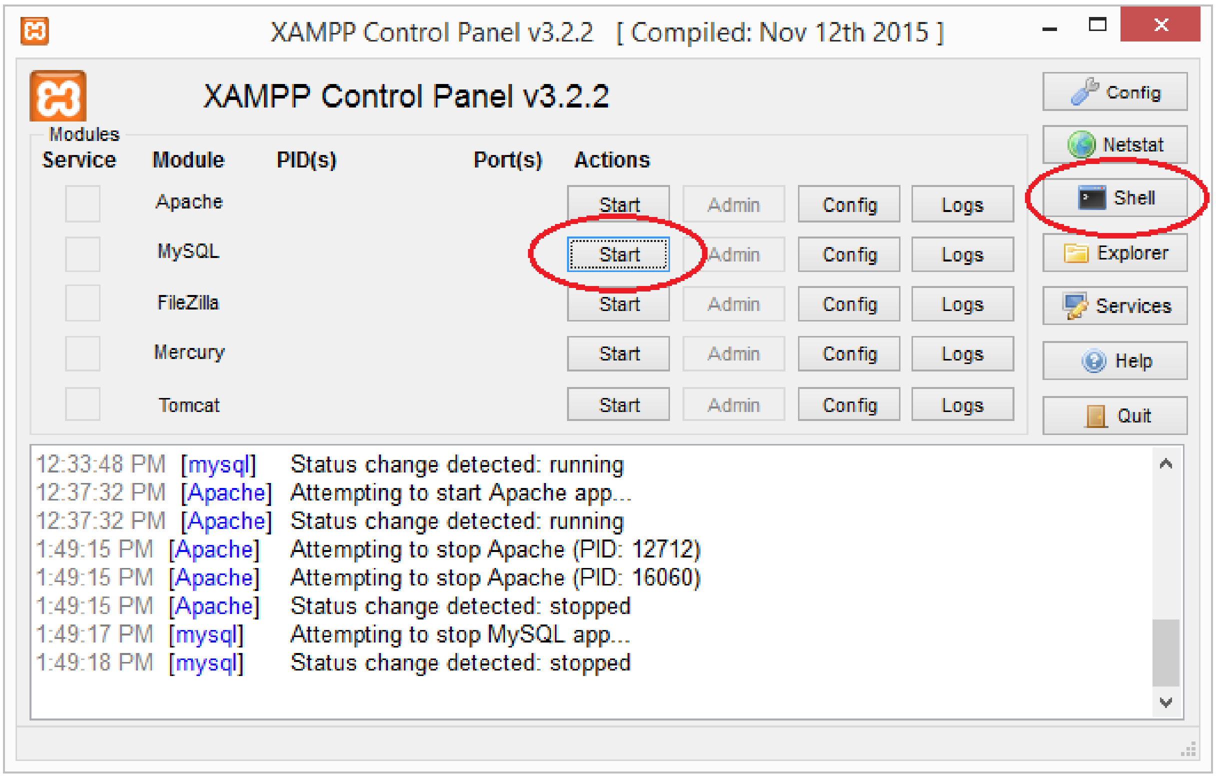
1. DATABASE CREATION USING SQL LANGUAGE

Goal: learn how to create and link database tables using the MySQL DBMS.

1.1. Install MySQL

It is recommended to install MySQL using one of the freely distributed WAMP (Windows, Apache, MySQL, and PHP) or LAMP (Linux, Apache, MySQL, and PHP) servers, such as OpenServer or XAMPP, to simplify the installation and subsequent use of the database management system (DBMS). In subsequent examples of laboratory work, the XAMPP server will be used.

