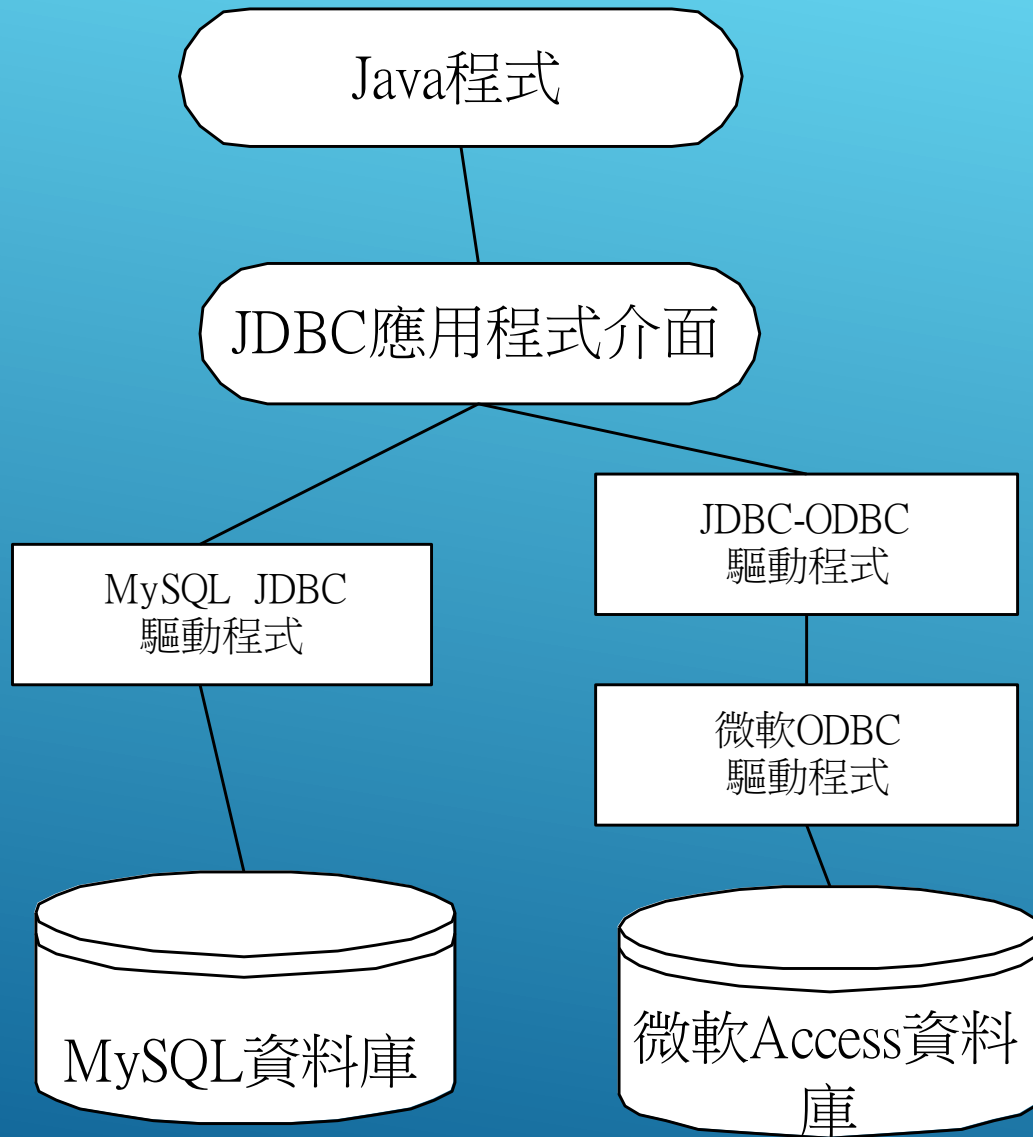


JDBC



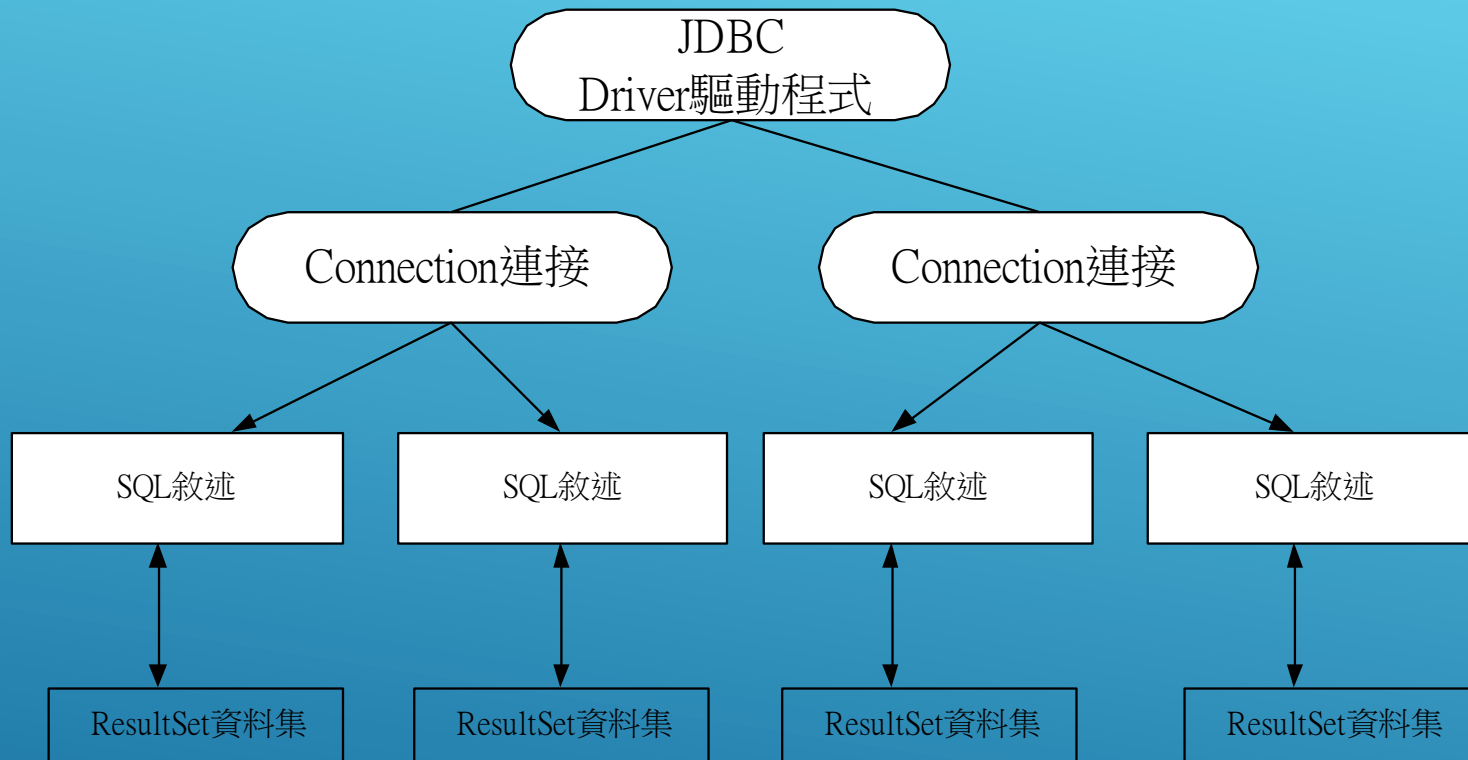
- ▶ 我們Java的程式可以使用JDBC介面來存取資料庫。在這邊，我們主要講解Java透過JDBC來存取MySQL資料庫。我們的Java程式透過JDBC介面來存取資料庫。資料庫包含了Oracle資料庫、Sybase資料庫、MSQL資料庫、MySQL資料庫和Access資料庫。我們在這裏主要是要講解Java程式如何透過JDBC介面去存取MySQL資料庫或Access資料庫。

1-1 簡介



- ▶ 這是Java存取資料的過程，JDBC驅動程式建立和資料庫的連接，然後建立SQL敘述並且交給資料庫，資料庫在作完運算後會得到資料集ResultSet，並且傳回給Java應用程式。

1-2存取資料的過程



- ▶ 資料定義語言DDL(data definition language)就是定義資料庫的schema(輪廓)，通常由資料庫的管理者或設計師所使用。通常資料庫管理系統都會有資料定義語言的編譯器來處理資料定義的敘述，而且將schema的描述儲存在資料庫管理系統。

1-3SQL語言

1-3-1 建立、移除、與選擇資料庫

- ▶ 我們使用create database 資料庫名稱來建立一個新的資料庫

```
mysql> create database tutorb;
```

- ▶ 語法：create database 資料庫名稱
- ▶ 在MySQL中使用create敘述來建立資料庫

```
mysql> use tutorb;  
Reading table information for completion of table and column names  
You can turn off this feature to get a quicker startup with -A  
  
Database changed
```

我們使用USE敘述作為連線到SERVER的
資料庫

語法：USE DATABASE 資料庫名稱

- ▶ 我們使用drop來移除資料庫
- ▶ 語法：drop database 資料庫名稱
- ▶ 使用drop敘述來建立資料表
- ▶ 我們使用show來顯示資料庫或資料表，show databases就可以顯示所有資料庫。

```
mysql> drop database tutorb;
```

```
mysql> show databases;  
+-----+  
| Database |  
+-----+  
| goddess  |  
| mysql    |  
| test     |  
| tutorb   |  
+-----+  
4 rows in set (0.01 sec)
```

