**ЗАДАНИЕ ДЛЯ САМОСТОЯТЕЛЬНОЙ ПРАКТИЧЕСКОЙ РАБОТЫ ПО ДИСЦИПЛИНЕ «СИСТЕМЫ ОБРАБОТКИ И ХРАНЕНИЯ ДАННЫХ»**

Создать базу данных и таблицы в ней по выбранной теме, на основе

разработанных моделей.

Результат работы в виде отчета должен содержать:

* снимки экрана (скриншоты) процесса разработки;
* снимок экрана завершенной базы данных;
* снимки экрана (скриншоты) списка созданных таблиц;
* снимки экрана (скриншоты) структуры созданных таблиц;
* снимки экрана (скриншоты) содержания созданных таблиц;
* итоговый скрипт БД.

# Теоретические основы.

На данном этапе нам гораздо важней разобраться с созданием запросов, нежели сразу усложнять себе жизнь созданием таблиц, загрузкой в них данных и извлечением их оттуда. В этом пункте разъясняются основные принципы ввода команд; на примере нескольких запросов вы можете поближе познакомиться с работой mysql.

Ниже приведена простая команда, запрашивающая у сервера

информацию об его версии и текущей дате.

mysql> SELECT VERSION(), CURRENT\_DATE;

+ + +

| VERSION() | CURRENT\_DATE |

+ + +

| 3.22.20a-log | 1999-03-19 |

+ + +

1 row in set (0.01 sec) mysql>

Этот запрос иллюстрирует следующие особенности mysql:

* команда обычно состоит из SQL-выражения, за которым следует точка с запятой. (Из этого правила есть и исключения - команды без точки с запятой. Одним из них является упомянутая выше команда QUIT, остальные мы рассмотрим позднее.)
* когда пользователь вводит команду, mysql отправляет ее серверу для выполнения и выводит на экран сначала результаты, а затем - новую строку mysql>, что означает готовность к выполнению новых команд.
* mysql выводит результаты работы запроса в виде таблицы (строк и столбцов). В первой строке этой таблицы содержатся заголовки столбцов, а в следующих строках - собственно результаты. Обычно заголовками столбцов становятся имена, полученные из таблиц базы. Если же извлекается не столбец таблицы, а значение выражения (как это происходит в приведенном выше примере), mysql дает столбцу имя запрашиваемого выражения.
* mysql сообщает количество возвращаемых строк и время выполнения запроса, что позволяет в некоторой степени составить представление о производительности сервера. Эти значения обычно весьма впечатляют, так как представляют обычное (а не машинное время), кроме того, на них оказывает влияние загрузка сервера и скорость работы сети.

Для ввода ключевых слов можно использовать любой регистр символов.

Приведенные ниже запросы абсолютно идентичны:

mysql> SELECT VERSION(), CURRENT\_DATE;

mysql> select version(), current\_date; mysql> SeLeCt vErSiOn(), current\_DATE;

А это - еще один запрос. В нем демонстрируется использование mysql в качестве несложного калькулятора:

mysql> SELECT SIN(PI()/4), (4+1)\*5;

+ + +

| SIN(PI()/4) | (4+1)\*5 |

+ + +

| 0.707107 | 25 |

+ + +

Все команды, представленные выше, были относительно короткими и состояли из одной строки. В одну строку можно поместить и несколько команд. Но каждая из них должна заканчиваться точкой с запятой:

mysql> SELECT VERSION(); SELECT NOW();

+ +

| VERSION() |

+ +

| 3.22.20a-log |

+ +

+ +

| NOW() |

+ +

| 1999-03-19 00:15:33 |

+ +

Втискивать все команды в одну строку совсем не обязательно, так что создание длинных команд, занимающих несколько строк, никаких проблем не вызывает. Для mysql признаком завершения выражения является точка с запятой, а не конец строки (другими словами, mysql принимает команды без форматирования: строки с командами собираются, но не исполняются до тех пор, пока программа не обнаружит точку с запятой).

