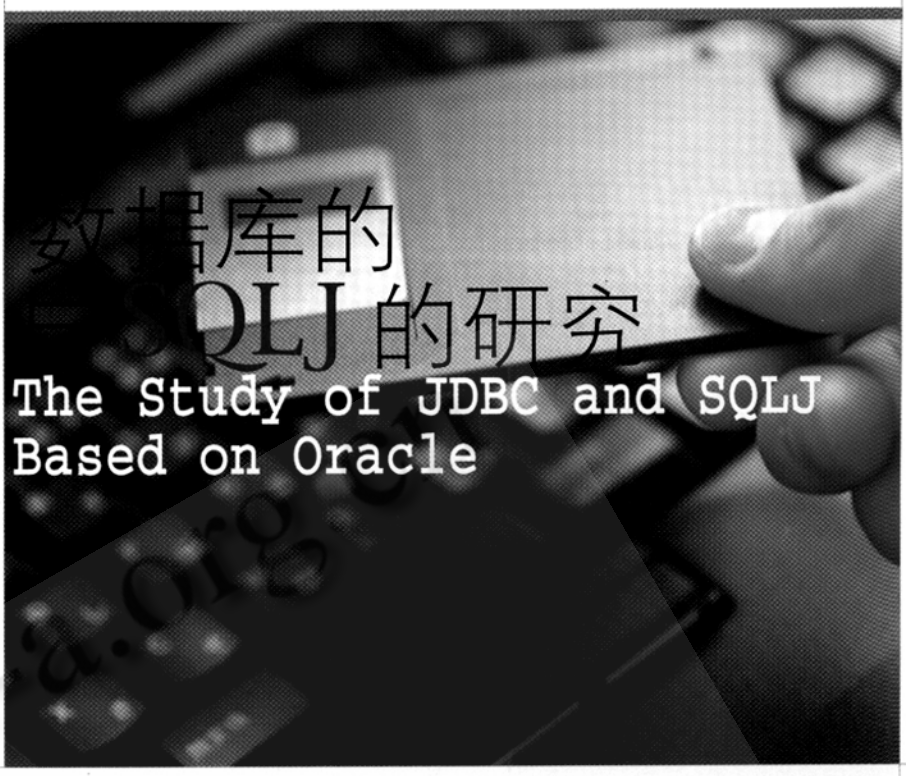
基于Oracle数据库的JDBC与SQLJ的研究

The Study of JDBC and SQLJ Based on Oracle



浦云明

（厦门集美大学信息工程学院　361021）

摘要：本文主要研究基于Oracle数据库的JAVA应用程序数据库编程接口JDBC和SQLJ，介绍了在J2SDK1.4.0环境下通过JDBC-ODBC桥实现数据操纵功能，介绍SQLJ程序的编写、翻译、编译过程，和使用Jdeveloper开发SQLJ程序步骤。

关键词：Oracle　JDBC　SQLJ

1 引言

Java语言因自动内存管理和自动垃圾收集等强大功能、可移植性、高效率而成为INTERNET应用的主要语言，Oracle 8i是Oracle的INTERNET数据库，因此Java与Oracle数据库的结合成为许多商业应用的较佳选择，这也要求Java与Oracle有简单和易以实现的编程连接技术，以方便、简单地调用SQL和PL/SQL。

Oracle提供了2种方法使JAVA数据库应用程序调用SQL和PL/SQL：JDBC与SQLJ。JDBC提供给Java一个动态嵌入式SQL接口，Java程序可以通过Java数据库连接（JDBC）方法调用来调用SQL语句；SQLJ提供了一个静态嵌入式SQL接口，是在Java编程语言中静态嵌入SQL，一个SQLJ程序实际上是一个包含静态嵌入式SQL语句的Java程序。

2 JDBC数据库连接及应用

JDBC（Java数据库连接）是JAVASOFT定义的JDBC API的技术规范，由SUN和Java技术合作伙伴开发的Java应用程序的标准数据库访问接口。JDBC由一系列API（API是支持数据库连接的Java接口、类、例外处理）和数据库厂家提供的数据库连接底层驱动程序组成。JDBC驱动程序可分为四类：

