

// 连接

getConnection();

private static void getConnection() throws SQLException {

String url = "jdbc:mysql://localhost:3306/books?characterEncoding=UTF-8&serverTimezone=Asia/Shanghai&ssl=true";

String username = "root";

// Exception in thread "main" java.sql.SQLException: Access denied for user 'root'@'localhost' (using password: YES)

String password = "000000";

Connection connection = DriverManager.getConnection(url, username, password);

if (connection != null){

System.out.println("数据库连接成功");

}else{

System.out.println("数据库连接失败");

}

}

CREATE TABLE `test` (

`id` int(20) NOT NULL AUTO\_INCREMENT,

`username` varchar(50) NOT NULL,

`email` varchar(50) NOT NULL,

`info` varchar(255) DEFAULT NULL,

PRIMARY KEY (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

新增

addSql();

private static void addSql() throws SQLException {

String url = "jdbc:mysql://localhost:3306/books?characterEncoding=UTF-8&serverTimezone=Asia/Shanghai&ssl=true";

String username = "root";

// Exception in thread "main" java.sql.SQLException: Access denied for user 'root'@'localhost' (using password: YES)

String password = "000000";

Connection connection = DriverManager.getConnection(url, username, password);

if (connection != null) {

System.out.println("数据库连接成功");

} else {

System.out.println("数据库连接失败");

}

// 创建一个基本的statement对象，用于执行静态的sql语句

Statement statement = connection.createStatement();

// 准备新增的sql语句

String sql = "insert into test(username,email,info) values('admin','admin@qq.com','info')";

int addId = statement.executeUpdate(sql); // 得到的变量是指受到影响的行数(可以在修改和删除中再具体体现)

if (addId > 0) {

System.out.println("新增数据成功：" + addId);

}

}

查询

selectSql();

private static void selectSql() throws SQLException {

String url = "jdbc:mysql://localhost:3306/books?characterEncoding=UTF-8&serverTimezone=Asia/Shanghai&ssl=true";

String username = "root";

// Exception in thread "main" java.sql.SQLException: Access denied for user 'root'@'localhost' (using password: YES)

String password = "000000";

Connection connection = DriverManager.getConnection(url, username, password);

if (connection != null) {

System.out.println("数据库连接成功");

} else {

System.out.println("数据库连接失败");

}

Statement statement = connection.createStatement();

String sql = "select \* from test";

// 作为一个结果集（结果的集合）

ResultSet resultSet = statement.executeQuery(sql);

// 处理结果集，将游标（箭头）移动到下一行，如果有结果返回true，否则返回false

while (resultSet.next()) {

// 处理每一行数据

// 从当前行获取指定列的值，需要注意列的数据类型

int id = resultSet.getInt("id"); // 里面是数据库的字段名

String name = resultSet.getString("username"); // 里面是数据库的字段名

String email = resultSet.getString("email"); // 里面是数据库的字段名

System.out.println("ID：" + id + "，username：" + name + "，email：" + email);