Вот пример несложного выражения, занимающего несколько строк:

mysql> SELECT

-> USER()

-> ,

-> CURRENT\_DATE;

+ + +

| USER() | CURRENT\_DATE |

+ + +

| joesmith@localhost | 1999-03-18 |

+ + +

Обратите внимание на то, как изменилась метка командной строки (с mysql> на ->) после ввода первой строки этого запроса. Таким образом программа mysql показывает, что завершенного выражения она пока что не получила и ожидает его полного ввода. Эта метка очень полезна, так как предоставляет весьма ценную информацию о состоянии программы. С ее помощью всегда можно узнать, чего ждет mysql.

Если вы решите отменить исполнение набираемой команды,

наберите \c:

mysql> SELECT

-> USER()

-> \c mysql>

Обратите внимание на метку: после ввода команды \c она снова принимает вид mysql>, показывая, что программа mysql перешла в режим ожидания указаний.

Ниже приведены все возможные варианта вида метки командной строки и соответствующие им состояния mysql:

|  |  |
| --- | --- |
| **Метка** | **Значение** |
| mysql> | Ожидание новой команды. |
| -> | Ожидание следующей строки многострочной команды. |
| '> | Ожидание следующей строки, сбор строкового выражения, начинающегося с одиночной кавычки (`''). |
| "> | Ожидание следующей строки, сбор строкового выражения, начинающегося с двойной кавычки (`"'). |

Обычно многострочные команды получаются случайно, когда хочешь создать обычную команду, но забываешь поставить завершающую точку с запятой. В таком случае mysql ожидает продолжения:

mysql> SELECT USER()

->

Если с вами произошло подобное (завершили команду, но программа выдает только метку ->), то mysql, вероятнее всего, ждет точки с запятой. Не обратив внимание на метку командной строки, можно довольно долго ждать выполнения команды, не понимая в чем дело. А достаточно лишь поставить точку с запятой, завершив команду, которую mysql и выполнит:

mysql> SELECT USER()

-> ;

+ +

| USER() |

+ +

| joesmith@localhost |

+ +

Метки '> и "> используются при сборе строк. В MySQL строки можно заключать как в одинарные (`''), так и в двойные (`"') кавычки (можно, например, написать 'hello' или "goodbye"), к тому же, mysql позволяет вводить строковые выражения, состоящие из нескольких строчек текста. Метка '> или "> обозначает, что вы ввели строку, открывающуюся символом

кавычек `'' или `"', но еще не ввели завершающую строковое выражение закрывающую кавычку.

Это нормально, если вы собираетесь создать большое строковое выражение из нескольких строчек. Но это не слишком частый случай. Гораздо чаще оказывается, что вы просто забыли поставить закрывающую кавычку. Например:

mysql> SELECT \* FROM my\_table WHERE name = "Smith AND age <

30;

">

Если ввести такую команду SELECT, нажать Enter и подождать

результатов, ничего не произойдет. Тут-то и нужно обратить внимание на метку командной строки, выглядящую вот так: ">. Это значит, что mysql ждет ввода завершающей части строки. (Теперь заметили ошибку в команде? В строке "Smith нет закрывающей кавычки.)

Что делать в этом случае? Проще всего было бы отменить команду. Однако теперь просто набрать \c нельзя, так как mysql примет эти символы за часть собираемой строки! Вместо этого нужно ввести закрывающие кавычки (тем самым дав mysql понять, что строка закончилась) и лишь затем набрать \c:

mysql> SELECT \* FROM my\_table WHERE name = "Smith AND age <

30;

"> "\c mysql>

Метка командной строки снова примет вид mysql>, показывая

готовность mysql к выполнению команд.

Знать значение меток '> и "> необходимо, так как при вводе незавершенной строки все последующие строки будут игнорироваться mysql - включая строку с командой QUIT! Это может основательно сбить с толку, особенно если не знать, что для отмены команды перед соответствующей последовательностью символов необходимо поставить закрывающую кавычку.

# Практическое занятие по созданию БД на языке SQL

Утилита mysql (иногда называемая также «терминальным монитором» или просто «монитором») представляет собой интерактивную программу, позволяющую подсоединяться к MySQL-серверу, запускать запросы, и просматривать результаты. Программа mysql может работать и в пакетном режиме: для этого необходимо записать все запросы в файл, а затем передать его содержимое на исполнение mysql. Ниже описаны оба способа использования mysql.

