

1. 소스코드

```

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.PreparedStatement;

public class createtable {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Connection conn = null;
        Statement stmt = null;
        ResultSet result = null;
        PreparedStatement preStmt = null;

        try {
            String jdbcDriver = "jdbc:mariadb://localhost:3306/lab";
            String dbUser = "root";
            String dbPass = "a1234";

            conn = DriverManager.getConnection(jdbcDriver, dbUser,
dbPass);

            stmt = conn.createStatement();

            String create_table_Vineyard = "CREATE TABLE IF NOT
EXISTS Vineyard("

                +"VineyardID INT PRIMARY KEY AUTO_INCREMENT,"

                +"owner CHAR(20),"

                +"tel_number CHAR(13),"

                +"stock_red INT CHECK(stock_red >= 0),"

                +"stock_white INT CHECK(stock_white >= 0),"

                +"money INT CHECK(money >= 0)"

                +");";

            String create_table_Field = "CREATE TABLE IF NOT EXISTS
Field("

                +"FieldID INT NOT NULL PRIMARY KEY AUTO_INCREMENT,"

                +"location CHAR(20),"

                +"area INT CHECK(area >= 0),"

                +"variety CHAR(5) CHECK(variety IN('red', 'white')),"

                +"VineyardID INT,"

                +"FOREIGN KEY (VineyardID) REFERENCES Vineyard(VineyardID)"

                +");";

            String create_table_GrapeProduction = "CREATE TABLE IF

```

```

NOT EXISTS GrapeProduction("
    +"GPID INT NOT NULL PRIMARY KEY AUTO_INCREMENT,"
    +"year CHAR(5),"
    +"amount INT CHECK (amount >= 0),"
    +"FieldID INT,"
    +"FOREIGN KEY (FieldID) REFERENCES Field(FieldID)"
    +");";

String create_table_Winery = "CREATE TABLE IF NOT
EXISTS Winery("
    +"WineryID INT PRIMARY KEY AUTO_INCREMENT,"
    +"owner CHAR(20),"
    +"tel_number CHAR(13),"
    +"stock_red INT CHECK (stock_red >= 0),"
    +"stock_white INT CHECK(stock_white >= 0),"
    +"money INT CHECK(money >= 0),"
    +"grade_ID CHAR(5) CHECK (grade_ID IN ('1st', '2nd', '3rd')),"
    +"VineyardID INT,"
    +"FOREIGN KEY (VineyardID) REFERENCES Vineyard(VineyardID)"
    +");";

String create_table_WineStock = "CREATE TABLE IF NOT
EXISTS WineStock("
    +"WineStockID INT PRIMARY KEY AUTO_INCREMENT,"
    +"categoryID CHAR(10) CHECK (categoryID IN ('red', 'white', 'blush',
'sparkling')),"
    +"amount INT CHECK(amount >= 0),"
    +"WineryID INT,"
    +"FOREIGN KEY (WineryID) REFERENCES Winery(WineryID)"
    +");";

String create_table_WineProduction = "CREATE TABLE IF
NOT EXISTS WineProduction("
    +"WPID INT PRIMARY KEY AUTO_INCREMENT,"
    +"categoryID CHAR(10) CHECK (categoryID IN ('red', 'white', 'blush',
'sparkling')),"
    +"year CHAR(5),"
    +"amount INT CHECK (amount >= 0),"
    +"WineryID INT,"
    +"FOREIGN KEY (WineryID) REFERENCES Winery(WineryID)"