}

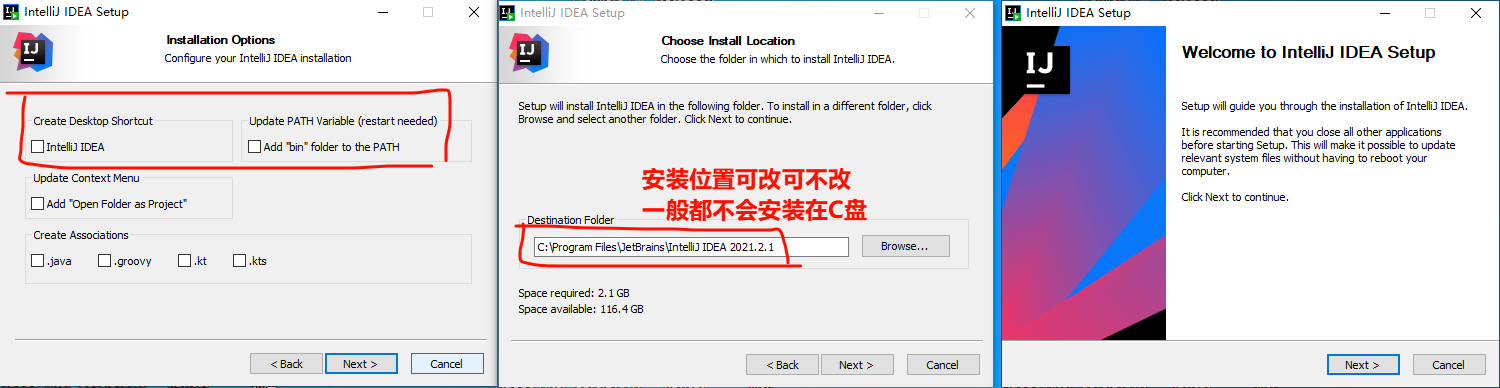
}

Idea下载

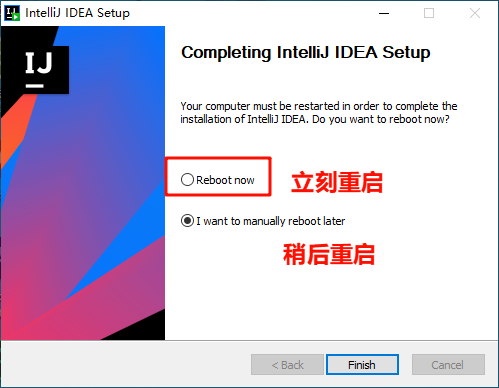
https://www.jetbrains.com/zh-cn/idea/download/?section=windows

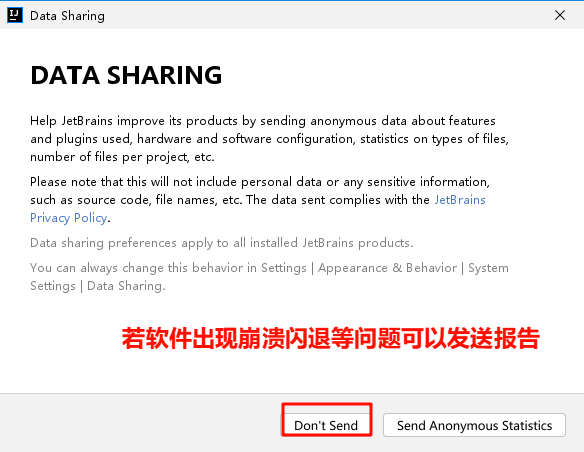


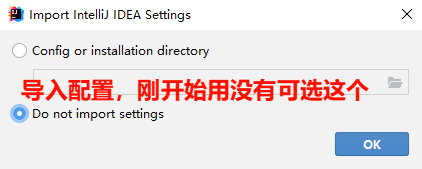
其他版本https://www.jetbrains.com/zh-cn/idea/download/other.html