Once the XAMPP server is installed, run the server control panel, run MySQL, and open the server command line (Figure 1.1).



```
12:33:48 PM  [mysql]   Status change detected: running
12:37:32 PM  [Apache]  Attempting to start Apache app...
12:37:32 PM  [Apache]  Status change detected: running
1:49:15 PM  [Apache]   Attempting to stop Apache (PID: 12712)
1:49:15 PM  [Apache]   Attempting to stop Apache (PID: 16060)
1:49:15 PM  [Apache]   Status change detected: stopped
1:49:17 PM  [mysql]    Attempting to stop MySQL app...
1:49:18 PM  [mysql]    Status change detected: stopped
```

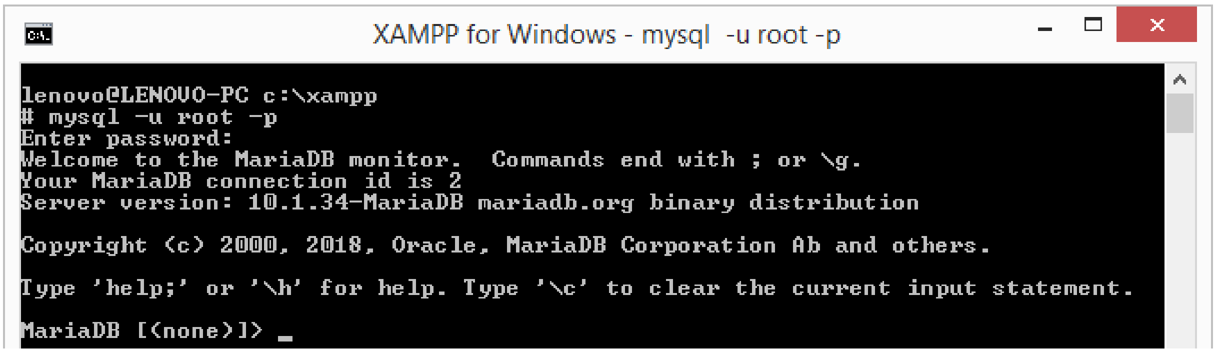
Figure 1.1

1.2. Connect to MySQL and create a database

Use the following command in the MySQL server command line:

mysql -u root -p

Then enter the password (Figure 1.2). Usually, the root user password is empty, so just press Enter.



```
lenovo@LENOVO-PC c:\xampp
# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 2
Server version: 10.1.34-MariaDB mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> _
```

Figure 1.2

To disconnect from MySQL, use the exit command.

The main commands that will be used from time to time when working with MySQL are the following:

1) USE database – select a database (DB) for further work;

2) SHOW DATABASES – get a list of databases;

3) SHOW TABLES – get a list of tables for the selected database;

4) SHOW COLUMNS FROM table – get information about the table;

5) SHOW INDEX FROM table – get information about the indexes defined for the table.

A database should be created using the command:

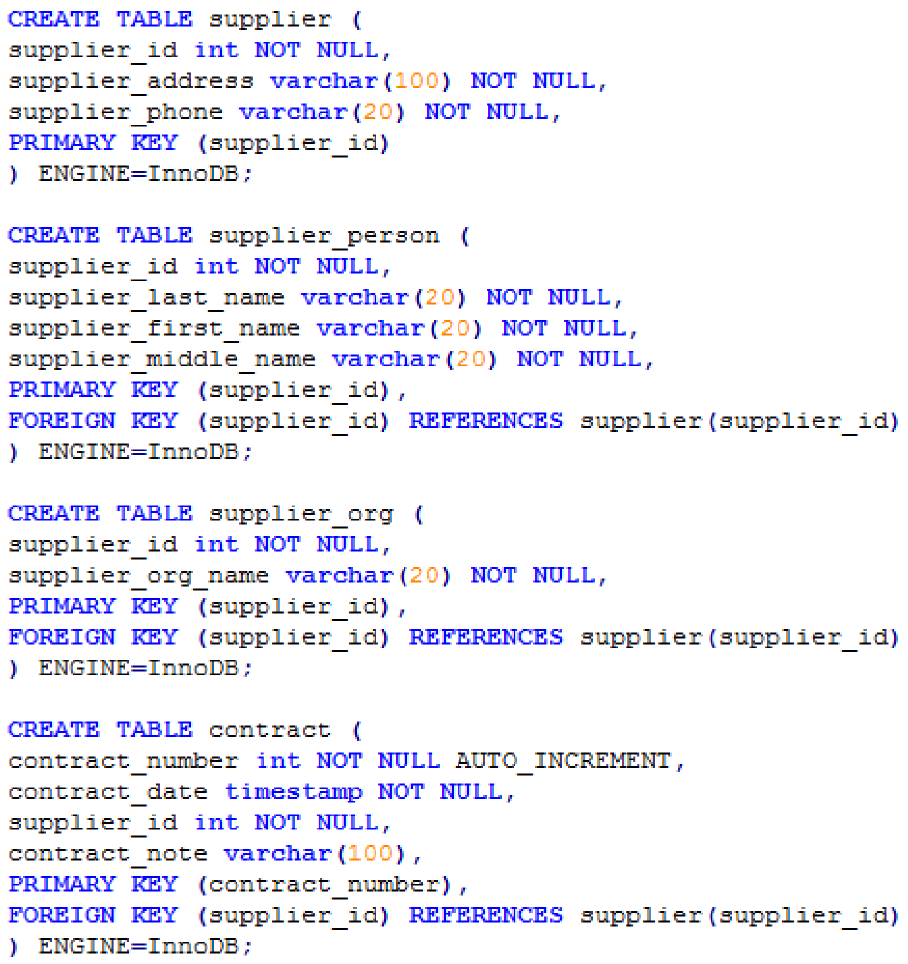
CREATE DATABASE supply;

