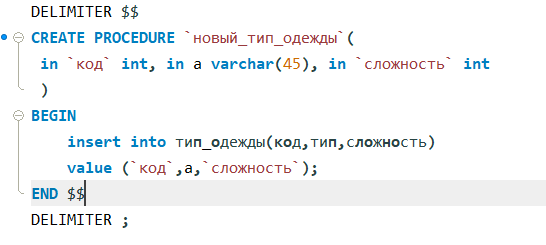


рис. новый тип одежды

Проверка процедуры. Рисунки 33 - 35.



рис.



рис.

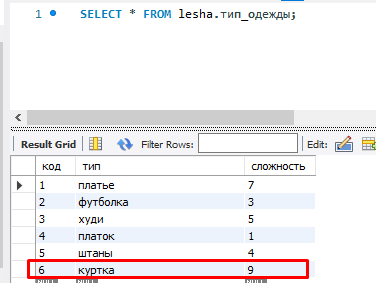


рис.

3.Процедура, выводящая количество заказов у определенного клиента. Рис36.

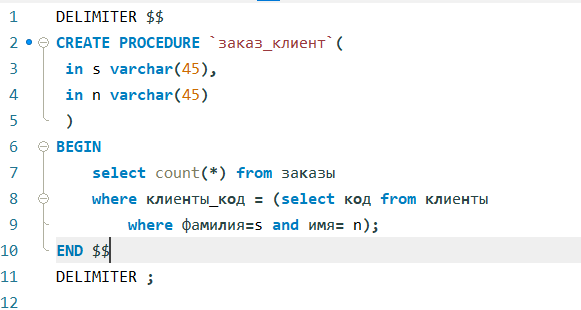


рис. Кол-во заказов

Проверка процедуры. Рисунки 37 - 38.

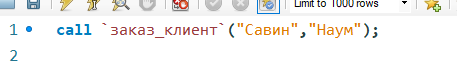


рис.

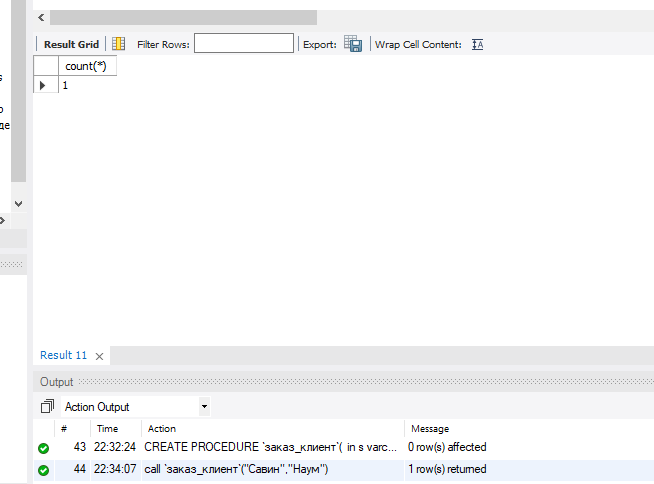


рис.

1. Процедура вычисления суммы заказов у определенного клиента. Рис.39

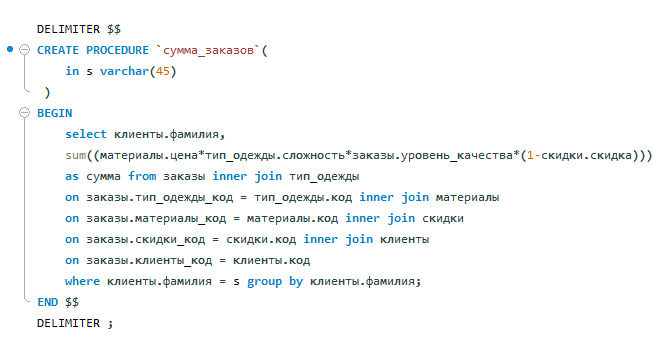


рис.

Проверка процедуры. Рис. 40.

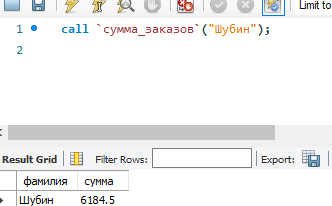


рис.

1. Процедура, которая обновляет цену материала. Рис.41

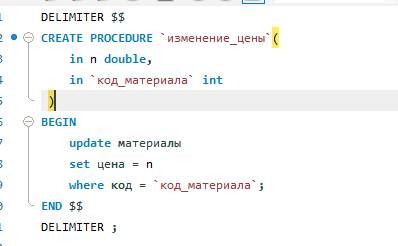


рис.