```

```
+");";
```

```
String create_table_WineTrade = "CREATE TABLE IF NOT  
EXISTS WineTrade("  
    +"WTID INT PRIMARY KEY AUTO_INCREMENT,"  
    +"categoryID CHAR(10) CHECK (categoryID IN ('red', 'white', 'blush',  
'sparkling')),"  
    +"year CHAR(5),"  
    +"amount INT CHECK (amount >= 0),"  
    +"WineryID INT,"  
    +"FOREIGN KEY (WineryID) REFERENCES Winery(WineryID)"  
    +");";
```

```
String create_table_WineCategory = "CREATE TABLE IF NOT  
EXISTS WineCategory("  
    +"categoryID INT PRIMARY KEY AUTO_INCREMENT,"  
    +"title CHAR(10) CHECK (title IN ('red', 'white', 'blush', 'sparkling')),"  
    +"standardPrice INT"  
    +");";
```

```
String create_table_WineGrade = "CREATE TABLE IF NOT  
EXISTS WineGrade("  
    +"gradeID INT PRIMARY KEY AUTO_INCREMENT,"  
    +"title CHAR(5) CHECK (title IN ('1st', '2nd', '3rd')),"  
    +"ratio numeric(3,1)"  
    +");";
```

```
String create_table_GrapeTrade = "CREATE TABLE IF NOT  
EXISTS GrapeTrade("  
    +"GT_ID INT PRIMARY KEY  
    AUTO_INCREMENT,"  
    +"year CHAR(5),"  
    +"variety CHAR(5) CHECK (variety IN ('red',  
'white')),"  
    +"amount INT NOT NULL CHECK(amount >=  
0),"  
    +"VineyardID INT,"  
    +"FOREIGN KEY (VineyardID) REFERENCES  
Vineyard(VineyardID),"  
    +"WineryID INT,"  
    +"FOREIGN KEY (WineryID) REFERENCES  
Winery(WineryID)"  
    +");";
```

```
stmt.executeUpdate(create_table_Vineyard);  
stmt.executeUpdate(create_table_Field);  
stmt.executeUpdate(create_table_GrapeProduction);
```

```

        stmt.executeUpdate(create_table_Winery);
        stmt.executeUpdate(create_table_WineStock);
