爬虫数据库描述：

运行环境sql server 2008

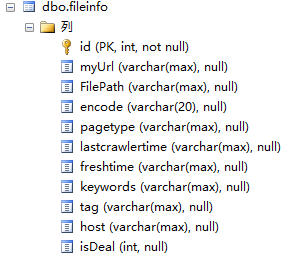
数据类型定义：

表定义：

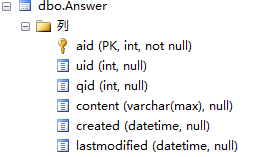
**(注意：)**

**(除了fileinfo之外其余的表都是空表，所以需要的属性可以重新定义)**

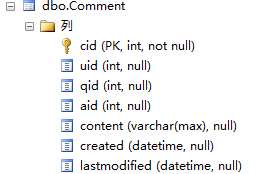
dbo.fileinfo



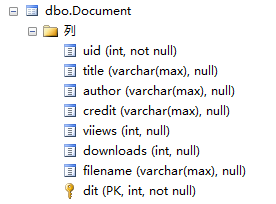
dbo.answer



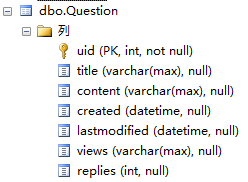
Dbo.Comment



Dbo.Document



Dbo.Question



数据库远程连接方法：

一：通过sql server客户端访问

数据库名称：10.2.28.78

身份验证：SQL Server身份验证

登录名：crawler

密码：aimashi2015

**二：通过java代码访问**

1. JDBC连接数据库
2. •创建一个以JDBC连接数据库的程序，包含7个步骤：
3. 1、加载JDBC驱动程序：
4. 在连接数据库之前，首先要加载想要连接的数据库的驱动到JVM（Java虚拟机），
5. 这通过java.lang.Class类的静态方法forName(String  className)实现。
6. 例如：
7. **try**{
8. //加载MySql的驱动类
9. Class.forName("com.mysql.jdbc.Driver") ;
10. }**catch**(ClassNotFoundException e){
11. System.out.println("找不到驱动程序类 ，加载驱动失败！");
12. e.printStackTrace() ;
13. }
14. 成功加载后，会将Driver类的实例注册到DriverManager类中。
15. 2、提供JDBC连接的URL
16. •连接URL定义了连接数据库时的协议、子协议、数据源标识。
17. •书写形式：协议：子协议：数据源标识
18. 协议：在JDBC中总是以jdbc开始
19. 子协议：是桥连接的驱动程序或是数据库管理系统名称。
20. 数据源标识：标记找到数据库来源的地址与连接端口。
21. 例如：（MySql的连接URL）
22. jdbc:mysql:
23. //localhost:3306/test?useUnicode=true&characterEncoding=gbk ;
24. useUnicode=**true**：表示使用Unicode字符集。如果characterEncoding设置为
25. gb2312或GBK，本参数必须设置为**true** 。characterEncoding=gbk：字符编码方式。
26. 3、创建数据库的连接
27. •要连接数据库，需要向java.sql.DriverManager请求并获得Connection对象，
28. 该对象就代表一个数据库的连接。
29. •使用DriverManager的getConnectin(String url , String username ,
30. String password )方法传入指定的欲连接的数据库的路径、数据库的用户名和
31. 密码来获得。
32. 例如：
33. //连接MySql数据库，用户名和密码都是root
34. String url = "jdbc:mysql://localhost:3306/test" ;
35. String username = "root" ;
36. String password = "root" ;
37. **try**{
38. Connection con =
39. DriverManager.getConnection(url , username , password ) ;
40. }**catch**(SQLException se){
41. System.out.println("数据库连接失败！");
42. se.printStackTrace() ;
43. }
44. 4、创建一个Statement
45. •要执行SQL语句，必须获得java.sql.Statement实例，Statement实例分为以下3
46. 种类型：
47. 1、执行静态SQL语句。通常通过Statement实例实现。
48. 2、执行动态SQL语句。通常通过PreparedStatement实例实现。
49. 3、执行数据库存储过程。通常通过CallableStatement实例实现。
50. 具体的实现方式：
51. Statement stmt = con.createStatement() ;
52. PreparedStatement pstmt = con.prepareStatement(sql) ;
53. CallableStatement cstmt =
54. con.prepareCall("{CALL demoSp(? , ?)}") ;
55. 5、执行SQL语句
56. Statement接口提供了三种执行SQL语句的方法：executeQuery 、executeUpdate
57. 和execute
58. 1、ResultSet executeQuery(String sqlString)：执行查询数据库的SQL语句
59. ，返回一个结果集（ResultSet）对象。
60. 2、**int** executeUpdate(String sqlString)：用于执行INSERT、UPDATE或
61. DELETE语句以及SQL DDL语句，如：CREATE TABLE和DROP TABLE等
62. 3、execute(sqlString):用于执行返回多个结果集、多个更新计数或二者组合的
63. 语句。
64. 具体实现的代码：
65. ResultSet rs = stmt.executeQuery("SELECT \* FROM ...") ;
66. **int** rows = stmt.executeUpdate("INSERT INTO ...") ;
67. **boolean** flag = stmt.execute(String sql) ;
68. 6、处理结果
69. 两种情况：
70. 1、执行更新返回的是本次操作影响到的记录数。
71. 2、执行查询返回的结果是一个ResultSet对象。
72. • ResultSet包含符合SQL语句中条件的所有行，并且它通过一套get方法提供了对这些
73. 行中数据的访问。
74. • 使用结果集（ResultSet）对象的访问方法获取数据：
75. **while**(rs.next()){
76. String name = rs.getString("name") ;
77. String pass = rs.getString(1) ; // 此方法比较高效
78. }
79. （列是从左到右编号的，并且从列1开始）
80. 7、关闭JDBC对象
81. 操作完成以后要把所有使用的JDBC对象全都关闭，以释放JDBC资源，关闭顺序和声
82. 明顺序相反：
83. 1、关闭记录集
84. 2、关闭声明
85. 3、关闭连接对象
86. **if**(rs != **null**){   // 关闭记录集
87. **try**{
88. rs.close() ;
89. }**catch**(SQLException e){
90. e.printStackTrace() ;
91. }
92. }
93. **if**(stmt != **null**){   // 关闭声明
94. **try**{
95. stmt.close() ;
96. }**catch**(SQLException e){
97. e.printStackTrace() ;
98. }
99. }
100. **if**(conn != **null**){  // 关闭连接对象
101. **try**{
102. conn.close() ;
103. }**catch**(SQLException e){
104. e.printStackTrace() ;
105. }
106. }