Execution of this command will allow creating a database, the work to be considered in the laboratory practice. You can check the database created using the SHOW DATABASES command.

1.3. Create database tables and link them

To study the peculiarities of working with a MySQL database, a database of a company that purchases goods from different suppliers will be considered. The purchase of goods is carried out in batches and is executed in the form of supply contracts. Each contract has a unique number and is concluded with only one supplier. The documents for each contract specify the name, the size of the delivered batch, and the price (in UAH).

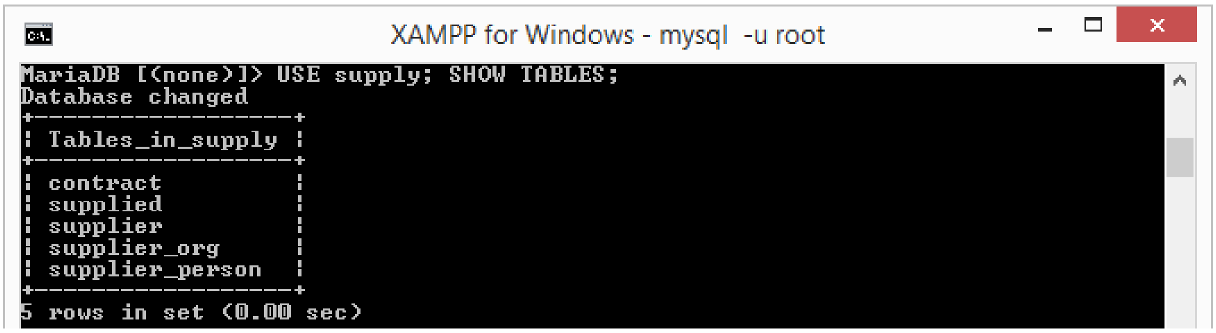
Creating tables is performed using the CREATE TABLE statement. Thus, for the current database, it is necessary to create the following tables:



```
CREATE TABLE supplier (
supplier_id int NOT NULL,
supplier_address varchar(100) NOT NULL,
supplier_phone varchar(20) NOT NULL,
PRIMARY KEY (supplier_id)
) ENGINE=InnoDB;

CREATE TABLE supplier_person (
supplier_id int NOT NULL,
supplier_last_name varchar(20) NOT NULL,
supplier_first_name varchar(20) NOT NULL,
supplier_middle_name varchar(20) NOT NULL,
PRIMARY KEY (supplier_id),
FOREIGN KEY (supplier_id) REFERENCES supplier(supplier_id)
) ENGINE=InnoDB;

CREATE TABLE supplier_org (
supplier_id int NOT NULL,
supplier_org_name varchar(20) NOT NULL,
PRIMARY KEY (supplier_id),
FOREIGN KEY (supplier_id) REFERENCES supplier(supplier_id)
) ENGINE=InnoDB;

CREATE TABLE contract (
contract_number int NOT NULL AUTO_INCREMENT,
contract_date timestamp NOT NULL,
supplier_id int NOT NULL,
contract_note varchar(100),
PRIMARY KEY (contract_number),
FOREIGN KEY (supplier_id) REFERENCES supplier(supplier_id)
) ENGINE=InnoDB;
```

```
CREATE TABLE supplied (
contract_number int NOT NULL,
supplied_product varchar(20) NOT NULL,
supplied_amount decimal(4,0) NOT NULL,
supplied_cost decimal(8,2) NOT NULL,
PRIMARY KEY (contract_number, supplied_product),
FOREIGN KEY (contract_number) REFERENCES contract(contract_number)
) ENGINE=InnoDB;
```