Увидеть список команд программы mysql можно, запустив ее с параметром - -help:

shell> mysql --help

На Вашем компьютере необходимо установить mysql и существить связь с сервером MySQL.

# Подсоединение к серверу и отсоединение от него

Для этого прежде всего надо запустить сервер MySQL. Идем в системное меню Пуск — Программы — MySQL — MySQL Server 5.1 — MySQL Command Line Client.

При подключении к серверу с помощью mysql обычно нужно ввести имя пользователя MySQL и, в большинстве случаев, пароль. Если сервер запущен не на том компьютере, с которого вы вошли в систему, необходимо также указать имя хоста. Параметры соединения (а именно - соответствующее имя хоста, пользователя и пароль) вы сможете узнать у администратора. Получив соответствующие параметры, подсоединиться к серверу можно следующим образом:

shell> mysql -h host -u user -p Enter password: \*\*\*\*\*\*\*\*

Символы \*\*\*\*\*\*\*\* обозначают ваш пароль; введите его, когда mysql выведет на экран запрос Enter password:.

Если все сработает, на экране должна появиться следующая информация и метка командной строки mysql>:

shell> mysql -h host -u user -p Enter password: \*\*\*\*\*\*\*\*

Welcome to the MySQL monitor. Commands end with ; or \g. Your MySQL connection id is 459 to server version: 3.22.20a-

log

Type 'help' for help. mysql>

Метка обозначает, что программа mysql готова к вводу команд.

В некоторых вариантах установки MySQL возможно подсоединение к запущенному на локальном хосте серверу без ввода имени пользователя (пользователь anonymous). Если ваша система настроена именно так, подсоединиться к серверу вы сможете, запустив mysql со следующими параметрами:

shell> mysql

После установки соединения можно в любой момент отключиться от сервера, набрав в командной строке mysql> команду QUIT:

mysql> QUIT Bye

Отсоединиться от сервера можно и при помощи сочетания клавиш

Control-D.

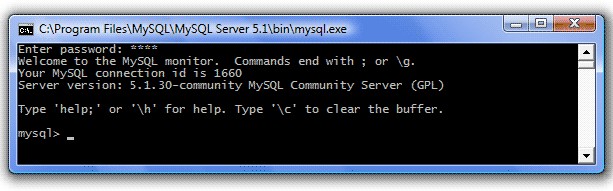
Большая часть приведенных ниже примеров построена с учетом того, что соединение с сервером уже установлено. Это видно по наличию в них командной строки mysql>.

# Создание базы данных и таблиц

Для этого прежде всего надо запустить сервер MySQL. Идем в системное меню Пуск — Программы — MySQL — MySQL Server 5.1 — MySQL Command Line Client. Откроется окно, предлагающее ввести пароль.



Нажимаем Enter на клавиатуре, если вы не указывали пароль при настройке сервера или указываем пароль, если вы его задавали. Ждем приглашения mysql>.



Нам надо создать базу данных, которую назовем banya. Для этого в SQL существует оператор create database. Создание базы данных имеет следующий синтаксис:

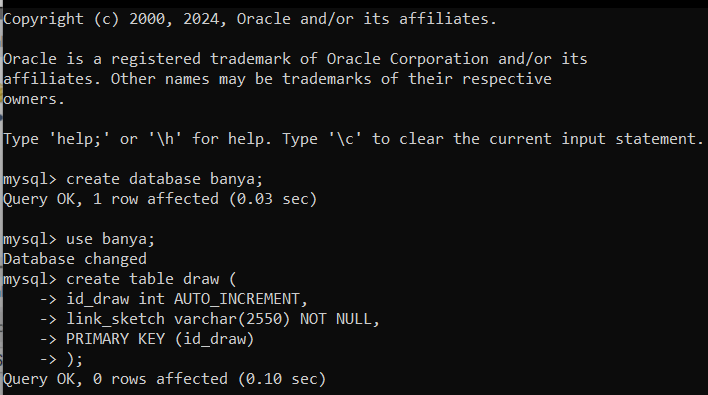
create database имя\_базы\_данных;

Максимальная длина имени БД составляет 64 знака и может включать буквы, цифры, символ «\_» и символ «$». Имя может начинаться с цифры, но не должно полностью состоять из цифр. Любой запрос к БД заканчивается

точкой с запятой (этот символ называется разделителем — delimiter).

