J. of Basic Sciences, Cheju Nat'l Univ.
15(2), 145~153, 2002

기초과학연구 제주대학교
15(2), 145~153, 2002

윈도우 SAS 시스템에서의 데이터베이스 연동

강 형 창, 김 철 수
제주대학교 전산통계학과

요 약

인터넷의 확산과 네트웍의 확충으로 인해 대용량의 자료가 쉽게 얻어지고 있으며 이를 활용해 유용한 정보를 얻어내는 일련의 과정이 매우 중요시 되고 있다. 이런한 자료들은 데이터베이스 형태 저장 되어 있어 데이터베이스의 자료를 쉽게 액세스하여 분석하고 처리하는 방법이 필요하게 되었다. 윈도우 SAS 시스템은 이러한 데이터베이스를 액세스하여 필요한 분석을 처리할 수 있는 강력한 기능을 가지고 있다. 본 논문에서는 윈도우 SAS 시스템을 이용하여 데이터베이스의 액세스 및 연동 과정에 대해 다루며, 윈도우 SAS 시스템에서 데이터베이스를 액세스하기 위한 과정 중의 일부를 JDBC라는 JAVA에서 지원하는 기술을 이용하여 액세스하는 방법에 대해서 논의한다

1. 서 론

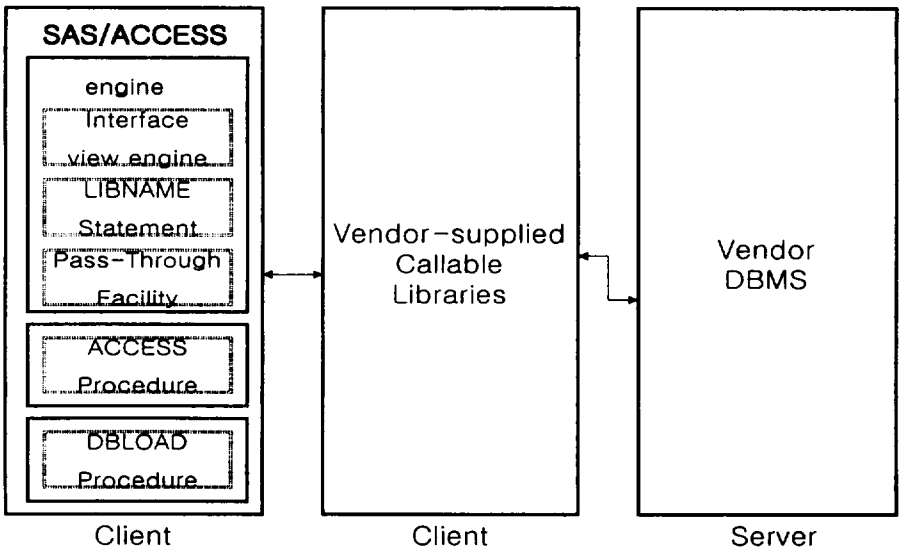
월드와이드웹이라는 개방된 환경이 새로운 컴퓨팅 환경으로 떠오르면서 데이터베이스도 변하기 시작했다. 이러한 월드와이드웹의 성격에 의해 빠르게 변하는 데이터베이스의 내용을 처리·분석하기 위해서는 데이터베이스를 직접 액세스하여 처리·분석에 필요한 데이터베이스의 테이블들을 선택하는 방법이 필요하다. 이때 대부분의 데이터베이스는 로컬 컴퓨터에 존재하는 것이 아니라, 원격의 컴퓨터에 데이터베이스가 존재하게 된다. 본 연구에서는 원격의 데이터베이스를 윈도우 SAS 시스템이 설치되어 있는 로컬 컴퓨터에서 연동하기 원한 방법에 대해서 논의한다. 이때 요구되는 사항으로 ODBC(Open DataBase Connectivity)가 필요하며 원격의 데이터베이스 연결을 위해 로컬 컴퓨터에 각 데이터베이스 연결을 위한 ODBC가 등록되어 있어야 하며, 등록된 ODBC를 이용하여 로컬 컴퓨터의 윈도우 SAS 시스템에서 원격의 데이터베이스를 직접 액세스 할 수 있다. 이때 윈도우 SAS 시스템에서는 원격의 데이터베이스 접속을 위한 SAS/ACCESS 모듈의 ODBC가 필요하다. 그리고 본 연구에서는 원격의 ORACLE 데이터베이스를 JDBC를 이용하여 접속한 다음 데이터베이스의 내용을 로컬 컴퓨터에 저장하여 사용할 수 있는 방법에 대해서 논의한다. 본 연구의 내용은 2장에서 윈도우 SAS 시스템이 데이터베이스와 연동하기 위한 ACCESS 구조와 ODBC 구조에 대하여 설명하고 실례로 MYSQL 데이터베이스와 연동하는 과정을 살펴본다. 3장에서는 JDBC를 이용하여 오라클 데이터베이스 연동을 위한 과정을 설명하며 마지막 4장에서는 결론 및 추후 연구방향에 대해 설명한다.