        stmt.executeUpdate(create_table_WineProduction);
        stmt.executeUpdate(create_table_WineTrade);
        stmt.executeUpdate(create_table_WineCategory);
        stmt.executeUpdate(create_table_WineGrade);
        stmt.executeUpdate(create_table_GrapeTrade);
        System.out.println("Table_create_sucess");

        String insert_value_WineCategory = "INSERT INTO
WineCategory(title, standardPrice) value(?, ?)";

        preStmt =
conn.prepareStatement(insert_value_WineCategory);

        String [] Winetitle = {"red", "white", "blush", "sparkling"};
        int [] StandardPrice = {10, 10, 14, 20};

        for (int i = 0; i < 4; i++) {
            preStmt.setString(1, Winetitle[i]);
            preStmt.setInt(2, StandardPrice[i]);
            preStmt.executeUpdate();
            preStmt.clearParameters();
        }
        System.out.println("Wine Category Sucess");

        String insert_value_WineGrade = "INSERT INTO
WineGrade(title, ratio) values(?, ?)";

        preStmt = conn.prepareStatement(insert_value_WineGrade);

        String [] Gradetitle = {"1st", "2nd", "3rd"};
        double [] ratio = {2.0, 1.0, 0.5};

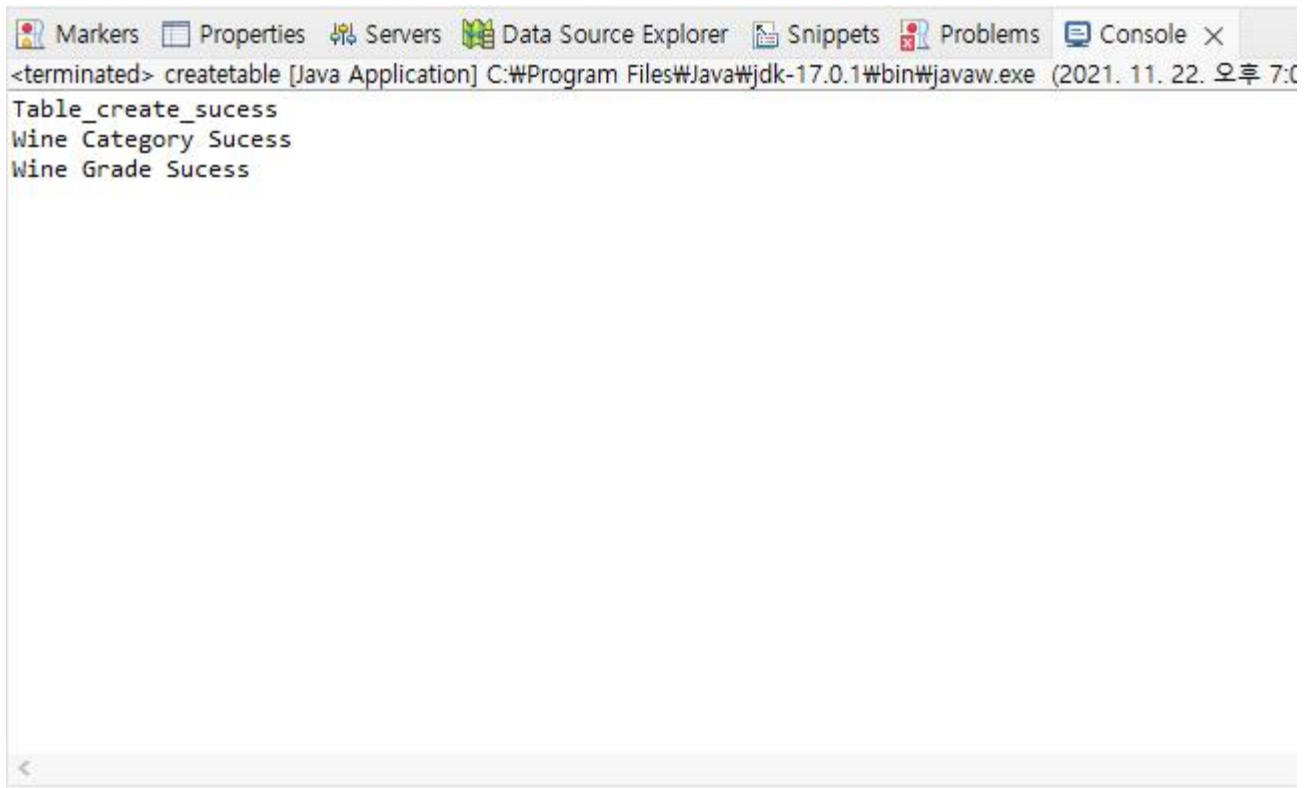
        for (int i = 0; i < 3; i++) {
            preStmt.setString(1, Gradetitle[i]);
            preStmt.setDouble(2, ratio[i]);
            preStmt.executeUpdate();
            preStmt.clearParameters();
        }

        System.out.println("Wine Grade Sucess");

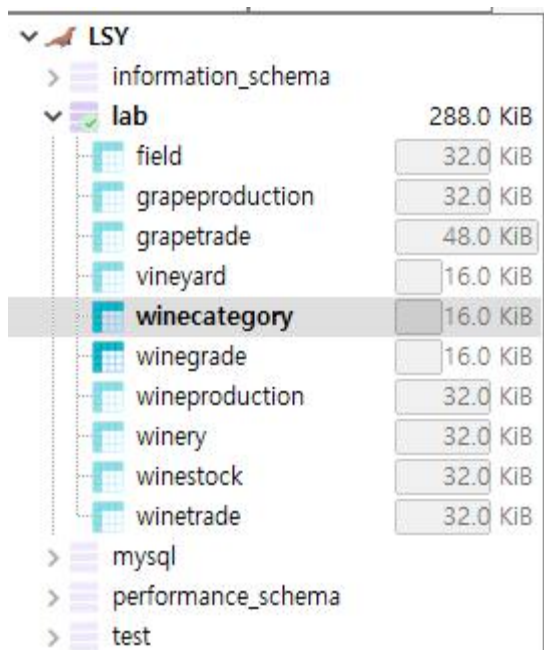
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        try {
            stmt.close();
            conn.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
}

