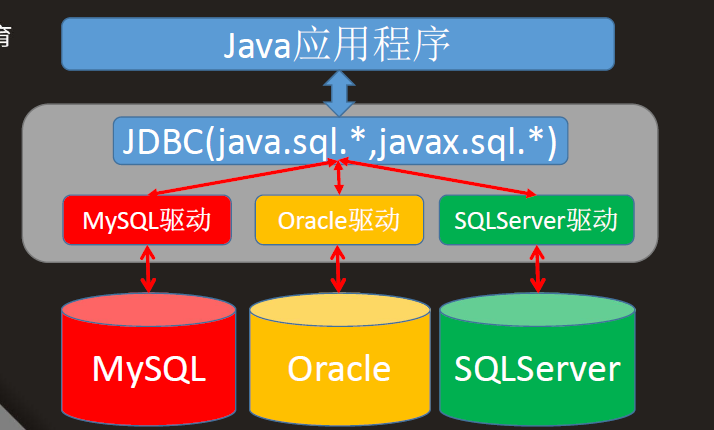
# 概述

## jdbc是什么?

JDBC（Java Data Base Connectivity,java数据库连接)，简单说:就是可以直接通过java语言，去操作数据库。

jdbc是一套标准,它是由一些接口与类组成的。

## JDBC原理示意图



## 什么是驱动?及jdbc驱动位置

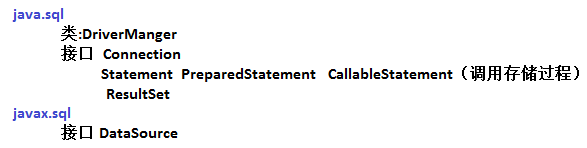
两个设备要进行通信，满足一定通信数据格式，数据格式由设备提供商规定，设备提供商为设备提供驱动软件，

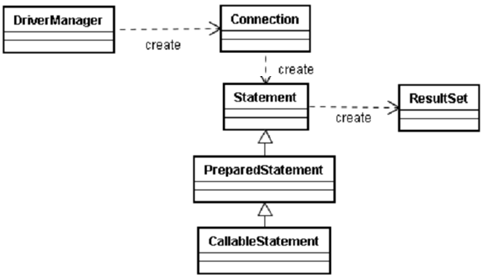
通过软件可以与该设备进行通信

Jdbc驱动位置：

C:\app\administrator\product\11.2.0\dbhome\_1\jdbc

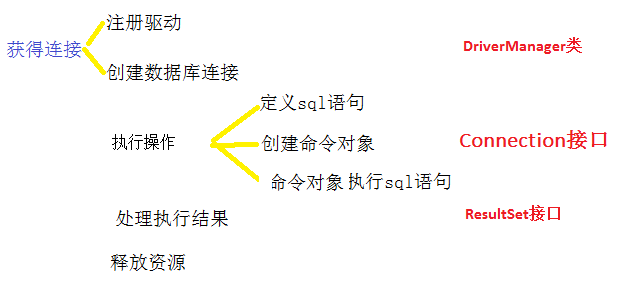
## 构成





# 连接数据库

## 开发步骤



## DriverManager类

### 注册驱动 (只做一次)

Class.forName(“com.mysql.jdbc.Driver”);

DriverManager.registerDriver(com.mysql.jdbc.Driver);

System.setProperty(“jdbc.drivers”, “com.mysql.jdbc.Driver”);

### 创建数据库连接

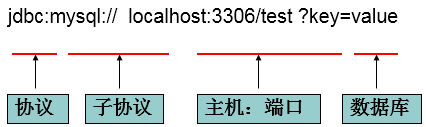
String url = "jdbc:mysql:///one";

String username = "root";

String password = "root";

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

### URL



常用数据库URL地址的写法：

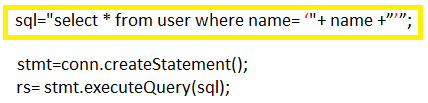
Oracle写法：jdbc:oracle:thin:@localhost:1521:sid