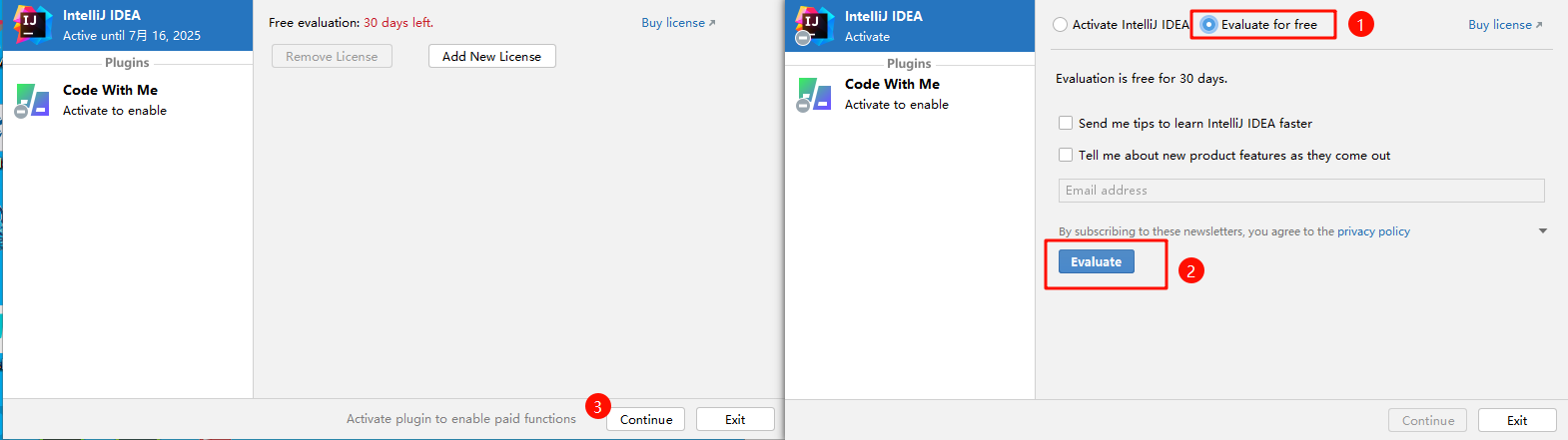
其他默认下一步

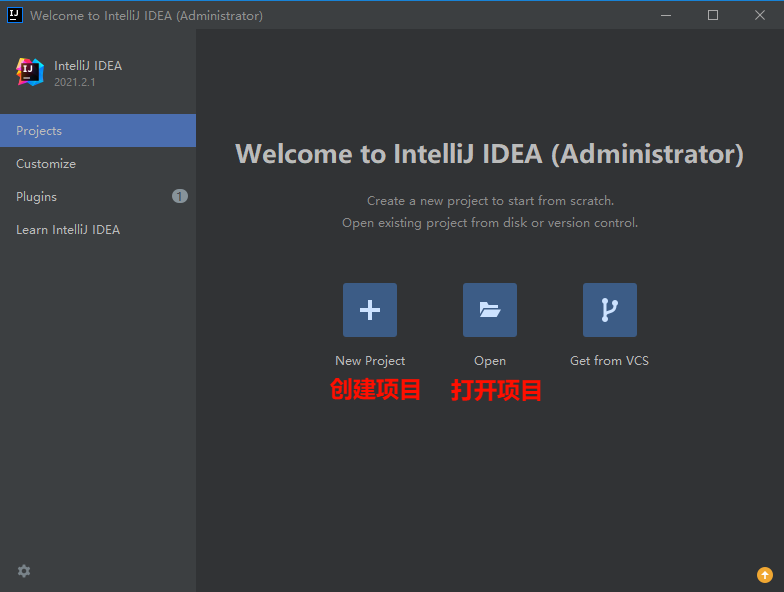


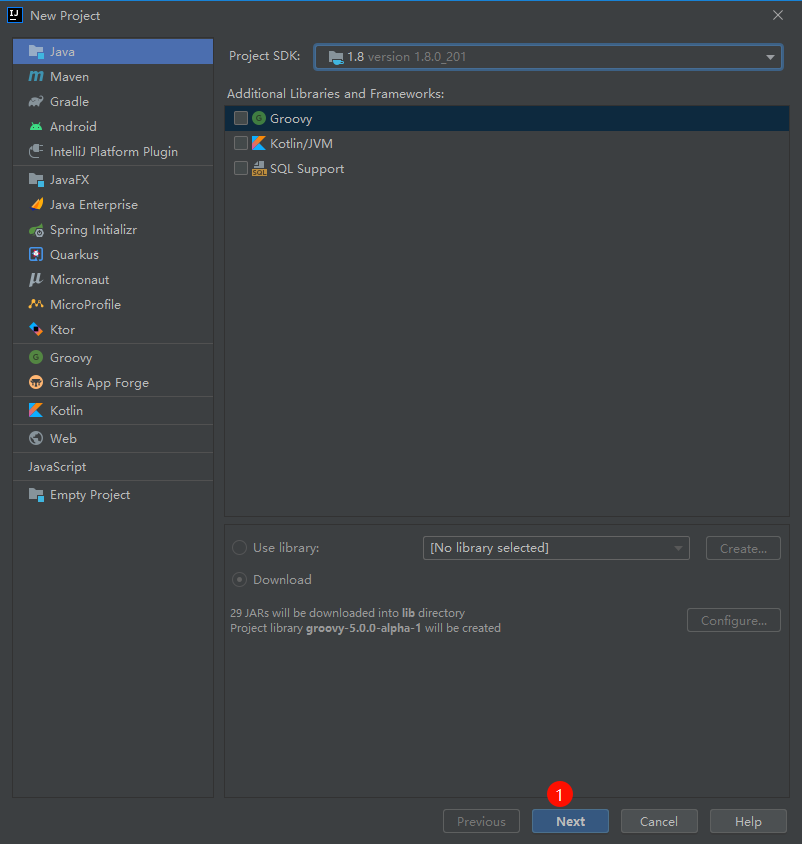


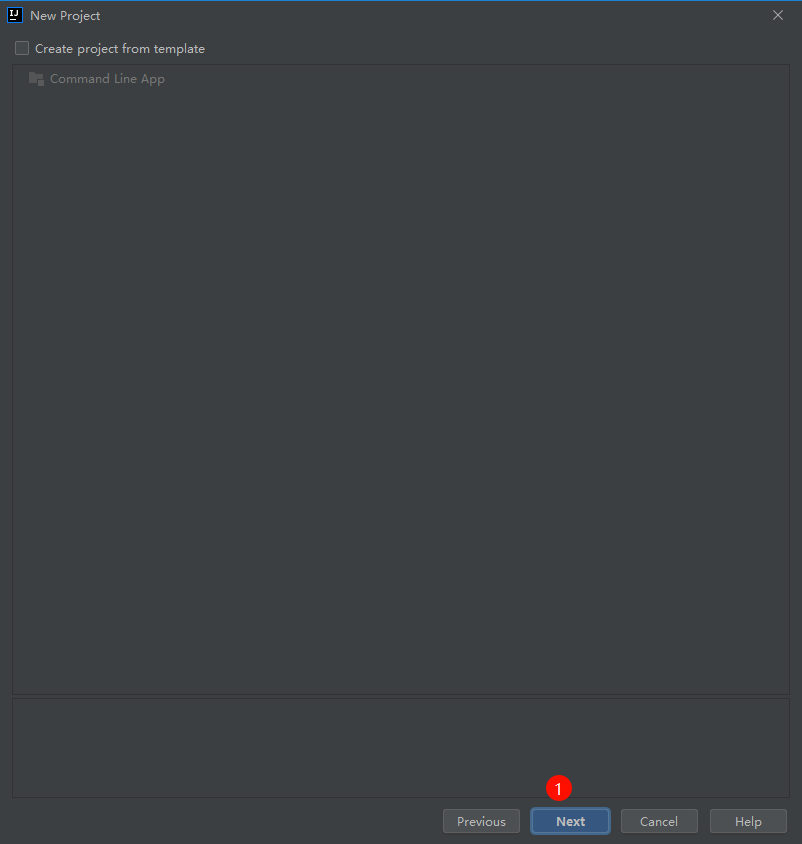


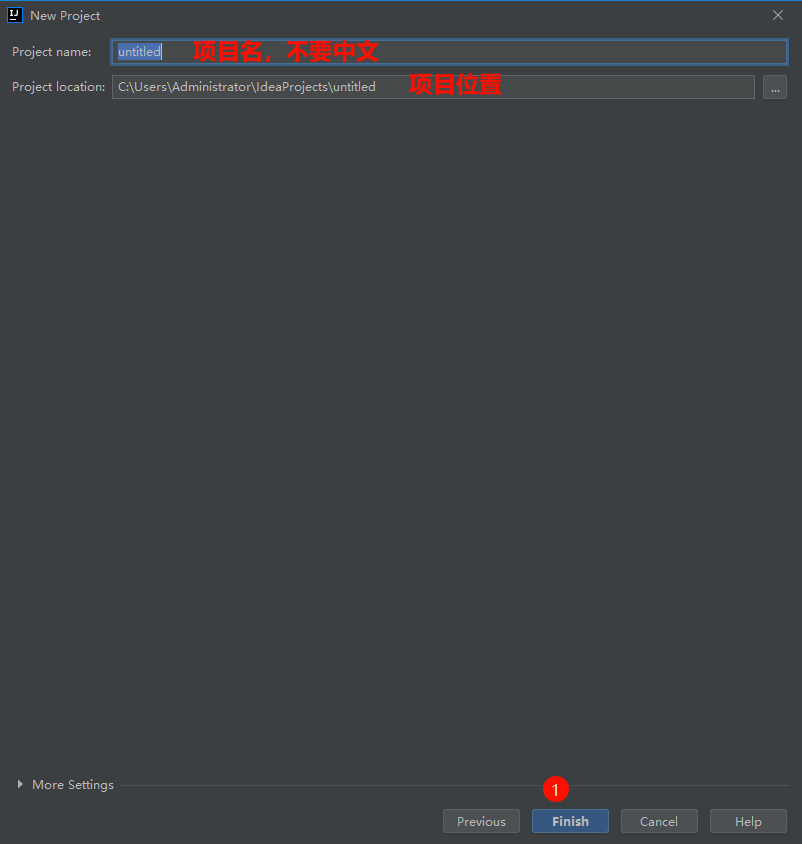
软件会有30天试用，网上有很多重置方法





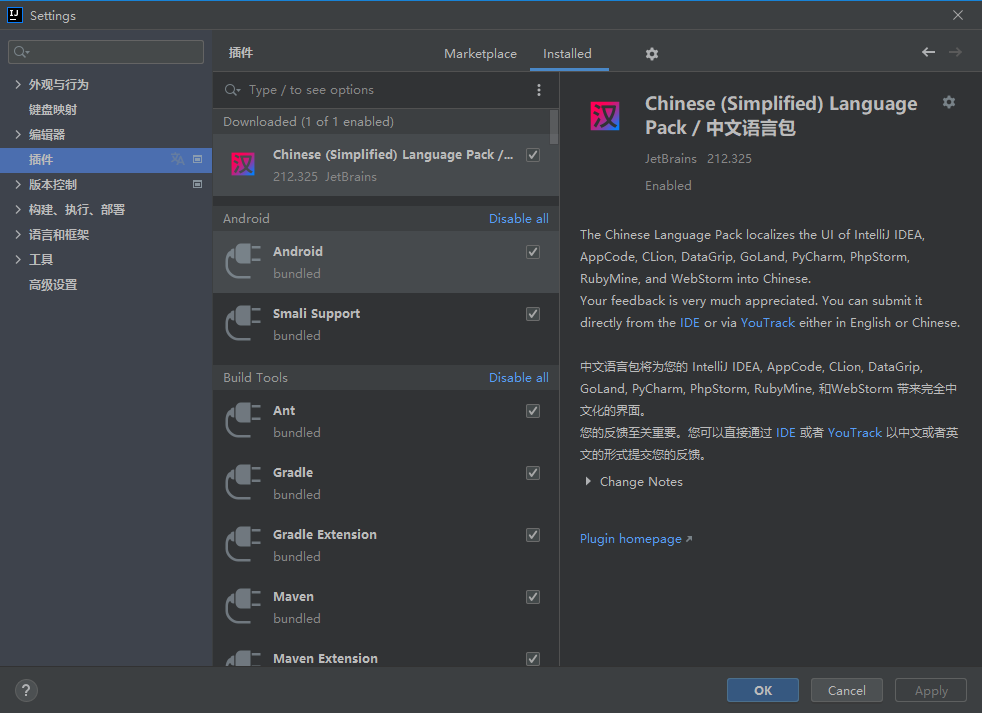






就可以了

刚开始不习惯英文可以安装中文的插件，安装后需要重启软件



删除和修改

deleteSql(); // alt + 回车

updateSql();

private static void updateSql() throws SQLException {

String url = "jdbc:mysql://localhost:3306/books?characterEncoding=UTF-8&serverTimezone=Asia/Shanghai&ssl=true";

String username = "root";

// Exception in thread "main" java.sql.SQLException: Access denied for user 'root'@'localhost' (using password: YES)

String password = "000000";

Connection connection = DriverManager.getConnection(url, username, password);

if (connection != null) {

System.out.println("数据库连接成功");