(1) JDBC-ODBC桥：通过ODBC驱动程序提供JDBC访问，适合于商业网络与三层应用，要求ODBC二进制代码在每个客户机安装。

(2) 部分Java技术的本地API驱动程序：将JDBC调用转换为客户端API对DBMS的调用，每个客户机安装一定的二进制代码。

(3) 全部基于Java技术的本地API驱动程序：将JDBC调用翻译为与DBMS无关的网络协议，网络服务器再将其翻译为DBMS协议。

(4) 全部基于Java技术的本地协议驱动程序：直接将JDBC调用转换为DBMS使用的网络协议。

下面我们使用JDBC-ODBC桥实现Java与Oracle数据库的连接，首先我们需要建立数据库，建立ODBC，通过SUN或Oracle等公司的ODBC-JDBC桥实现连接，表1是创建JDBC步骤和在J2SDK1.4.0环境下使用JDBC实现数据库操作的程序段，从学生表STUDENT中查询相应数据。

3 SQLJ数据库连接及应用

SQLJ或称嵌入SQL的Java，已成为ANSI/ISO的工业标准，SQLJ由一系列子句组成，这些子句扩展了Java程序，它得到了IBM、JAVASOFT、Oracle等主流数据库厂商的支持。SQLJ提供方便的方法来建造数据库存储过程，

表1

序号	创建JDBC步骤	在J2SDK1.4.0环境下运行的实例
1	导入JDBC API包等，Java.sql.*	import java.sql.*; import java.awt.*;

2	装入和注册JDBC驱动程序，然后建立连接对象	// URL中指定ODBC设置的DSN，名为ORATEST
		String url = "jdbc:odbc:oratest";
		String username = "system";
		String password = "manager";
		//加载驱动程序连接数据库
		try{
		Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
		Connection=DriverManager.getConnection(url,username,password);
		}

3	创建connection对象，使用SUN公司的JDBC-ODBC桥连接数据库，数据连接例外捕获处理	//捕获例外
		catch(ClassNotFoundException cnfex){
		.....
		System.exit(1);
		}
		catch(SQLException sqlex){
		.....
		System.exit(1);
		}

4	创建Statement实例，向数据库发送SQL语句。可使用动态或静态嵌入式SQL语句。	//连接成功，建立GUI
		String testcomm="select * from student";
		inputQuery = new JtextArea(testcomm,4,30);
		SubmitQuery = new Jbutton("查询");
		//button event
		submitQuery.addActionListener(
		new ActionListener(){
		public void actionPerformed(ActionEvent e)
		{
		getTable();
		}
		}
		)

5	用JDBC Statement实例执行由Statement类指定的数据库任务，并进行例外处理。	Private void getTable()
		{
		try{ //执行sql语句
		String query=inputQuery.getText();
		Statement = connection.createStatement();
		ResultSet = statement.executeQuery(query);
		DisplayResultSet(ResultSet);
		}
		catch (SQLException sqlex) {
		sqlex.printStackTrace();
		}
		}

6	用JDBC ResultSet实例存储数据库查询结果并用next()方法检索数据	Private void displayResultSet(ResultSet rs) throws SQLException
		{
		boolean moreRecords=rs.next();
		Vector columnHeads = new Vector();
		Vector rows = new Vector();
		try{
		ResultSetMetaData rsmd=rs.getMetaData();// 取字段名
		For (int i=1;i<=rsmd.getColumnCount();i++)
		ColumnHeads.addElement(rsmd.getColumnName(i));
		Do {
		Rows.addElement(getNextRow(rs,rsmd));// 取记录集
		}while (rs.next());
		}
		}
		Private Vector getNextRow(ResultSet rs, RsultSetMetaData rsmd)
		throws SQLException
		{
		Vector currentRow = new Vector();
		For (int i = 1; i <= rsmd.getColumnCount(); ++i)
		CurrentRow.addElement(rs.getString(i));
		Return currentRow;
		}

7	关闭Statement，ResultSet实例，关闭数据库连接	Public void shutDown()
		{
		try {
		connection.close();// 断开数据库连接
		}
		}

触发器、EJB、CORBA等。SQLJ比JDBC更易使用，也可将两者结合起来开，快速而有效地开发Java与Oracle数据的INTERNET应用。

Oracle公司提供了Oracle Designer，Jdeveloper，Developer200等因特网开发工具，其中Jdeveloper是用户界面较好的开发工具，可建立、调试、配置使用JAVA或SQLJ开发数据库应用。其步骤如下：创建工作空间(workspace)，创建工程(project)，把SQLJ文件加入到工程中，创建源程序文件，设置工程属性，编译程序，运行程序。

