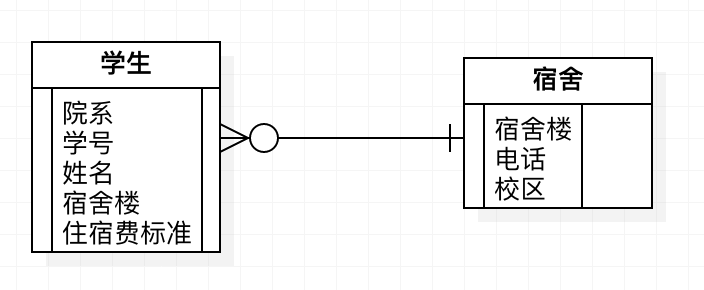
具体maven工程及代码见https://github.com/xiekailian/dbHomework2

第一题

1、



2、

public void createTables() {

try {

Connection connection = DBConfig.getConn();

Statement stmt = (Statement) connection.createStatement();

String sql ="CREATE TABLE IF NOT EXISTS `students`(\n" +

" `college\_name` VARCHAR(40) NOT NULL,\n" +

" `student\_num` VARCHAR(20) NOT NULL,\n" +

" `student\_name` VARCHAR(20) NOT NULL,\n" +

" `sex` VARCHAR(10) NOT NULL,\n" +

" `dormitory\_num` VARCHAR(20) NOT NULL,\n" +

" PRIMARY KEY ( `student\_num` )\n" +

")ENGINE=InnoDB DEFAULT CHARSET=utf8;\n" ;

stmt.execute(sql);

sql ="CREATE TABLE IF NOT EXISTS `dormitories`(\n" +

" `dormitory\_num` VARCHAR(20) NOT NULL,\n" +

" `dormitory\_phone` VARCHAR(20) NOT NULL, \n" +

" `campus\_name` VARCHAR(20) NOT NULL, \n" +

" `dormitory\_payment` int(11),\n" +

" PRIMARY KEY ( `dormitory\_num` )\n" +

")ENGINE=InnoDB DEFAULT CHARSET=utf8;\n" ;

stmt.execute(sql);

stmt.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

3、

public void initData() {

try {

Connection connection = DBConfig.getConn();

Statement stmt = (Statement) connection.createStatement();

String sql ="load data local infile '/Users/kailian/Desktop/sharingBike/src/main/java/data/phone.txt' into table dormitories character set utf8 fields terminated by';';" ;

stmt.execute(sql);

sql ="load data local infile '/Users/kailian/Desktop/sharingBike/src/main/java/data/arrange\_plan.txt' into table students character set utf8 fields terminated by',';" ;

stmt.execute(sql);

stmt.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

4、

public void question4() {

try {

Connection connection = DBConfig.getConn();

Statement stmt = (Statement) connection.createStatement();

String sql ="SELECT DISTINCT(college\_name) FROM students WHERE dormitory\_num IN (SELECT dormitory\_num FROM students WHERE student\_name = '王小星');";

stmt.execute(sql);

stmt.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

5、

public void question5() {

try {

Connection connection = DBConfig.getConn();

Statement stmt = (Statement) connection.createStatement();

String sql ="UPDATE dormitories SET dormitory\_payment = 1200 WHERE dormitory\_num = '陶园1舍';\n";

stmt.execute(sql);

stmt.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

6、

public void question6() {

try {

Connection connection = DBConfig.getConn();

Statement stmt = (Statement) connection.createStatement();

String sql ="UPDATE students SET dormitory\_num = '陶园1舍' WHERE college\_name = '软件学院' AND sex = '男';\n";

stmt.execute(sql);

sql ="UPDATE students SET dormitory\_num = '南园1舍' WHERE college\_name = '软件学院' AND sex = '女';\n";

stmt.execute(sql);

stmt.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

第二题

1、

public void createTables() {

try {

Connection connection = DBConfig.getConn();

Statement stmt = (Statement) connection.createStatement();

String sql = "CREATE TABLE IF NOT EXISTS `users`(\n" +

" `user\_id` int(11), \n" +

" `user\_name` VARCHAR(20) NOT NULL,\n" +

" `user\_phone` VARCHAR(20) NOT NULL,\n" +

" `user\_address` VARCHAR(20) NOT NULL,\n" +

" `money` double NOT NULL,\n" +

" PRIMARY KEY ( `user\_id` )\n" +

")ENGINE=InnoDB DEFAULT CHARSET=utf8;" ;

stmt.execute(sql);

sql = "CREATE TABLE IF NOT EXISTS `orders`(\n" +

" `user\_id` int(11) NOT NULL, \n" +

" `bike\_id` int(11) NOT NULL, \n" +

" `start\_address` VARCHAR(20) NOT NULL,\n" +

" `start\_time` datetime NOT NULL,\n" +

" `stop\_address` VARCHAR(20) NOT NULL,\n" +

" `end\_time` datetime NOT NULL,\n" +

" `cost` double NOT NULL\n" +

")ENGINE=InnoDB DEFAULT CHARSET=utf8;";

stmt.execute(sql);

sql = "CREATE TABLE IF NOT EXISTS `bikes`(\n" +

" `bike\_id` int(11),\n" +

" `is\_work` boolean, \n" +

" `time\_last\_month` int(11),\n" +

" `last\_address` VARCHAR(20),\n" +

" PRIMARY KEY ( `bike\_id` )\n" +

")ENGINE=InnoDB DEFAULT CHARSET=utf8;";

stmt.execute(sql);

stmt.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

public void initData(){

try {

PreparedStatement pstmt = null;

File file = new File("src/main/java/data/user.txt");