Получив запрос, сервер выполняет его и в случае успеха выдает сообщение

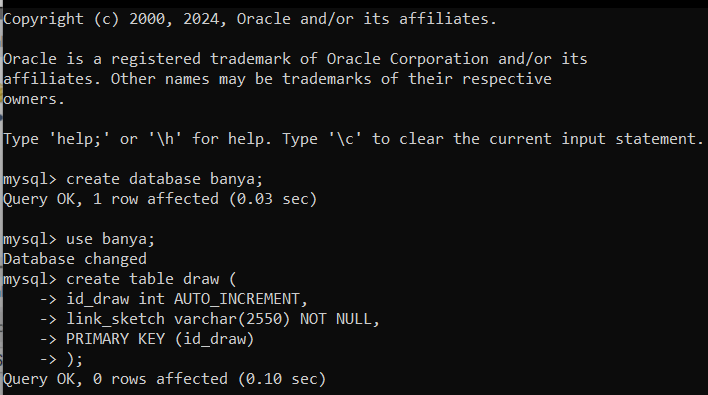
«Query OK …».

\Итак, создадим БД banya:

Нажимаем Enter и видим ответ «Query OK …», означающий, что БД была создана:

Вот так все просто. Теперь в этой базе данных нам надо создать 3 таблицы: темы, пользователи и сообщения. Но перед тем, как это делать, нам надо указать серверу в какую именно БД мы создаем таблицы, т.е. надо выбрать БД для работы. Для этого используется оператор use. Синтаксис выбора БД для работы следующий:

use имя\_базы\_данных;

Итак, выберем для работы нашу БД banya:

Нажимаем Enter и видим ответ «Database changed» — база данных выбрана.

Выбирать БД необходимо в каждом сеансе работы с MySQL.

Для создания таблиц в SQL существует оператор create table.

Создание базы данных имеет следующий синтаксис:

create table имя\_таблицы (имя\_первого\_столбца тип, имя\_второго\_столбца тип, ..., имя\_последнего\_столбца тип );

Требования к именам таблиц и столбцов такие же, как и для имен БД. К каждому столбцу привязан определенный тип данных, который ограничивает характер информации, которую можно хранить в столбце (например, предотвращает ввод букв в числовое поле). MySQL поддерживает несколько типов данных: числовые, строковые, календарные и специальный тип NULL, обозначающий отсутствие информации. Подробно о типах данных мы будем говорить в следующем уроке, а пока вернемся к нашим таблицам. В них у нас всего два типа данных — целочисленные значения (int) и строки (text). Итак, создадим таблицы:

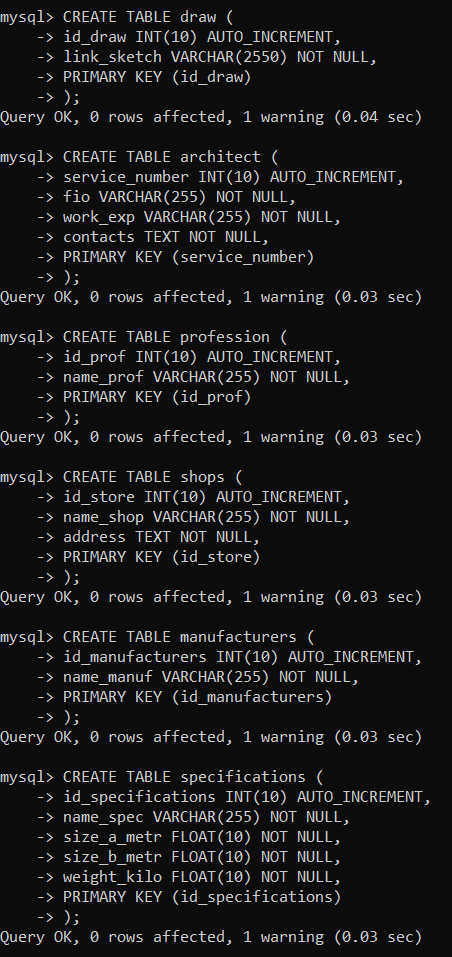


Рисунок 1 – Создание таблиц

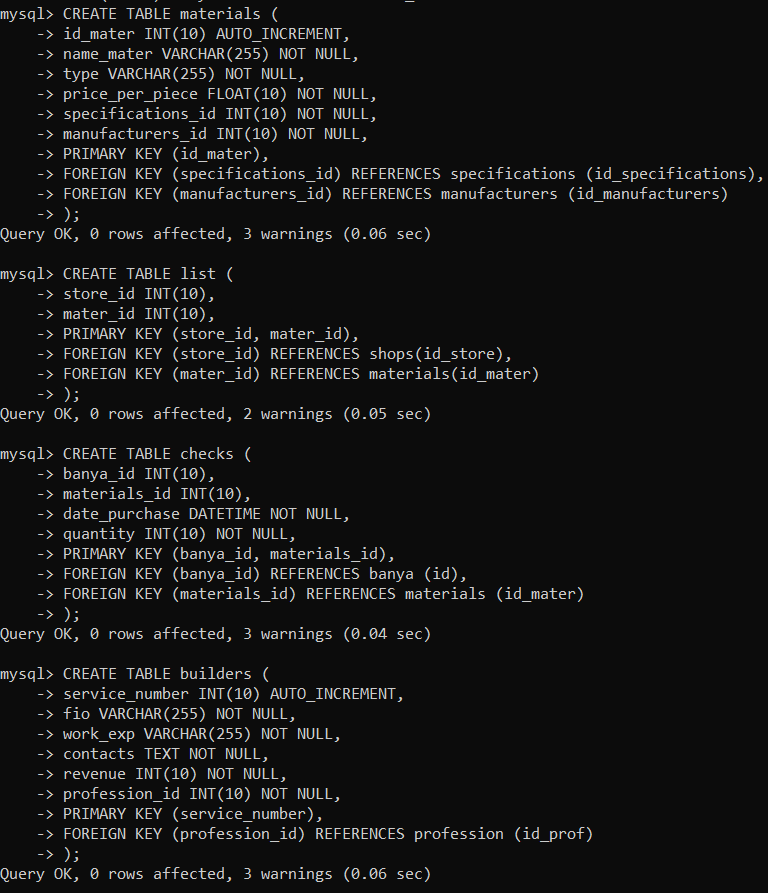


Рисунок 2 – Создание таблиц

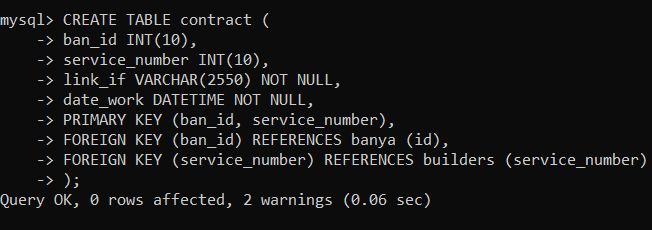


Рисунок 3 – Создание таблиц

Нажимаем Enter — таблица создана:

Для этого используется оператор INSERT. Синтаксис можно использовать двух видов. Первый вариант используется для внесения данных во все поля таблицы:

1. INSERT INTO имя\_таблицы VALUES ('значение\_первого\_столбца','значение\_второго\_столбца',

..., 'значение\_последнего\_столбца');

Давайте попробуем внести в нашу таблицу shops следующие значения:

INSERT INTO shops (id\_store, name\_shop, address) VALUES

(1, 'Петрович', 'рабочий посёлок Боброво, с67Ю'),

(2, 'Лемана Про', 'МКАД, 24-й километр, 1А'),

(3, 'Благовар', 'МКАД, 25-й километр, вл1'),

(4, 'Империя Печей', 'Московская улица 10, Чехов'),

(5, 'Магазин "ЭкоСтрой"', 'г. Москва, ул. Природная, д. 6'),

(6, 'Магазин "Качественные материалы"', 'г. Москва, ул. Качества, д. 7'),

(7, 'Магазин "Баня плюс"', 'г. Москва, ул. Солнечная, д. 8'),

(8, 'Магазин "Все для ремонта"', 'г. Москва, ул. Ремонтная, д. 9'),

(9, 'Магазин "СтройПартнер"', 'г. Москва, ул. Партнерская, д. 10'),

(10, 'Магазин "СтройГрад"', 'г. Москва, ул. Тихая, д. 5');