2. 윈도우 SAS 시스템에서 데이터베이스 연동

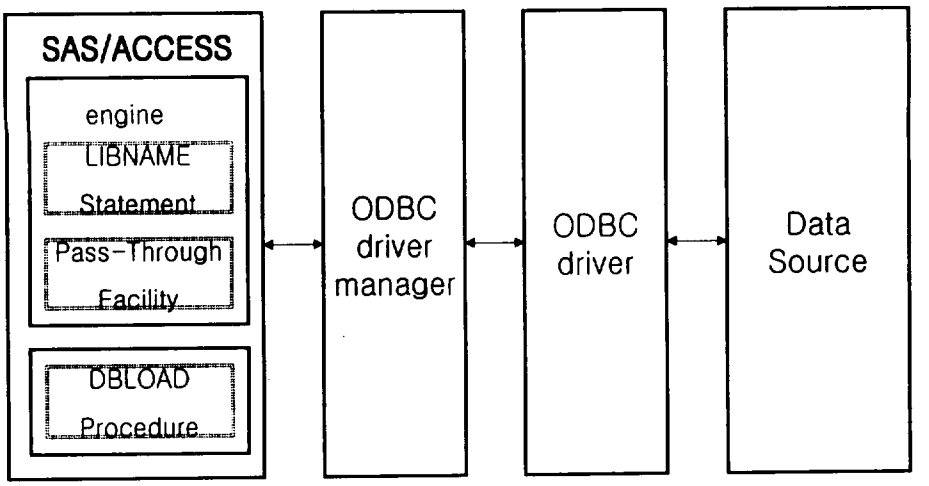
원격의 데이터베이스를 로컬 컴퓨터로 액세스하기 위해서는 첫째, ODBC(Open DataBase Connectivity)가 필요하다. ODBC(Open DataBase Connectivity)는 데이터베이스를 액세스하기 위한 표준 개방형 응용 프로그램 인터페이스로서 ODBC를 이용하면 MYSQL, ORACLE, MS-SQL 데이터베이스 외에 MS-ACCESS, DBASE, DB2, EXCEL, TEXT 등 여러 가지 종류의 데이터베이스를 액세스 할 수 있다. 그리고 데이터베이스를 액세스하기 위해서는 ODBC 소프트웨어 외에, 액세스할 각 데이터베이스마다 별도의 모듈이나 드라이버가 필요하다. 두 번째로는 등록된 ODBC를 이용하여 윈도우 SAS 시스템으로 액세스하기 위한 SAS/ACCESS ODBC 인터페이스가 필요하다. SAS/ACCESS 모듈은 MEA(Multi Engine Architecture)를 가능하게 하는 모듈로서 데이터가 저장된 소스나 포맷에 상관없이 투명하고 일관된 방식으로 데이터에 엑세스할 수 있는 기능을 제공하며 애플리케이션의 변경 없이도 다른 데이터베이스의 내용을 처리할 수 있다. 그리고, ADABAS, AS/400, DB2, DB2/6000, INFOMIX, INGRES, MYSQL, ORACLE 등의 데이터베이스와 다양한 PC 파일 이외에도 테입형태의 데이터 등 50여가지의 데이터를 엑세스할 수 있다.

데이터 엑세스 엔진은 DBMS와 파일 구조에 맞게 데이터를 읽고 쓸 수 있으며 두 가지 포맷으로 데이터를 추출하는데, 하나는 원래 데이터 소스의 로직컬한 뷰를 보여주는 것이고, 또 하나는 SAS 시스템 데이터셋 형식으로 원래 데이터를 추출하는 것이다. 사용자들은 SQL이나 다른 데이터베이스의 특정 질의어를 배우지 않고도 정보를 추출할 수 있으며 'SQL Path-Through'기능을 이용하여 SQL 문장을 RDBM과 연계하여 최적으로 실행시킬 수 있다.