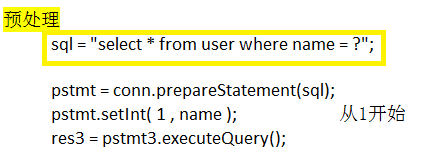
MySql—jdbc:mysql://localhost:3306/user

简写形式： jdbc:mysql:///user

常用属性：useUnicode=true&characterEncoding=UTF-8

## 定义sql语句

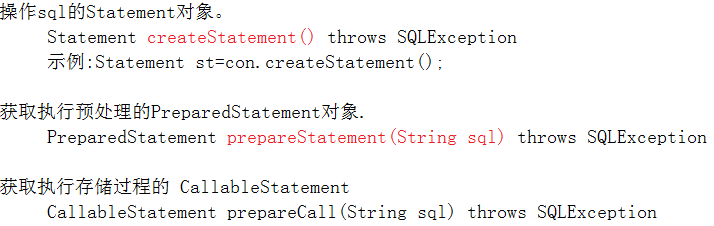




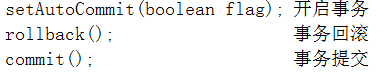
## Connection接口

它代表的是一个连接对象。简单说，就是我们程序与数据库连接。

### 创建命令对象

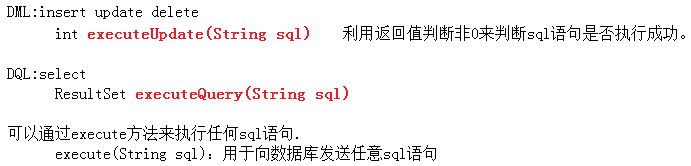


### 操作事务

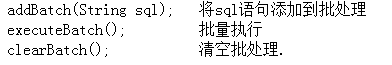


## Statement接口

### 命令对象执行sql语句



### 批处理操作

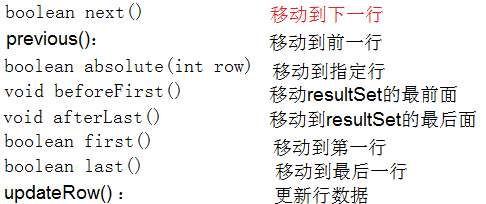


## ResultSet接口

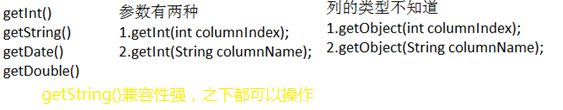
### next()方法

public boolean next();

用于判断是否有下一条记录。如果有返回true,并且让游标向下移动一行。如果没有返回false.



### getXxx()



### 处理执行结果

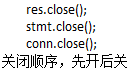


## 释放资源

Jdbc程序运行完后，切记要释放程序在运行过程中，创建的那些与数据库进行交互的对象，这些对象通常是ResultSet, Statement和Connection对象。

特别是Connection对象，它是非常稀有的资源，用完后必须马上释放，如果Connection不能及时、正确的关闭，极易导致系统宕机。Connection的使用原则是尽量晚创建，尽量早的释放。