String sql = "INSERT INTO `users` (`user\_id`,`user\_name`,`user\_phone`,`money`,`user\_address`) VALUES (?,?,?,?,?)";

pstmt = (PreparedStatement) DBConfig.getConn().prepareStatement(sql);

BufferedReader bufferedReader = new BufferedReader(new FileReader(file));

while (bufferedReader.readLine()!=null){

String array[] = bufferedReader.readLine().split(";");

pstmt.setInt(1,Integer.parseInt(array[0]));

pstmt.setString(2,array[1]);

pstmt.setString(3,array[2]);

pstmt.setDouble(4,Double.parseDouble(array[3]));

pstmt.setString(5,"");

pstmt.execute();

}

file = new File("src/main/java/data/record.txt");

sql = "INSERT INTO `orders` (`user\_id`,`bike\_id`,`start\_address`,`start\_time`,`stop\_address`,`end\_time`,`cost`) VALUES (?,?,?,?,?,?,?)";

pstmt = (PreparedStatement) DBConfig.getConn().prepareStatement(sql);

bufferedReader = new BufferedReader(new FileReader(file));

while (bufferedReader.readLine()!=null){

String array[] = bufferedReader.readLine().replaceAll(""+(char)65279,"").split(";");

pstmt.setInt(1,Integer.parseInt(array[0]));

pstmt.setInt(2,Integer.parseInt(array[1]));

pstmt.setString(3,array[2]);

pstmt.setTimestamp(4, Timestamp.valueOf(array[3].replace("-"," ").replace("/","-")));

pstmt.setString(5,array[4]);

pstmt.setTimestamp(6, Timestamp.valueOf(array[5].replace("-"," ").replace("/","-")));

pstmt.setDouble(7,0);

pstmt.execute();

}

file = new File("src/main/java/data/bike.txt");

sql = "INSERT INTO `bikes` (`bike\_id`,`is\_work`) VALUES (?,?)";

pstmt = (PreparedStatement) DBConfig.getConn().prepareStatement(sql);

bufferedReader = new BufferedReader(new FileReader(file));

while (bufferedReader.readLine()!=null){

String array[] = bufferedReader.readLine().split(";");

pstmt.setInt(1,Integer.parseInt(array[0]));

pstmt.setBoolean(2,true);

pstmt.execute();

}

pstmt.close();

} catch (SQLException e) {

e.printStackTrace();

} catch (FileNotFoundException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

}

}

2、

public void estimateUserAddress() {

Statement stmt;

try {

stmt= (Statement) DBConfig.getConn().createStatement();

stmt.execute("CREATE TEMPORARY TABLE ua SELECT user\_id,stop\_address,COUNT(user\_id) AS num FROM `orders` WHERE DATE\_FORMAT(end\_time,'%H')>=18 AND DATE\_FORMAT(end\_time,'%H')<=24 GROUP BY user\_id,stop\_address;");

stmt.execute("CREATE TEMPORARY TABLE ub SELECT user\_id,MAX(num) as max\_num FROM ua GROUP BY user\_id;");

stmt.execute("CREATE TEMPORARY TABLE uc select ua.user\_id,ua.stop\_address FROM ua,ub WHERE ua.user\_id = ub.user\_id AND ua.num = ub.max\_num; ");

stmt.execute("UPDATE users,uc SET user\_address = stop\_address WHERE users.user\_id = uc.user\_id;");

stmt.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

3、

public void countOrders() {

Statement stmt;

try {

stmt= (Statement) DBConfig.getConn().createStatement();

stmt.execute("UPDATE orders SET cost = 1 WHERE UNIX\_TIMESTAMP(end\_time) - UNIX\_TIMESTAMP(start\_time)<= 1800 ;");

stmt.execute("UPDATE orders SET cost = 2 WHERE UNIX\_TIMESTAMP(end\_time) - UNIX\_TIMESTAMP(start\_time)> 1800 AND UNIX\_TIMESTAMP(end\_time) - UNIX\_TIMESTAMP(start\_time)<= 3600;");

stmt.execute("UPDATE orders SET cost = 3 WHERE UNIX\_TIMESTAMP(end\_time) - UNIX\_TIMESTAMP(start\_time)> 3600 AND UNIX\_TIMESTAMP(end\_time) - UNIX\_TIMESTAMP(start\_time)<= 5400;");

stmt.execute("UPDATE orders SET cost = 4 WHERE UNIX\_TIMESTAMP(end\_time) - UNIX\_TIMESTAMP(start\_time)> 5400 ;");

stmt.execute("UPDATE users,orders SET users.money = users.money - orders.cost WHERE users.user\_id = orders.user\_id;");

stmt.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

4、

public void banAndPreserveBikes() {

Statement stmt;

try {

stmt= (Statement) DBConfig.getConn().createStatement();

stmt.execute("UPDATE bikes SET time\_last\_month = 0;");

stmt.execute("UPDATE bikes,orders SET bikes.time\_last\_month = bikes.time\_last\_month + UNIX\_TIMESTAMP(end\_time) - UNIX\_TIMESTAMP(start\_time) WHERE bikes.bike\_id = orders.bike\_id AND DATE\_FORMAT(NOW(),'%m')-DATE\_FORMAT(end\_time,'%m')=1;");

stmt.execute("UPDATE bikes SET is\_work = FALSE WHERE time\_last\_month > 200 \* 60 \* 60;");

stmt.execute("UPDATE bikes SET last\_address = (SELECT stop\_address FROM `orders` WHERE bike\_id = bikes.bike\_id ORDER BY end\_time LIMIT 1);");

stmt.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

5、

建立临时表、表连接过多的时候可以考虑一定的冗余存储、使用嵌套子查询代替表连接。

总的运行时间截图：