SAS/ACCESS 모듈을 통한 데이터베이스 연결은 다음과 같이 나눌 수 있다. 첫 번째, ACCESS 프로시져를 이용하여 데이터베이스의 테이블들을 직접 액세스하거나 동적 SQL 문장을 특정 테이블을 선택하여 테이블의 컬럼



[그림 2.1 윈도우 SAS 시스템의 데이터베이스 연결]



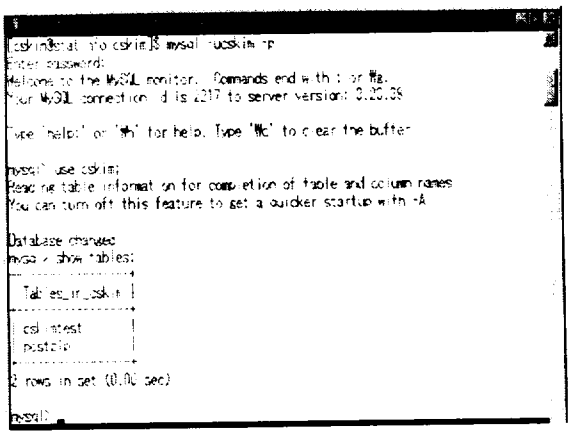
[그림 2.2 윈도우 SAS 시스템의 ODBC 인터페이스]

을 선택하여 액세스하는 방법이 있다. 두 번째, DBLOAD 프로시져를 이용하여 SAS 데이터 셋의 내용을 데이터베이스의 테이블로 생성할 수 있으며, 데이터베이스의 특정 테이블에 대하여 동적 SQL 문장을 이용하여 데이터베이스의 테이블에 새로운 컬럼을 추가하거나 레코드를 추가할 수 있다. 세 번째, SQL 프로시져와 CONNECT TO 문장을 이용하여 데이터베이스의 테이블을 액세스할 수 있다.

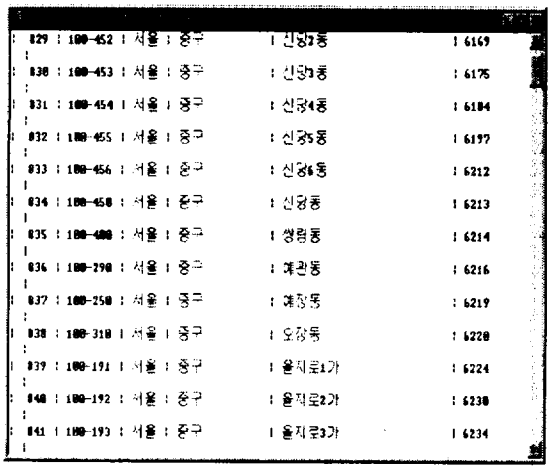
SAS/ACCESS 모듈 중에서 SAS/ACCESS ODBC 인터페이스는 ODBC에 등록된 각 데이터베이스들을 SAS 데이터 셋과 같이 처리할 수 있게 해준다. SAS/ACCESS ODBC 인터페이스는 클라이언트 인터페이스, ODBC 드라이버 관리자, 그리고 ODBC 드라이버 세 가지로 구성된다. 이때 클라이언트 인터페이스는 ACCESS 인터페이스 ODBC에서 제공된다. ODBC 드라이버 관리자는 클라이언트 인터페이스와 ODBC 드라이버를 상호 연결한다. ODBC 드라이버는 외부 데이터 또는 특정 데이터베이스를 다룰 수 있도록 연결시켜준다.

2.1 윈도우 SAS 시스템에서 MYSQL 데이터베이스 연동

다음 그림은 실제 LINUX용 MYSQL 데이터베이스의 테이블에 내용이다. 데이터베이스의 내용을 보면 사용자 cskim의 데이터베이스로 cskimtest와 postzip이라는 테이블이 있음을 볼 수 있다.