为确保资源释放代码能运行，资源释放代码也一定要放在finally语句中。



# 注意（面试题）

## Statement 和PreparedStatement 的区别？



## execute、executeUpdate 和executeQuery 方法的区别？

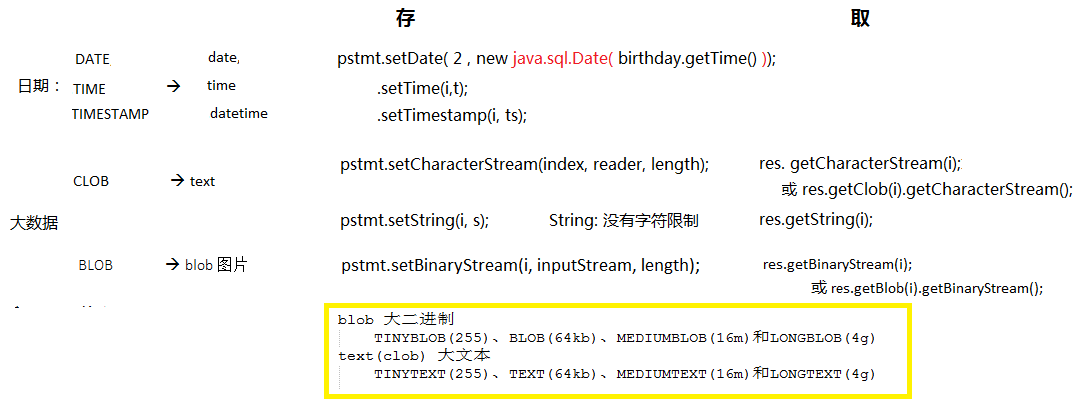
\* 【面试题】execute、executeUpdate、executeQuery三个方法的区别？  
\* 可以执行的SQL语句  
\* execute 执行：任意，特别注意：execute返回返回false说明这条SQL语句执行失败？？？  
\* 返回值是boolean布尔类型，有结果集，返回true，无结果集，返回false  
\* 执行更新语句，执行成功，返回false。失败会抛出异常！！！！！！  
\* 执行结果返回一个ResultSet对象或一个Object，返回true？  
\* executeUpdate执行更新：insert、update、delete等DML、DDL  
\* 返回值是int，更新成功的记录条数  
\* executeQuery执行查询：DQL（select）  
\* 返回值是ResultSet对象，结果集

# 数据类型

## 常用数据类型转换表



## 日期、大数据



public class MyTextTest {

// 存储

@Test

public void save() throws SQLException, FileNotFoundException {

String sql = "insert into mytext values(null,?)";

// 1.获取Connection

Connection con = JdbcUtils.getConnection();

// 2.获取PreparedStatement

PreparedStatement pst = con.prepareStatement(sql);

// 3.插入值

File file = new File("D:\\java1110\\workspace\\day17\_3\\a.txt");

FileReader fr = new FileReader(file);

pst.setCharacterStream(1, fr, (int) (file.length()));

pst.executeUpdate();

// 4.释放资源

pst.close();

con.close();

}

// 获取

@Test

public void get() throws SQLException, IOException {

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

// 1.获取Connection

Connection con = JdbcUtils.getConnection();

// 2.获取PreparedStatement

PreparedStatement pst = con.prepareStatement(sql);

pst.setInt(1, 1);

// 3.得到结果集

ResultSet rs = pst.executeQuery();

// 4.遍历结果集

if (rs.next()) {

Reader r = rs.getCharacterStream("content");

FileWriter fw = new FileWriter("d:/笔记.txt");

int len = -1;

char[] ch = new char[1024 \* 100];

while ((len = r.read(ch)) != -1) {

fw.write(ch, 0, len);

fw.flush();

}

fw.close();

r.close();

}

pst.close();

con.close();

}

}

public class MyBlobTest {

// 添加

@Test

public void save() throws SQLException, IOException {

String sql = "insert into myblob values(null,?)";

// 1.获取Connection

Connection con = JdbcUtils.getConnection();

// 2.获取PreparedStatement

PreparedStatement pst = con.prepareStatement(sql);

// 3.插入值

File file = new File("D:\\java1110\\day17-jdbc\\视频\\3.jdbc快速入门.avi");

FileInputStream fis = new FileInputStream(file);

pst.setBinaryStream(1, fis, (int) (file.length()));

int row = pst.executeUpdate();

if (row != 0) {

System.out.println("插入成功");

}

// 4.释放资源

fis.close();

pst.close();

con.close();

}

// 获取

@Test

public void get() throws SQLException, IOException {

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

// 1.获取Connection

Connection con = JdbcUtils.getConnection();

// 2.获取PreparedStatement

PreparedStatement pst = con.prepareStatement(sql);

pst.setInt(1, 1);

// 3.得到结果集

ResultSet rs = pst.executeQuery();

// 4.遍历结果集

if (rs.next()) {

// System.out.println(rs.getInt("id"));

InputStream is = rs.getBinaryStream("content");// 得到的这个输入流它的源可以理解成就是数据库中的大二进制信息

FileOutputStream fos = new FileOutputStream("d:/a.avi");

int len = -1;

byte[] b = new byte[1024 \* 100];

while ((len = is.read(b)) != -1) {

fos.write(b, 0, len);

fos.flush();

}

fos.close();

is.close();

}

// 5.关闭

rs.close();

pst.close();

con.close();

}

}