Проверка процедуры. Рис.42.

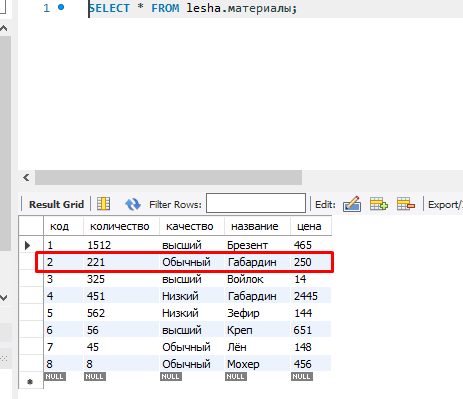


рис.

# Функции

В MySQL функция - это хранимая программа, в которую вы можете передавать параметры и возвращать значение.

Функция отличается от процедуры. Таблица 11

Таблица

|  |  |
| --- | --- |
| Функция | Процедура |
| Функция должна возвращать значение | Хранимая процедура может как возвращать, так и не возвращать значение. |
| Функции не могут возвращать несколько результирующих наборов. | Хранимая процедура может сформировать и вернуть несколько результирующих наборов данных. |
| Функции можно использовать в операторе SELECT. | Процедуры нельзя использовать в операторе SELECT и во всех его секциях (WHERE, JOIN, HAVING и т.д.), так как процедуры вызываются с помощью команды EXECUTE или EXEC. |
| Из функции нельзя вызвать хранимые процедуры. | В хранимых процедурах можно вызывать и функции, и другие хранимые процедуры. |
| В функциях можно использовать только табличные переменные, временные таблицы использовать не получится. | В хранимых процедурах можно использовать как табличные переменные, так и временные таблицы |
| В функциях можно использовать только входные параметры. | В хранимых процедурах можно использовать как входные, так и выходные параметры. |
| В функциях можно использовать только оператор SELECT на выборку данных.  Операторы DML (INSERT, UPDATE, DELETE) для модификации данных использовать нельзя. | В хранимых процедурах можно использовать оператор SELECT, а также операторы DML (INSERT, UPDATE, DELETE) для модификации данных. |

1. Функция возвращает количество работ у мастера. Рис.43

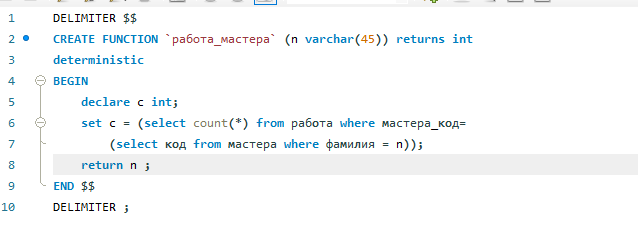


рис. количество работ у мастера

Проверка функции рис. 44.

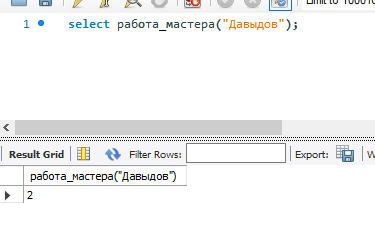


рис.

1. Функция показывает день, когда началась работы. Рис.45.

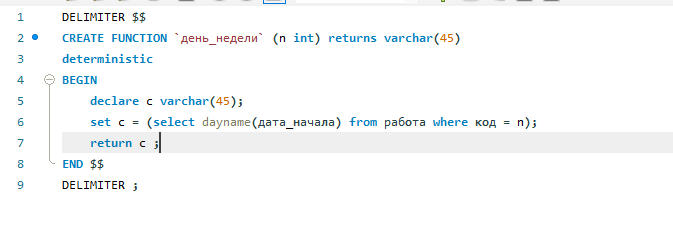


рис. день, когда началась работы

Проверка функции. Рис.46.

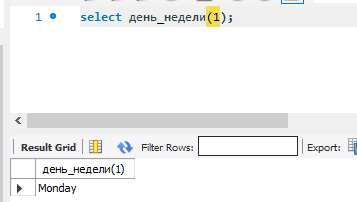


рис.

1. Функция выводит ФИО с помощью конкатенации строк . Рис.47.