Check the generated tables in the supply database (Figure 1.3).



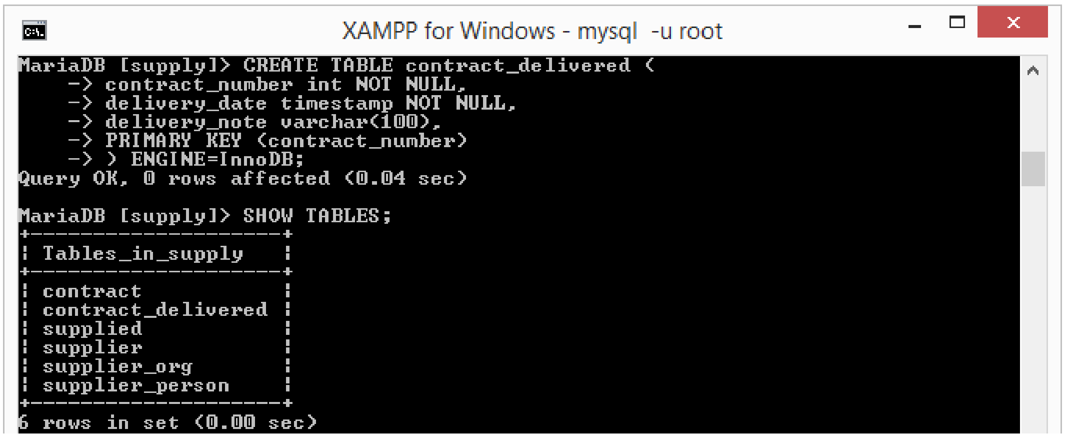
```
MariaDB [(none)]> USE supply; SHOW TABLES;
Database changed
+------------------+
| Tables_in_supply |
+------------------+
| contract         |
| supplied         |
| supplier         |
| supplier_org     |
| supplier_person  |
+------------------+
5 rows in set (0.00 sec)
```

Figure 1.3

1.4. Modification of table structure

Change the structure of an existing table using the ALTER TABLE statement. Assume that you need to create another table in the supply database, which will be used to store data on the facts of implementation of supply contracts (Figure 1.4).

```
CREATE TABLE contract_delivered (
contract_number int NOT NULL,
delivery_date timestamp NOT NULL,
delivery_note varchar(100),
PRIMARY KEY (contract_number)
) ENGINE=InnoDB;
```