```

2. 실행 결과



```
<terminated> createtable [Java Application] C:\Program Files\Java\jdk-17.0.1\bin\javaw.exe (2021. 11. 22. 오후 7:00)
Table_create_sucess
Wine Category Sucess
Wine Grade Sucess
```



LSY	
information_schema	
lab	288.0 KiB
field	32.0 KiB
grapeproduction	32.0 KiB
grapetrade	48.0 KiB
vineyard	16.0 KiB
winecategory	16.0 KiB
winegrade	16.0 KiB
wineproduction	32.0 KiB
winery	32.0 KiB
winestock	32.0 KiB
winetrade	32.0 KiB
mysql	
performance_schema	
test	

데이터베이스 필터 테이블 필터

호스트: 127.0.0.1 데이터베이스: lab 테이블: winecategory 데이터 쿼리

LSY

- information_schema
- lab
 - field 32.0 KiB
 - grapeproduction 32.0 KiB
 - grapetrade 48.0 KiB
 - vineyard 16.0 KiB
 - winecategory 16.0 KiB
 - winegrade 16.0 KiB
 - wineproduction 32.0 KiB
 - winery 32.0 KiB
 - winestock 32.0 KiB
 - winetrade 32.0 KiB
- mysql
- performance_schema
- test
- yonseivalley

lab.winecategory: 4 행 (중) (대략적)

categoryID	title	standardPrice
1	red	10
2	white	10
3	blush	14
4	sparkling	20

필터: 정규 표현식

```
31 SELECT * FROM information_schema.REFERENTIAL_CONSTRAINTS WHERE CONSTRAINT_SCHEMA='lab' AND TABLE_NAME='winecategory';
32 SELECT * FROM information_schema.KEY_COLUMN_USAGE WHERE TABLE_SCHEMA='lab' AND TABLE_NAME='winecategory' AND REFERENC
33 SHOW CREATE TABLE `lab`.`winecategory`;
34 SELECT CONSTRAINT_NAME, CHECK_CLAUSE FROM `information_schema`.`CHECK_CONSTRAINTS` WHERE CONSTRAINT_SCHEMA='lab' AND TABLE_
35 SELECT * FROM `lab`.`winecategory` LIMIT 1000;
```

Undo r1 : c1 연결됨: 00:01 MariaDB 10.5.13 가동 시간: 1 일, 06:05 h 서버 시간: 오 유류

데이터베이스 필터 테이블 필터

호스트: 127.0.0.1 데이터베이스: lab 테이블: winegrade 데이터 쿼리

LSY

- information_schema
- lab
 - field 32.0 KiB
 - grapeproduction 32.0 KiB
 - grapetrade 48.0 KiB
 - vineyard 16.0 KiB
 - winecategory 16.0 KiB
 - winegrade 16.0 KiB
 - wineproduction 32.0 KiB
 - winery 32.0 KiB
 - winestock 32.0 KiB
 - winetrade 32.0 KiB
- mysql
- performance_schema
- test
- yonseivalley

lab.winegrade: 3 행 (중) (대략적)

gradeID	title	ratio
1	1st	2.0
2	2nd	1.0
3	3rd	0.5

필터: 정규 표현식

```
33 SHOW CREATE TABLE `lab`.`winecategory`;
34 SELECT CONSTRAINT_NAME, CHECK_CLAUSE FROM `information_schema`.`CHECK_CONSTRAINTS` WHERE CONSTRAINT_SCHEMA='lab' AND TABLE_
35 SELECT * FROM `lab`.`winecategory` LIMIT 1000;
36 SHOW CREATE TABLE `lab`.`winegrade`;
37 SELECT * FROM `lab`.`winegrade` LIMIT 1000;
```

r1 : c1 연결됨: 00:01 MariaDB 10.5.13 가동 시간: 1 일, 06:05 h 서버 시간: 오 유류