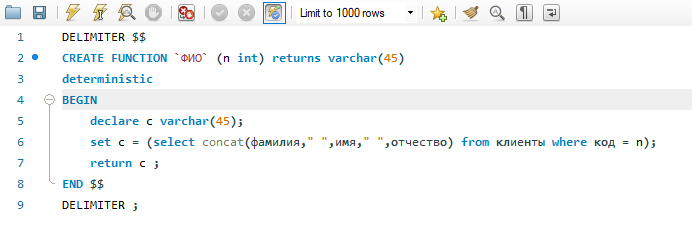


рис. ФИО

Проверка функции. Рис.48.

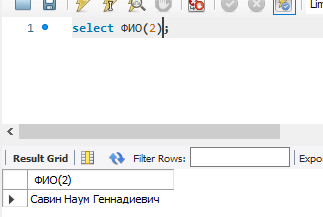


рис.

# триггеры

Триггер в MySQL — это определяемая пользователем SQL-команда, которая автоматически вызывается во время операций INSERT, DELETE или UPDATE. Код триггера связан с таблицей и уничтожается после удаления таблицы.

1. Триггер добавляет данные в таблицу при удаление . Рис.49.

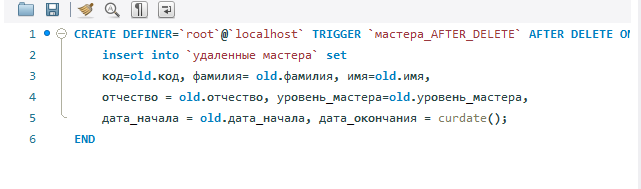


рис.

Проверка триггера.

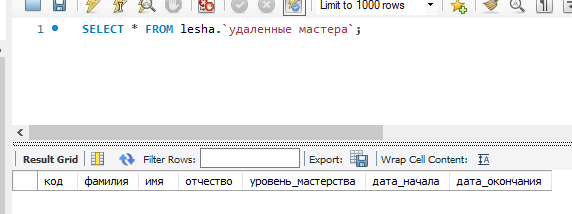


рис. Таблица удаленные мастера.

удаляем последнюю строку. Рис.51.

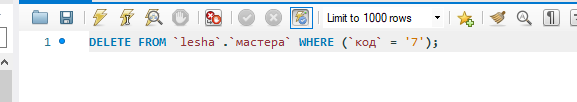


рис.

Проверяем таблицу. Рис.52

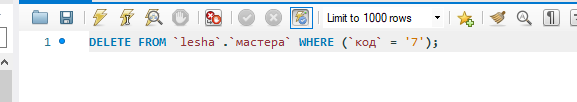


рис.

1. Триггер, меняет дату на новую при изменение. Рис.53.

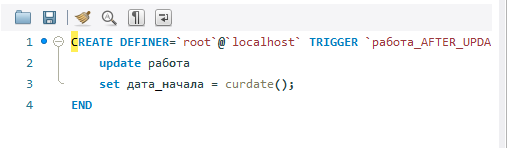


рис.

# Заключение

Основываясь на выполненной курсовой работе, можно прийти к выводу о том, что СУБД MySQL, которая служит в качестве системы управления базами данных, является очень хорошим инструментом для того, чтобы создавать собственные реляционные базы данных.

В данной курсовой работе была спроектирована информационная система, автоматизирующая работу ателье, реализующую следующие процессы: прием аказов от клиентов и распределение их по мастерам с учетом их квалификации или выполняемых функций, учет услуг ателье и учет цен на услуги, хранение информации о сотрудниках и клиентах ателье. Разработать запросы, перечень которых указан в разделе исходных данных по курсовой. Разработать хранимые процедуры: внесение данных о новом сотруднике ателье, внесение новой услуги, проверка наличия заказа конкретного клиента, подсчет суммы заказа и общей суммы, если у конкретного клиента несколько заказов, изменение стоимости конкретной услуги, изменение даты заказа (например, отсрочка заказа на неделю при определенных условиях).

Выполняя курсовую работу, были созданы 8 сущностей, а также разнообразные запросы на выборку, функции, триггеры и хранимые процедуры. Кроме того, были проведены тесты, для того чтобы проверить автоматизацию базы данных.

# Список литературы

* Изучаем SQL. Генерация, выборка и обработка данных, 3-е изд./

Алан Болье; пер. с англ. И.В. Красикова. — Киев. : “Диалектика”,

2021. — 402 с.: ил. — Парал. тит . англ.