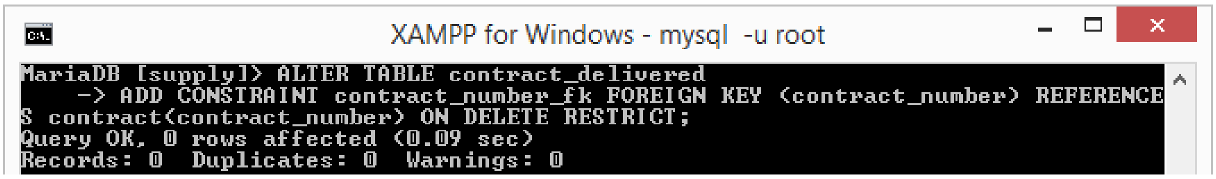


```
MariaDB [supply]> CREATE TABLE contract_delivered (
    -> contract_number int NOT NULL,
    -> delivery_date timestamp NOT NULL,
    -> delivery_note varchar(100),
    -> PRIMARY KEY (contract_number)
    -> ) ENGINE=InnoDB;
Query OK, 0 rows affected (0.04 sec)

MariaDB [supply]> SHOW TABLES;
+--------------------+
| Tables_in_supply   |
+--------------------+
| contract           |
| contract_delivered |
| supplied           |
| supplier           |
| supplier_org       |
| supplier_person    |
+--------------------+
6 rows in set (0.00 sec)
```

Figure 1.4

To link the created contract_delivered table with the contract table, apply the ALTER TABLE command (Figure 1.5).

```
ALTER TABLE contract_delivered
ADD CONSTRAINT contract_number_fk FOREIGN KEY (contract_number)
    REFERENCES contract(contract_number);
```

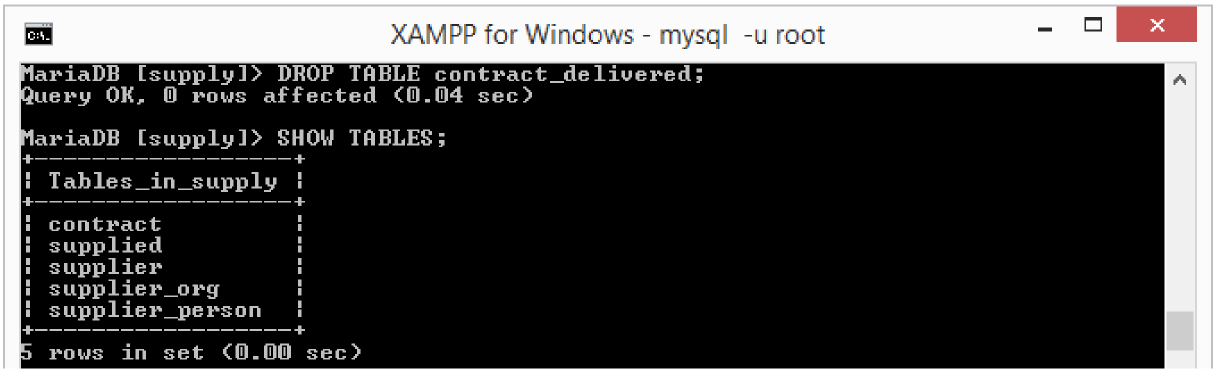


```
MariaDB [supply]> ALTER TABLE contract_delivered
    -> ADD CONSTRAINT contract_number_fk FOREIGN KEY (contract_number) REFERENCE
S contract(contract_number) ON DELETE RESTRICT;
Query OK, 0 rows affected (0.09 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Figure 1.5

1.5. Delete tables

Delete a table using the DROP TABLE statement. Since the created contract_delivered table will not be used in future work, it can be deleted using this command (Figure 1.6).



```
MariaDB [supply]> DROP TABLE contract_delivered;
Query OK, 0 rows affected (0.04 sec)

MariaDB [supply]> SHOW TABLES;
+------------------+
| Tables_in_supply |
+------------------+
| contract         |
| supplied         |
| supplier         |
| supplier_org     |
| supplier_person  |
+------------------+
5 rows in set (0.00 sec)
```

Figure 1.6

1.6. Questions

23. How to access the command line of the MySQL server?

24. How to make a connection to the MySQL server using the name and password of the specific user?

25. List the basic commands used for MySQL server administration and their purposes.

26. Which command is used to create the new database? How to check that database is created?

27. Which SQL statements are used to create and link tables?

28. Which SQL statement is used to modify the table’s structure?

29. Which SQL statement is used to delete tables from a database?

30. How to check the presence or absence of the created or removed tabled respectively?

31. How to set the name of a foreign key while linking tables?

32. Which shortcomings are present in the current database structure? How to resolve these issues?