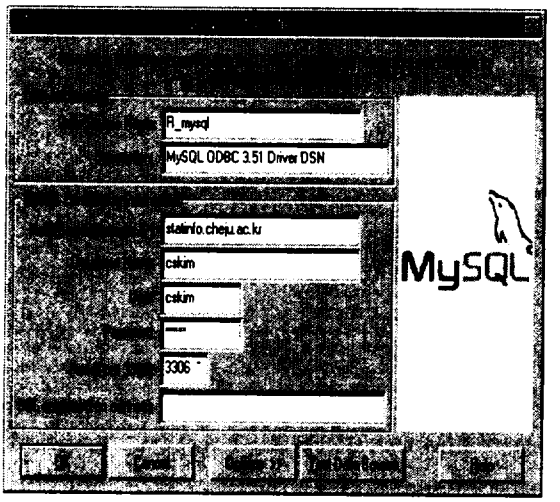
[그림 2.3 MYSQL 데이터베이스의 사용자 cskim의 테이블]



[그림 2.4 MYSQL 데이터베이스의 사용자 cskim의 postzip 테이블의 내용]

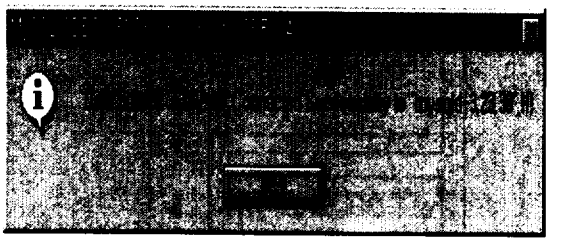
실제 윈도우 SAS 시스템을 이용하여 위의 MYSQL 데이터베이스의 내용을 액세스하는 과정을 살펴보자. 윈도우 SAS 시스템에서 원격의 MYSQL 데이터베이스를 연동하기 위해서는 다음과 같은 절차가 필요한다.

먼저 MYSQL 데이터베이스를 로컬 컴퓨터에서 사용할 수 있게 하기 위해서 ODBC에 MYSQL ODBC를 등록한다.

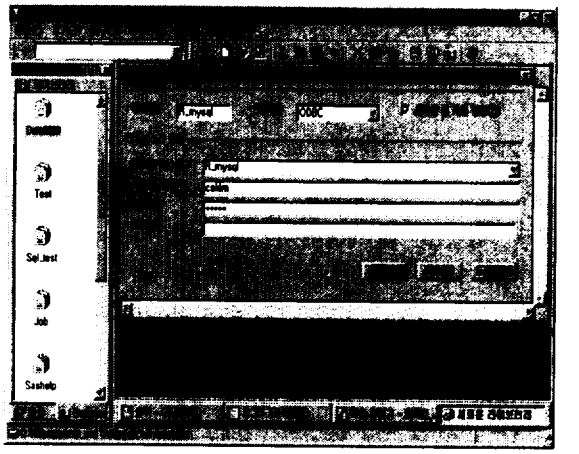


[그림 2.5 MYSQL ODBC 등록]

설정된 MYSQL ODBC가 제대로 수행되는지 Test Data Source를 클릭해보면 연결 성공여부를 알 수 있다.



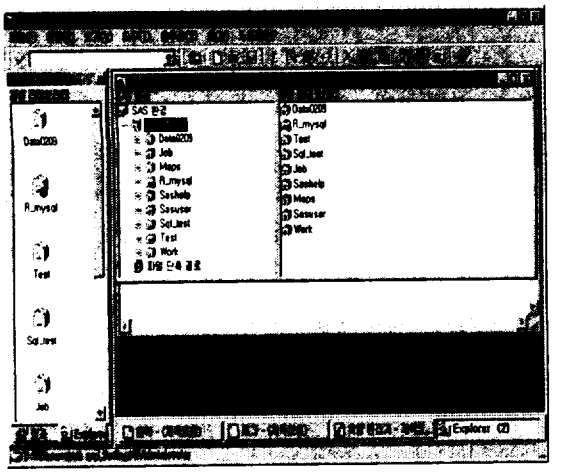
[그림 2.6 MYSQL ODBC 등록 확인]



[그림 2.7 로컬 윈도우 SAS 시스템에서 원격 MYSQL 데이터베이스 연결]

MYSQL ODBC가 성공적으로 등록되었다면 윈도우 SAS 시스템에서 사용하기 위해서 다음과 같은 과정을 거쳐야 한다. 윈도우 SAS 시스템에서 새로운 라이브러리를 선택하고 엔진을 ODBC로 선택한다.