**这里以我们的具体情况为例，有以下代码：**

**功能是查看fileinfo里面有多少项，返回结果，实验结果完美**

**(使用前要加入sqlserver2008.jar作为外部依赖)**

**import** java.sql.\*;

**public** **class** Conn{

**private** **static** Connection *conn* = **null**;

**private** **static** Statement *st* = **null**;

**private** **static** ResultSet *rs* = **null**;

//建立数据库的连接

**public** **static** **void** main(String[] args){

System.***out***.println("here");

String url = "jdbc:sqlserver://10.2.28.78;databaseName=Crawler";

String user = "crawler";

String password = "aimashi2015";

**try** {

Class.*forName*("com.microsoft.sqlserver.jdbc.SQLServerDriver");

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

*st* = *conn*.createStatement(ResultSet.***TYPE\_SCROLL\_INSENSITIVE***, ResultSet.***CONCUR\_READ\_ONLY***);

} **catch** (ClassNotFoundException e) {

e.printStackTrace();

} **catch** (SQLException e) {

e.printStackTrace();

}

**try**{

*rs*= *st*.executeQuery("select \* from fileinfo");

}**catch**(SQLException e){

e.printStackTrace();

}

**try** {

System.***out***.println(*rs*.getRow());

*rs*.last();

System.***out***.println(*rs*.getRow());

} **catch** (SQLException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

**try** {

*conn*.close();

} **catch** (SQLException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

}

相关定义说明：

ResultSet为定义好的返回集，类似于游标机制，通过接口可以访问查询结果。

Statement为要执行的sql语句，可以通过executeQuery方法执行string格式的语句

这样就可以在任意的java代码里面访问并且使用我们的数据库了！

祝大家使用愉快