} else {

System.out.println("数据库连接失败");

}

// 创建一个基本的statement对象，用于执行静态的sql语句

Statement statement = connection.createStatement();

// 准备修改的sql语句

String sql = "update test set email = 'admin@qq.com' where id = 6";

int addId = statement.executeUpdate(sql); // 得到的变量是指受到影响的行数(可以在修改和删除中再具体体现)

if (addId > 0) {

System.out.println("修改数据成功：" + addId);

}

}

private static void deleteSql() throws SQLException {

String url = "jdbc:mysql://localhost:3306/books?characterEncoding=UTF-8&serverTimezone=Asia/Shanghai&ssl=true";

String username = "root";

// Exception in thread "main" java.sql.SQLException: Access denied for user 'root'@'localhost' (using password: YES)

String password = "000000";

Connection connection = DriverManager.getConnection(url, username, password);

if (connection != null) {

System.out.println("数据库连接成功");

} else {

System.out.println("数据库连接失败");

}

// 创建一个基本的statement对象，用于执行静态的sql语句

Statement statement = connection.createStatement();

// 准备删除的sql语句

String sql = "delete from test where id = 5";

int addId = statement.executeUpdate(sql); // 得到的变量是指受到影响的行数(可以在修改和删除中再具体体现)

if (addId > 0) {

System.out.println("删除数据成功：" + addId);

}

}

Sql注入

// 问题

System.out.println(sqlStatement("admin1","123")); // 正常查询

// TODO sql注入问题

System.out.println(sqlStatement("admin1' -- ","123"));

// 生成的sql语句：select \* from test where username = 'admin1' -- ' and password = '123'

private static boolean sqlStatement(String username, String password) throws SQLException {

String url = "jdbc:mysql://localhost:3306/books?characterEncoding=UTF-8&serverTimezone=Asia/Shanghai&ssl=true";

String user = "root";

// Exception in thread "main" java.sql.SQLException: Access denied for user 'root'@'localhost' (using password: YES)

String pwd = "000000";

Connection connection = DriverManager.getConnection(url, user, pwd);

if (connection != null) {

System.out.println("数据库连接成功");

} else {

System.out.println("数据库连接失败");

}

String sql = "select \* from test where username = '" + username + "' and password = '" +password+"'";

Statement statement = connection.createStatement();

ResultSet resultSet = statement.executeQuery(sql);

return resultSet.next();

}

预处理

addSqlStatement();

private static void addSqlStatement() throws SQLException {

String url = "jdbc:mysql://localhost:3306/books?characterEncoding=UTF-8&serverTimezone=Asia/Shanghai&ssl=true";

String user = "root";

// Exception in thread "main" java.sql.SQLException: Access denied for user 'root'@'localhost' (using password: YES)

String pwd = "000000";

Connection connection = DriverManager.getConnection(url, user, pwd);

if (connection != null) {

System.out.println("数据库连接成功");

} else {

System.out.println("数据库连接失败");

}

// 这个是之前的

// String sql = "insert into test(username,email,info) values('admin','admin@qq.com','info')";

// 这个是现在的

// 变动1：使用英文问号代替值，作为一个占位符

String sql = "insert into test(username,password,email,info) values(?,?,?,?)";

// 变动2：预处理的方法名

// 这个是之前的

// Statement statement = connection.createStatement();

// 这个是现在的

// 得到一个预处理的对象

PreparedStatement preparedStatement = connection.prepareStatement(sql);

// 变动3：为占位符设置值

// 需要注意值的数据类型，参数一是索引，从1开始，参数二是值

// 还有就是顺序不能乱，如username 对应 1,"zhangsan"

preparedStatement.setString(1,"zhangsan");

preparedStatement.setString(2,"123");

preparedStatement.setString(3,"zhangsan@qq.com");

preparedStatement.setString(4,"测试预处理");

int addId = preparedStatement.executeUpdate(); // 得到的变量是指受到影响的行数(可以在修改和删除中再具体体现)

if (addId > 0) {

System.out.println("新增数据成功：" + addId);