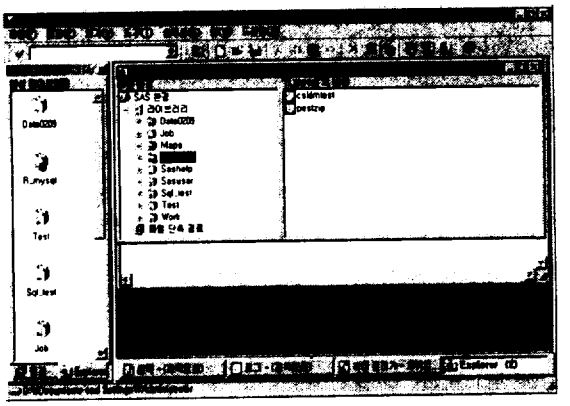
새로운 라이브러리 화면에서 라이브러리 이름은 데이터베이스를 구분할 수 있는 적당한 이름을 넣는다. 데이터 소스는 앞서 등록한 ODBC의 사용자 DSN(R_mysql)을 선택한다. 마지막으로 MYSQL 데이터베이스 사용자 ID와 암호를 입력한다.



[그림 2.8 윈도우 SAS 시스템에서 MYSQL 데이터 베이스 연결]

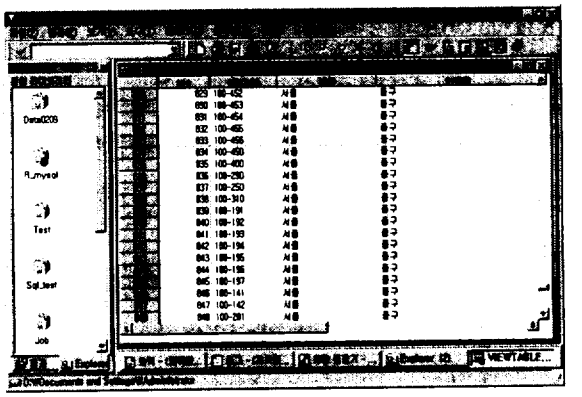
이 과정이 끝나면 [그림 2.8]과 같이 MYSQL 데이터베이스와 윈도우 SAS 시스템이 연결되어 있음을 볼 수 있다.

다음 [그림2.9]에서 왼쪽 SAS 환경의 내용은 윈도우 SAS 시스템에 등록되어 있는 라이브러리들을 보여주고 있다. 오른쪽의 R_mysql 라이브러리의 내용을 보면 cskimtest라는 데이터 셋과 psotzip이라는 데이터 셋이 생성되어 있음을 볼 수 있는데 이 데이터 셋들은 실제 MYSQL 데이터베이스에 저장되어 있는 사용자 cskim의 실제 테이블들 이다.



[그림 2.9 MYSQL 데이터베이스의 테이블]

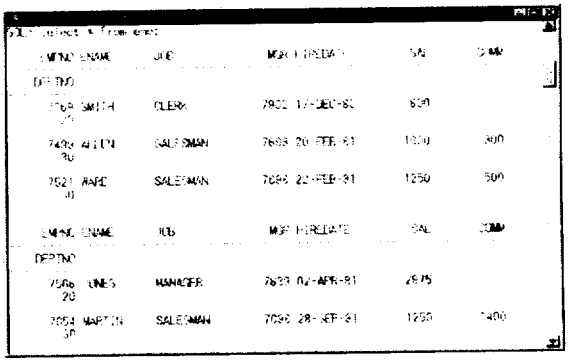
다음 [그림2.10]에서 윈도우 SAS 시스템으로 연결한 실제 MYSQL 데이터베이스의 사용자 cskim의 테이블에서 postzip이라는 테이블의 내용을 볼 수 있다.



[그림 2.10 윈도우 SAS 시스템에서의 postzip 테이블 내용]

3. JDBC를 이용한 ORACLE 데이터베이스 연동

다음 [그림3.1]은 telnet을 이용하여 ORACLE 데이터베이스에 접속한 후 emp라는 테이블에 대한 내용을 SQL 문장을 이용하여 출력한 내용이다.