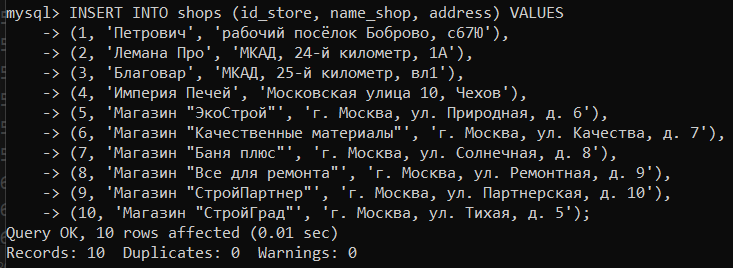


Рисунок 4 – Внесение данных в таблицу shops

Теперь внесем данные во вторую таблицу — architect:

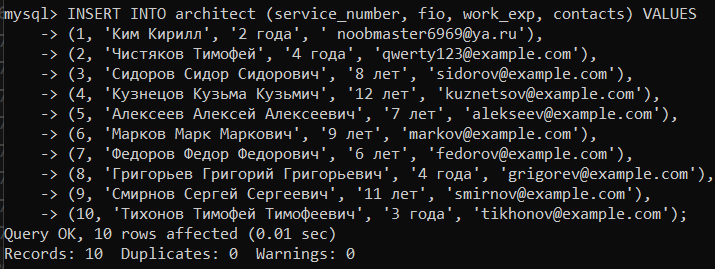


Рисунок 5 – Внесение данных в таблицу architect

Теперь внесем данные в третью таблицу — manufacturers:

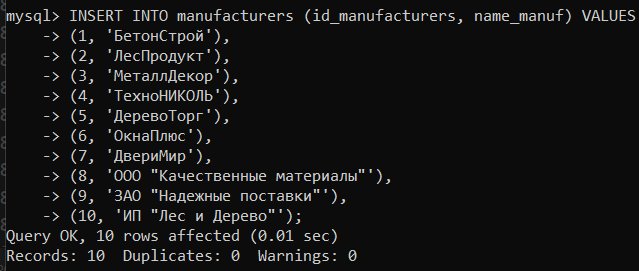


Рисунок 6 – Внесение данных в таблицу manufacturers

Теперь внесем данные в четвертую таблицу — profession:

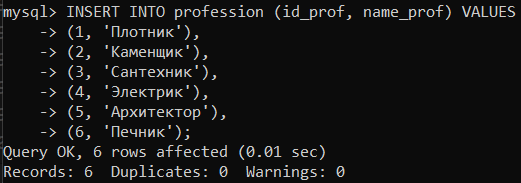


Рисунок 7 – Внесение данных в таблицу profession

Теперь внесем данные в пятую таблицу — draw:

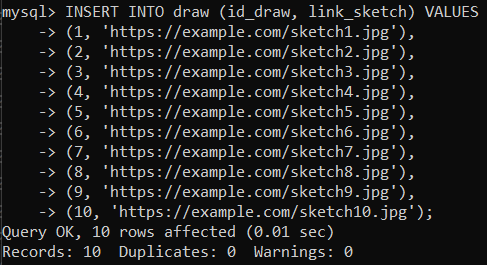


Рисунок 8 – Внесение данных в таблицу draw

Теперь внесем данные в шестую таблицу — specifications:

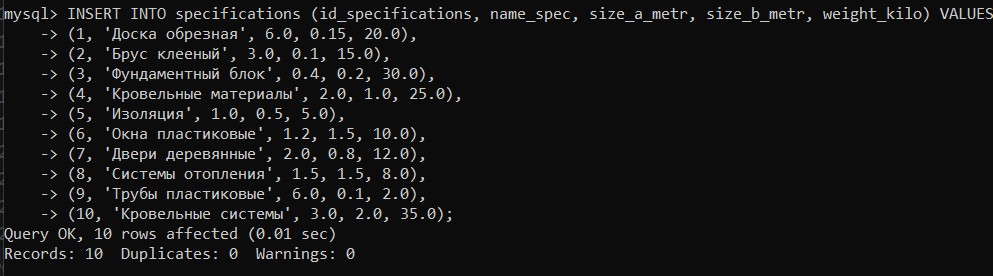


Рисунок 9 – Внесение данных в таблицу specifications

Теперь внесем данные в седьмую таблицу — project:

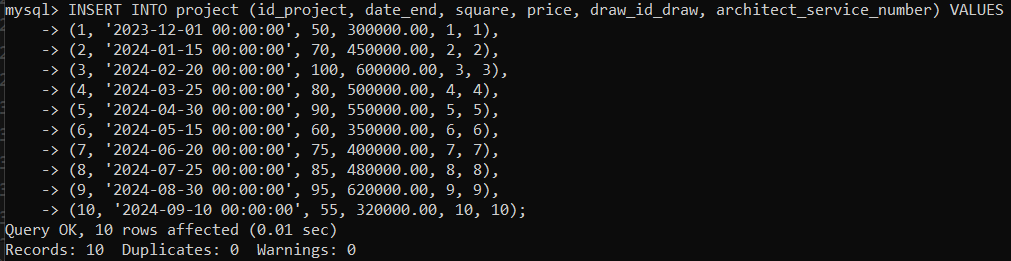


Рисунок 10 – Внесение данных в таблицу project

Теперь внесем данные в восьмую таблицу — banya:

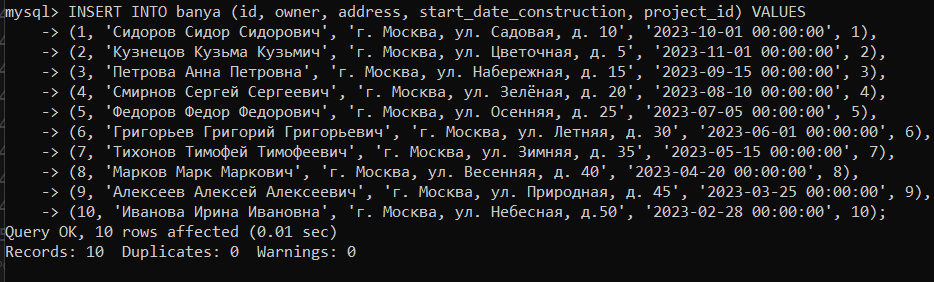


Рисунок 11 – Внесение данных в таблицу banya

Теперь внесем данные в девятую таблицу — materials:

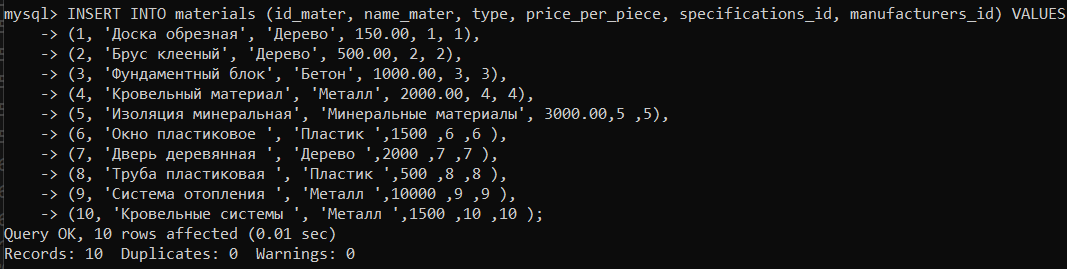


Рисунок 12 – Внесение данных в таблицу materials

Теперь внесем данные в десятую таблицу — list:

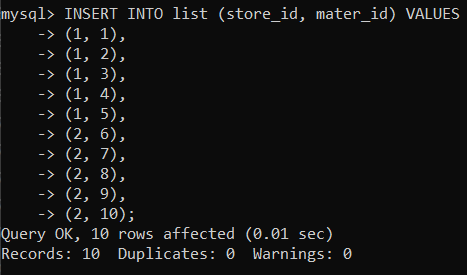


Рисунок 13 – Внесение данных в таблицу list

Теперь внесем данные в одиннадцатую таблицу — checks:

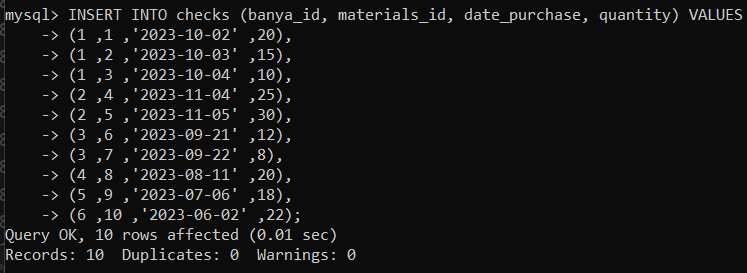


Рисунок 14 – Внесение данных в таблицу checks

Теперь внесем данные в двенадцатую таблицу — builders:

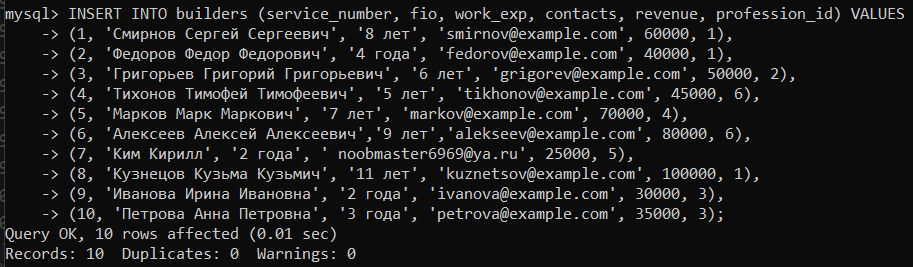


Рисунок 15 – Внесение данных в таблицу builders

Теперь внесем данные в тринадцатую таблицу — contract:

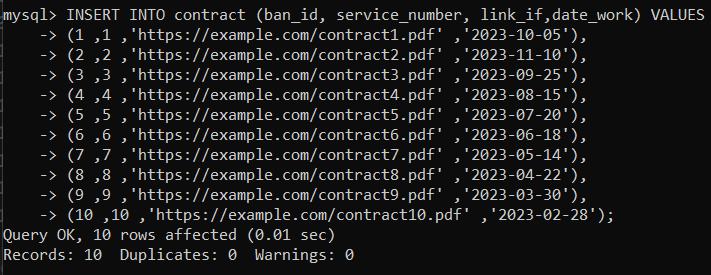


Рисунок 16 – Внесение данных в таблицу contract

Итак, в нашей БД banya есть таблицы: contract, builders, checks, list, materials, banya, project, specifications, manufacturers, shops, profession, architect, draw. И мы хотим посмотреть, какие данные в них содержатся. Для этого в SQL существует оператор SELECT. Синтаксис его использования следующий:

1. SELECT что\_выбрать FROM откуда\_выбрать;

Вместо «что\_выбрать» мы должны указать либо имя столбца, значения которого хотим увидеть, либо имена нескольких столбцов через запятую, либо символ звездочки (\*), означающий выбор всех столбцов таблицы. Вместо

«откуда\_выбрать» следует указать имя таблицы.

Давайте посмотрим все столбцы из таблицы architect:

SELECT \* FROM architect;

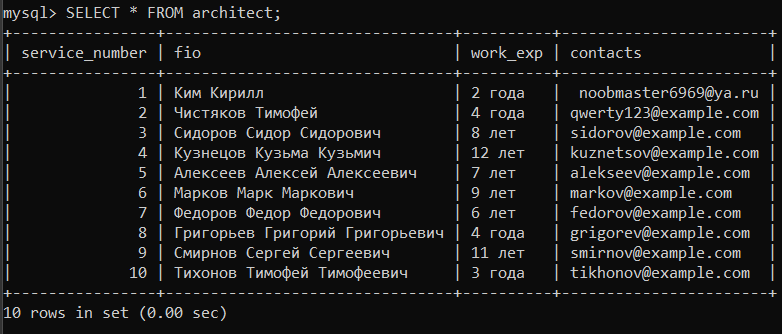


Рисунок 17 – Данные из таблицы architect