}

}

selectSqlStatement();

private static void selectSqlStatement() throws SQLException {

String url = "jdbc:mysql://localhost:3306/books?characterEncoding=UTF-8&serverTimezone=Asia/Shanghai&ssl=true";

String user = "root";

// Exception in thread "main" java.sql.SQLException: Access denied for user 'root'@'localhost' (using password: YES)

String pwd = "000000";

Connection connection = DriverManager.getConnection(url, user, pwd);

if (connection != null) {

System.out.println("数据库连接成功");

} else {

System.out.println("数据库连接失败");

}

String sql = "select \* from test where id = ?";

PreparedStatement preparedStatement = connection.prepareStatement(sql);

preparedStatement.setInt(1, 7);

ResultSet resultSet = preparedStatement.executeQuery();

while (resultSet.next()) {

int id = resultSet.getInt("id");

String name = resultSet.getString("username");

String password = resultSet.getString("password");

String email = resultSet.getString("email");

System.out.println("ID：" + id + "，username：" + name + "，password：" + password + "，email：" + email);

}

}

updateSqlStatement();

private static void updateSqlStatement() throws SQLException {

String url = "jdbc:mysql://localhost:3306/books?characterEncoding=UTF-8&serverTimezone=Asia/Shanghai&ssl=true";

String user = "root";

// Exception in thread "main" java.sql.SQLException: Access denied for user 'root'@'localhost' (using password: YES)

String pwd = "000000";

Connection connection = DriverManager.getConnection(url, user, pwd);

if (connection != null) {

System.out.println("数据库连接成功");

} else {

System.out.println("数据库连接失败");

}

// 这个是之前的

// String sql = "insert into test(username,email,info) values('admin','admin@qq.com','info')";

// 这个是现在的

// 变动1：使用英文问号代替值，作为一个占位符

String sql = "update test set email = ? where id = ?";

// 变动2：预处理的方法名

// 这个是之前的

// Statement statement = connection.createStatement();

// 这个是现在的

// 得到一个预处理的对象

// 它将sql语句预先发送到数据库中进行编译

PreparedStatement preparedStatement = connection.prepareStatement(sql);

// 变动3：为占位符设置值

// 需要注意值的数据类型，参数一是索引，从1开始，参数二是值

// 还有就是顺序不能乱，如username 对应 1,"zhangsan"

preparedStatement.setString(1, "test@qq.com");

preparedStatement.setInt(2, 7);

int addId = preparedStatement.executeUpdate(); // 得到的变量是指受到影响的行数(可以在修改和删除中再具体体现)

if (addId > 0) {

System.out.println("修改数据成功：" + addId);

}

}

案例准备

建表sql

演示

CREATE TABLE `users` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`username` varchar(50) CHARACTER SET utf8mb4 NOT NULL,

`password` varchar(50) CHARACTER SET utf8mb4 NOT NULL,

`email` varchar(50) CHARACTER SET utf8mb4 NOT NULL,

`role` varchar(50) CHARACTER SET utf8mb4 DEFAULT 'user',

PRIMARY KEY (`id`),

UNIQUE KEY `username` (`username`),

UNIQUE KEY `email` (`email`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

完整建表语句

CREATE TABLE `users` (

`id` int(0) NOT NULL AUTO\_INCREMENT,

`username` varchar(50) CHARACTER SET utf8mb4 NOT NULL,

`password` varchar(100) CHARACTER SET utf8mb4 NOT NULL,

`email` varchar(100) CHARACTER SET utf8mb4 NOT NULL,

`role` enum('admin','user') CHARACTER SET utf8mb4 NULL DEFAULT 'user',

`created\_at` timestamp(0) NULL DEFAULT CURRENT\_TIMESTAMP(0),

PRIMARY KEY (`id`) USING BTREE,

UNIQUE INDEX `username`(`username`) USING BTREE,

UNIQUE INDEX `email`(`email`) USING BTREE

) ENGINE = InnoDB AUTO\_INCREMENT = 3 CHARACTER SET = utf8mb4 ROW\_FORMAT = Dynamic;