[그림 3.1 telnet을 이용한 ORACLE 데이터베이스 내용]

위와 같은 내용을 로컬 컴퓨터의 윈도우 SAS 시스템에서 직접액세스 하기 위해서는 반드시 SQL*NET 클라이언트가 설치되어 있어야 하며 SQL*NET 클라이언트는 원격의 오라

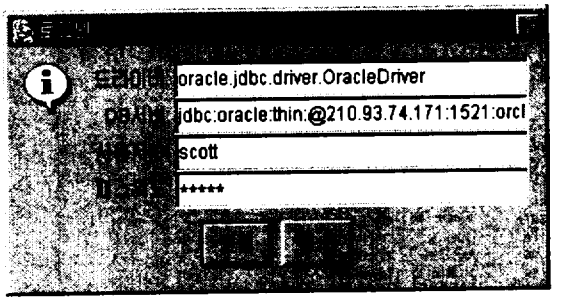
클 데이터베이스 서버를 로컬에서 직접 접속하기 위한 모듈이다. 이때 원격의 오라클 데이터베이스 서버에는 SQL*NET listner(TNS listener)가 기동되어 있어야한다.

본 논문에서는 ORACLE 데이터베이스를 접속하기 위해서 SQL*NET 클라이언트 없이 ORACLE 데이터베이스를 접속하는 방법으로 JDBC를 이용하여 연결 API(Application Programming Interface)를 사용하였다.

다음 코드는 오라클을 연결시키는 부분으로서 이 부분은 자바와 오라클에서 제공하는 것을 그대로 사용한다. 자바의 특성상 해당 데이타베이스의 연결에 알맞는 인수를 집어넣으면 해당 데이타베이스와 연동하게 된다. 여기서는 Oracle Thin 드라이버를 사용했으며, 오라클 ODBC의 패스를 잡아주었다. 그리고, 각 데이타베이스와 자바의 연결부분은 클래스화 되어있기 때문에 클래스 패스를 잡아주고 사용하였다.

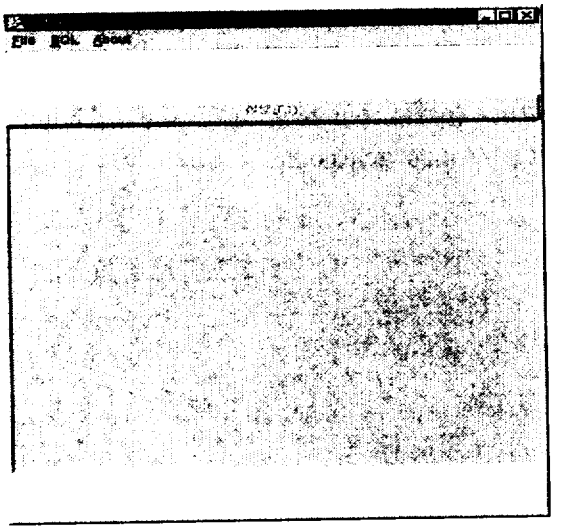
```
String user = "";
String password = "";
String url = "jdbc:oracle:thin:@210.93.74.171:1521:orcl"; //Default URL
String driver = "oracle.jdbc.driver.OracleDriver";  //Default 드라이버 (오라클 thin 드라이버)
if(JOptionPane.showOptionDialog(f, p,  "로그인", JOptionPane.DEFAULT_OPTION,
JOptionPane.INFORMATION_MESSAGE, null, option, option[0]) == 0)
{
        driver = textfieldDriver.getText();
        url = textfieldServer.getText();
        user = textfieldUserName.getText();
        password = new String(textfieldUserPwd.getPassword());
}

try
{
        Class.forName(driver);                  // 드라이버를 로드한다.
        connection = DriverManager.getConnection(url, user, password);
        statement = connection.createStatement();
}
catch (ClassNotFoundException cnfe)
{
        System.err.println(cnfe);  //드라이버를 찾을 수 없음
}
catch (SQLException sqle)
{
        System.err.println(sqle);   // 데이터베이스 연결 오류
}
```

다음 코드는 SQL문을 실행했을 때 처리하는 부분으로 반환값이 있는 결과값을 가져 오게 된다. 결과값은 AbstractTableModel 클래스를 사용하여 모든 값을 테이블화 하여 가져오게 되며 테이블화 할 수 없는 값은 오라클 자체 오류를 반환하게 되지만, 해당 SQL문은 실행이 된다.