当编写一个SQLJ程序时，所编写的是一个Java程序并按一定的语法规则嵌入SQL语句，文件存储为*.sqlj，然后运行一个SQLJ翻译器将SQLJ程序转换为一个标准Java程序，一个Oracle翻译器完成如下工作：对嵌入式SQL构造进行语法检查，Java和SQL数据类型检查，模式检查。

翻译时，SQL翻译器把嵌入式SQLJ语句替换为SQLJ运行库中的方法调用以实现SQL操作。这样翻译的结果是得到一个可使用任何Java翻译器进行编译的Java源程序，Java源程序经编译后，Java执行程序就可在任何数据库上运行。

SQLJ运行环境由纯Java实现的小SQLJ运行库组成，该运行库转而调用相应数据库的JDBC驱动程序。因此SQLJ底层是由JDBC驱动程序实现的，下图显示了SQLJ翻译器如何与Java编译器和运行环境执行一个SQLJ程序。

SQLJ是在Java编程语言中静态嵌入SQL，即一个SQLJ程序是一个包含静态嵌入式SQL语句的Java程序。SQLJ存储子程序的调用方式与PL/SQL存储子程序完全相同，下面简单介绍SQLJ使用数据库连接语句、单行查询语句、PL/SQL块。

(1) SQLJ首先要连接数据库，用Oracle.sqlj.runtime.Oracle类的connect()方法，该方法需要的参数包括：JDBC驱动程序(jdbc:oracle:thin)，数据库主机(jsjx-server)，SQL*NET服务器端口(1521)，和数据库SID(ORCL)，以及用户和口令(SCOTT和TIGER)，以下是连接程序：

```
Oracle.connect("jdbc:oracle:thin:@jsjx-server:1521:ORCL"," scott"," tiger"")
```

(2) SQLJ执行语句以#sqlj开头，接着是花括号内的SQL操作，并以分号结尾。以下是带JAVA宿主变量的单行查询语句：

```
//SQLJ语句将查询结果存入JAVA宿主变量：studentname和:studentno
String studentname = null;
String studentno = null;
#sqlj { select sname,sno into :studentname,:studentno
    from student
    where sage>20
};
```

(3) SQLJ允许PL/SQL作为可执行语句：

```
#sqlj{
    declare
        studentNo number;
    begin
        studentNo := 1;
        while studentNo <= 5000 loop //2002级招生人数小于5000
            insert into student(sno) values (2002+studentNo);
            studentNo := studentNo + 1;
        end loop
    end;
};
```

一个SQLJ程序实际上是一Java程序，可以在多种环境下运行，因此SQLJ程序有很好的移植性，SQLJ可用来开发多线程应用程序，SQLJ运行时支持多个线程共享同一连接环境，但执行环境不能被共享，如果共享执行环境，那么一个线程进行的SQL操作的结果将会被另一个线程看到，如果两个线程都是可执行SQL操作，则会出现竞争或抛出运行例外，因此每一个线程须使用独立的执行环境。

4 JDBC与SQLJ的比较

SQLJ程序是静态的，所有的SQL语句必须在编译前确定。但在一个SQLJ语句内包含JDBC会使该程序成为动态的。SQLJ由一系列子句组成，这些子句扩展了Java程序。编写的SQLJ程序实际上是一个Java程序，并按一定的语法规则嵌入SQL语句，然后运行SQLJ翻译器将SQLJ程序转换为标准Java程序。

JDBC是在JAVA程序中使用动态SQL语句的技术方法。Java程序使用JDBC来查询和更新表中的数据，数据库对象信息存放在字符串中在运行时确定。然而许多应用程序使用的SQL语句是确定的，在此情况下，SQLJ就可用来在Java程序中嵌入静态SQL语句。在静态SQL中，Java程序中所有的SQL语句在编译时确定，因此SQLJ程序在运行时比动态SQL具有更快的执行速度。

另外，SQLJ程序比JDBC程序所需代码少，SQLJ易于调试：SQLJ可在编译时对数据库连接代码进行语法和语义检查；SQLJ对查询结果和其他返回值提供强类型检查，而JDBC在编译时对传入和传出SQL的值不做检查。

参考文献

1 Nirva Morisseau-Leroy, Oracle8i SQLJ Programming, McGraw-Hill 2000。

2 Oracle数据库管理员基础教程，机械工业出版社，1999年3月。

3 Java2应用开发指南，电子工业出版社。