**难点**

**1、使用JDBC驱动程序处理元数据**

Java通过JDBC获得连接以后，得到一个Connection对象，可以从这个对象获得有关数据库管理系统的各种信息，包括数据库中的各个表，表中的各个列，数据类型，触发器，存储过程等各方面的信息。根据这个信息，JDBC可以访问一个现实事先并不了解的数据库。

获取这些信息的方法都是在DataBaseMetaData类的对象上实现的，而DataBaseMetaData对象是在Connection对象上获得的。

**2、DataBaseMetaData类（描述ResultSet的元数据对象）**

1. DataBaseMetaData类中提供了许多方法用于获得数据源的各种信息，通过这些方法可以非常详细的了解数据库的信息：

—getURL()：返回一个String类对象，代表数据库的URL。

—getUserName()：返回连接当前数据库管理系统的用户名。

—isReadOnly()：返回一个boolean值，只是数据库是否只允许读操作。

—getDatebaseProductName()：返回数据库的产品名称。

—getDatebaseProductVersion()：返回数据库的版本号。

—getDriverName()：返回驱动程序的名称。

—getDriverVersion()：返回驱动程序的版本号。

1. 可用于获取关于ResultSet对象中列的类型和属性信息的对象：

—getColumnName(int column)：获取指定列的对象。

—getColumnCount()：返回当前ResultSet对象中的列数。

—getColumnLable(int column)：获取指定的列的别名，索引从1开始

—getColumnTypeName(int column)：检索指定列的数据库特定的类型名称。

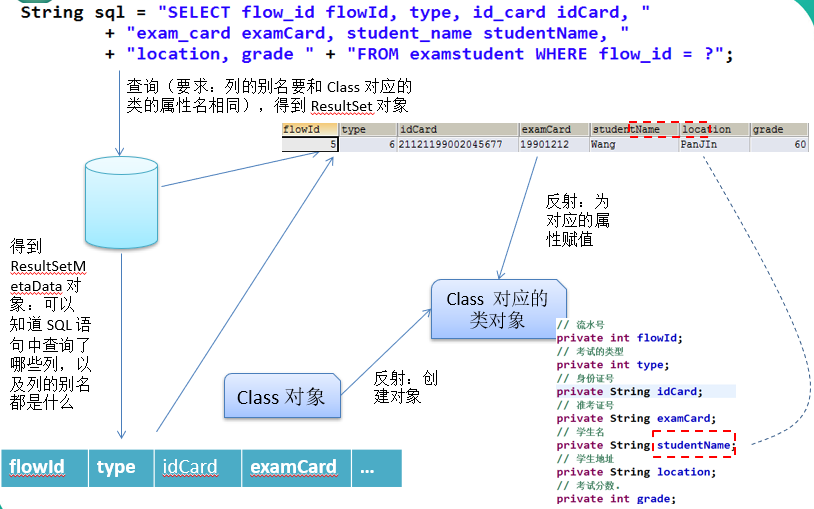
—getColumnDisplaySize(int column)：指示指定列的最大标准宽度，以字符为单位。

—isNullable(int column)：指示指定列中的值是否可以为null。

—isAutoIncrement(int column)：指示是否自动为指定列进行编号，这样这些列任然是只读的。

**3、编写通用查询方法的步骤：**

1. 先利用SQL进行查询，得到结果集；
2. 利用反射创建实体类对象：创建Student对象；
3. 获取结果集的列的别名：idCard、 studentName；
4. 再获取结果集的每一列的值，结合3得到一个Map，键：列的别名，值：列的值。{flowId:5,type:6,idCard:xxx......}；
5. 再利用反射为2）的对应的属性赋值：属性即为Map的键，值即为Map的值。



*/\*\*  
 \* description: 通用的查询方法：根据传入的SQL，Class对象返回SQL对应的记录的对象  
 \*  
 \** ***@param:*** *[clazz, sql, args]  
 \** ***@return:*** *T  
 \** ***@author:*** *nino  
 \** ***@date:*** *2019/3/27 10:10  
 \*/***public** <T> T get(Class<T> clazz, String sql, Object... args) {  
 T entity = **null**;  
  
 Connection conn = **null**;  
 PreparedStatement ps = **null**;  
 ResultSet rs = **null**;  
  
 **try** {  
 *//1.得到ResultSet对象* conn = JDBCUtil.*getConnection*();  
 ps = conn.prepareStatement(sql);  
 *//填充占位符* **for** (**int** i = 0; i < args.**length**; i++) {  
 ps.setObject(i + 1, args[i]);  
 }  
 rs = ps.executeQuery();  
  
 *//2.得到ResultSetMetaData对象* ResultSetMetaData rsmd = rs.getMetaData();  
  
 *//3.创建一个Map<String, Object>对象，键：SQL查询的列的别名，值：列的值* HashMap<String, Object> values = **new** HashMap<>();  
 *//处理结果集，利用ResultSetMetaData填充3对应的Map对象* **if** (rs.next()) {  
 **for** (**int** i = 0; i < rsmd.getColumnCount(); i++) {  
 String columnLabel = rsmd.getColumnLabel(i + 1);  
 Object columnValue = rs.getObject(i + 1);  
  
 values.put(columnLabel, columnValue);  
 }  
 }  
 *//5.若Map不为空集，利用反射创建clazz对应的对象* **if** (values.size() > 0) {  
 entity = clazz.newInstance();  
  
 *//遍历Map对象，再利用反射为Class对象的对应的属性赋值* **for** (Map.Entry<String, Object> entry : values.entrySet()) {  
 String fieldName = entry.getKey();  
 Object fieldvalue = entry.getValue();  
  
 ReflectionUtils.*setFieldValue*(entity, fieldName, fieldvalue);  
 }  
 }  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 } **finally** {  
 JDBCUtil.*releaseDB*(rs, ps, conn);  
 }  
  
 **return** entity;  
}

@Test  
**public void** testGet() {  
 String sql = **"SELECT id,name,email,birthday FROM customers WHERE id=?"**;  
  
 Customer customer = get(Customer.**class**, sql, 4);  
 System.***out***.println(customer);  
  
 *//起别名，对应Student类的属性名* sql = **"SELECT flow\_id flowId,type,id\_card idCard,"** +  
 **"exam\_card examCard,student\_name studentName,"** +  
 **"location,grade "** +  
 **"FROM examstudent WHERE flow\_id=?"**;  
  
 Student student = get(Student.**class**, sql, 10);  
 System.***out***.println(student);  
}