[그림 3.3 JDBC를 이용한 ORACLE 데이터베이스 로그인 화면]

```
public void executeSQL()
{
        String query = command.getText();
        if (query == null)
                return;
        try
        {
                model.setResultSet(statement.executeQuery(query));
                status.setText("Resultset has " + model.getRowCount() + "rows.");
        }
        catch (SQLException sqle)
        {
                status.setText(sqle.getMessage());
        }
}
```



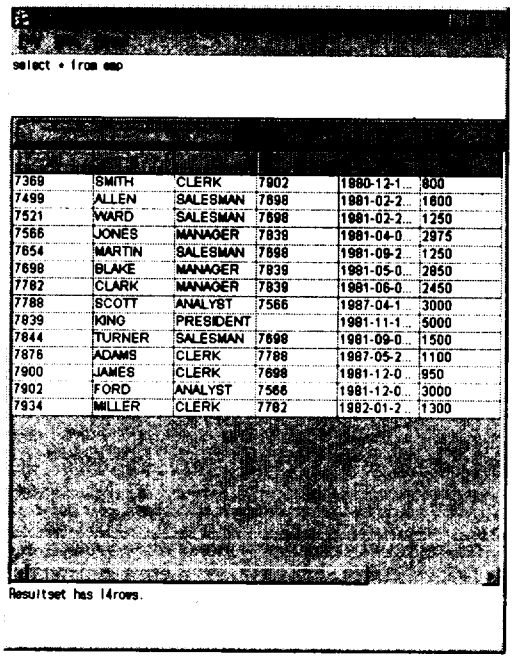
[그림 3.2 JDBC를 이용한 ORACLE 데이터베이스 연결 화면]

다음은 JDBC를 이용하여 작성한 프로그램을 이용하여 실행한 결과들이다.

[그림 3.2]와 [그림 3.3]은 ORACLE 데이터베이스 연결을 위한 화면으로 JAVA 프로그램을 수행하면 [그림 3.2]와 같은 연결 화면이 나타나고 [그림 3.3]과 같이 사용자 ID 및 패스워드를 입력하면 ORACLE 데이터베이스와 연결된다.

다음 그림은 연결된 ORACLE 데이터베이스로부터 테이블의 내용을 쿼리한 결과이다.

[그림 3.4]의 결과를 보면 telnet을 이용하여 ORACLE 데이터베이스의 emp 테이블 쿼리한 결과 [그림 3.1]과 동일함을 알 수 있다.



[그림 3.4 JDBC를 이용한 ORACLE 데이터베이스 실행 화면]

4. 결론 및 고찰

윈도우 SAS 시스템에서 데이터베이스를 연동하기 위한 방법에 대하여 앞에서 살펴보았다.

원격의 데이터베이스를 로컬 컴퓨터에서 액세스하기 위해서는 먼저 ODBC를 등록하고 등록된 ODBC를 이용하여 윈도우 SAS 시스템에서 데이터베이스를 액세스하고 원격의 데이터베이스를 로컬 컴퓨터에 저장되어 있는 형태와 유사하게 사용할 수 있다. MYSQL 데이터베이스의 경우는 ODBC를 등록한 후 윈도우 SAS 시스템에서 새로운 라이브러리를 이용하여 라이브러리 등록후 로컬 컴퓨터에 저장된 형태처럼 사용할 수 있으나, ORACLE 데이터베이스의 경우는 ODBC를 등록하고, SQL*NET 클라이언트를 설치하여만 ORACLE 데이터베이스를 액세스 할 수 있다. 이에 본 연구에서는 JDBC를 이용하여 SQL*NET 클라이언트 없이 ORACLE 데이터베이스를 접속할 수 있는 방법에 대해서 논의하였다. 차후 연구방향으로는 JDBC를 이용하여 ORACLE 데이터베이스 외에 MYSQL, MSSQL 등의 데이터베이스를 액세스하여 로컬 컴퓨터에 저장하여 윈도우 SAS 시스템에서 액세스하는 연구가 진행될 예정이며 더 나아가 SAS/ACCESS ODBC 인터페이스 없이 윈도우 SAS 시스템에서 액세스할 수 있는 방법에 대해 연구될 것이다.

참고문헌

http://ftp.sas.com/techsup/down/technote
http://otn.oracle.co.kr

Windows SAS system for Database connectivity

Hyung Chang Kang, Chul Soo Kim
Department of Computer Science and Statistics, Cheju National University

This paper explains how windows SAS system works so we can get data sets from database of DBMS system of remote DB servers using MYSQL or ORACLE.
To access DBMS data sets from remote servers we can use ODBC and JDBC.