CREATE TABLE `books` (

`id` int(0) NOT NULL AUTO\_INCREMENT,

`title` varchar(100) CHARACTER SET utf8mb4 NOT NULL,

`author` varchar(100) CHARACTER SET utf8mb4 NOT NULL,

`isbn` varchar(20) CHARACTER SET utf8mb4 NOT NULL,

`publisher` varchar(100) CHARACTER SET utf8mb4 NULL DEFAULT NULL,

`publish\_date` date NULL DEFAULT NULL,

`category` varchar(50) CHARACTER SET utf8mb4 NULL DEFAULT NULL,

`quantity` int(0) NULL DEFAULT 1,

`created\_at` timestamp(0) NULL DEFAULT CURRENT\_TIMESTAMP(0),

PRIMARY KEY (`id`) USING BTREE,

UNIQUE INDEX `isbn`(`isbn`) USING BTREE

) ENGINE = InnoDB AUTO\_INCREMENT = 3 CHARACTER SET = utf8mb4 ROW\_FORMAT = Dynamic;

CREATE TABLE `borrow\_records` (

`id` int(0) NOT NULL AUTO\_INCREMENT,

`user\_id` int(0) NOT NULL,

`book\_id` int(0) NOT NULL,

`borrow\_date` date NOT NULL,

`return\_date` date NULL DEFAULT NULL,

`status` enum('borrowed','returned') CHARACTER SET utf8mb4 NULL DEFAULT 'borrowed',

PRIMARY KEY (`id`) USING BTREE,

INDEX `user\_id`(`user\_id`) USING BTREE,

INDEX `book\_id`(`book\_id`) USING BTREE,

CONSTRAINT `borrow\_records\_ibfk\_1` FOREIGN KEY (`user\_id`) REFERENCES `users` (`id`) ON DELETE RESTRICT ON UPDATE RESTRICT,

CONSTRAINT `borrow\_records\_ibfk\_2` FOREIGN KEY (`book\_id`) REFERENCES `books` (`id`) ON DELETE RESTRICT ON UPDATE RESTRICT

) ENGINE = InnoDB AUTO\_INCREMENT = 1 CHARACTER SET = utf8mb4 ROW\_FORMAT = Dynamic;

封装数据库工具类（提取共有内容）

// 数据库的工具类

public class DBUtil {

// 私有 静态 不可改

private static final String url = "jdbc:mysql://localhost:3306/books?characterEncoding=UTF-8&serverTimezone=Asia/Shanghai&ssl=true";

private static final String user = "root";

private static final String password = "000000";

// 这里需要在类加载的时候执行

static {

try {

Class.forName("com.mysql.jdbc.Driver");

} catch (ClassNotFoundException e) {

e.printStackTrace();

}

}

// 获取数据库连接的方法

public static Connection getConnection() throws SQLException {

return DriverManager.getConnection(url,user,password);

}

// 关闭数据库连接的方法

public static void close(Connection conn , Statement stmt, ResultSet rs){

try {

if (conn != null) conn.close();

if (stmt != null) stmt.close();

if (rs != null) rs.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

// 执行更新操作的方法

public static int executeUpdate(String sql,Object... params){ // Object... params是可变参数的设置，可以0个、1个、2个等

Connection conn = null;

PreparedStatement stmt = null;

try{

conn = getConnection();

stmt = conn.prepareStatement(sql);

// 遍历参数数组，将参数设置到stmt中

for (int i = 0; i < params.length; i++) {

stmt.setObject(i+1,params[i]);

}

// 执行更修操作，返回受影响行数

return stmt.executeUpdate();

}catch (Exception e){

e.printStackTrace();

return -1;

}finally {

close(conn,stmt,null);

}

}

// 执行查询操作的方法

public static ResultSet executeQuery(String sql,Object... params){ // Object... params是可变参数的设置，可以0个、1个、2个等

Connection conn = null;

PreparedStatement stmt = null;

try{

conn = getConnection();

stmt = conn.prepareStatement(sql);

// 遍历参数数组，将参数设置到stmt中

for (int i = 0; i < params.length; i++) {

stmt.setObject(i+1,params[i]);

}

// 执行更修操作，返回受影响行数

return stmt.executeQuery();

}catch (Exception e){

e.printStackTrace();

return null;

}

}

}