1-3-2資料的型態

- ▶ 由這個表格來看，姓名是字串、電話是字串(或數值都可)、住址是字串、年齡是數值、性別是字串或布林、編號是數值(且自動累加)。
- ▶ 因此當我們要建立表格時就要設定欄位的資料型態。MySQL資料庫資料的形態有數值資料、字串資料和日期時間資料。MySQL認識數種資料型態。

姓名	電話	住址	年齡	性別	編號
王大名	23362152	中正區	28	男	1
陳小單	27522222	大安區	27	女	2
黃品德	87353686	信義區	28	男	3
林可愛	21123333	大同區	22	女	4

- ▶ 我們可以建立、索引與修改、刪除資料表。
- ▶ (1)建立資料表並索引
- ▶ 我們使用create來建立資料表
- ▶ 語法：create table 資料表名稱
- ▶ (欄位名稱 資料型態(資料大小),
- ▶ 欄位名稱 資料型態(資料大小),
- ▶ 欄位名稱 資料型態(資料大小),
- ▶ 欄位名稱 資料型態(資料大小),
- ▶
- ▶ primary key(欄位名稱));

1-3-3建立、索引與修改、刪除資料表

- ▶mysql>use tutorb;
 - ▶mysql>create table student
 - ▶->(number int not null,
 - ▶->name char(20) not null,
 - ▶->grade int not null,
 - ▶->primary key(number))
- TYPE=MyISAM;

我們使用CREATE來建立資料表，並用PRIMARY KEY(欄位)來建立資料表的索引，資料表的類型TYPE為MYISAM。

```
mysql> describe student;
```

Field	Type	Null	Key	Default	Extra
number	int(11)		PRI	0	
name	char(20)				
grade	int(11)			0	

3 rows in set (0.00 sec)

在這裡NUMBER為數值型態，NAME為字串型態，GRADE為數值型態。欄位有三個，分別是NUMBER，NAME，及GRADE，而PRIMARY KEY為NUMBER

- ▶ 資料處理語言DML(data manipulation language)。當資料庫的schema已經被編譯，使用者可以使用資料處理語言DML來操作資料庫。一般的資料處理語言包括抽取資料、插入資料、刪除資料和修改資料這幾種。
- ▶ 我們經常用到選取查詢(select)敘述、插入(insert)敘述、更新敘述(update)、刪除敘述(delete)，這些都是SQL的資料操作。

1-4SQL的資料處理語言

- ▶ select 欄位名稱串列
- ▶ from 資料表串列
- ▶ where 條件
- ▶ group by 群組條件
- ▶ order by 排序條件
- ▶ having by 篩選經過group by 群組之後的資料。
- ▶ 使用select來選取total_tutor資料表中的資料編號為3號的資料。
- ▶ select * from total_tutor where number=3

1-4-1 選取查詢SELECT

查詢結果 select * from total_tutor where number=3

Number	name	sex	telephone	experience	address	depar	grade	trans	bestc	tcourse	jschedule	location	salary	email
3	吳佳諺	男	87323435	8	苗栗市	中國文學系	大學畢業	機車	"數學"	"程式設計"	都可	台北市	400	chaiyen@cml.hinet.net

- ▶ 選取total_tutor資料表的number和name欄位，我們使用total_tutor.Number和total_tutor.name這兩個欄位，並且查出編號number=3的資料。
- ▶ `select total_tutor.Number,total_tutor.name from total_tutor where number=3`

Number	name
3	吳佳謬

- ▶ 一個SQL的查詢語言可以由六個子句所組成，只有select和from是必須要的
- ▶ select <欄位或函數串列>
- ▶ from <表格串列>
- ▶ where<條件>
- ▶ group by <群組欄位>
- ▶ having <群組條件>
- ▶ order by <欄位>
- ▶ 我們使用tutor.number as n來設定編號的代號為n，並且使用having限制條件時使用運算式，運算式使用資料表的代號n來作運算。
- ▶ Select tutor.number as n,tutor.name
- ▶ From tutor
- ▶ Where tutor.number <256 and tutor.number >250
- ▶ Group by experience
- ▶ Having n >250 and n<256
- ▶ Order by salary

1-4.2SQL的查詢語言

n	name
255	周志浩
253	李承穎
252	黃天賜
251	許憶雯

這是編號250到256的資料，並且使用薪水作為排序的根據。

- ▶ 我們使用insert指令就可以把資料輸入資料庫
- ▶ 語法：
- ▶ 我們將資料 “陳小胖” ,800,“台北市北投區” 按照欄位(name,salary,location)依序插入到tutor資料表中。
- ▶ insert into tutor
- ▶ (name,salary,location)
- ▶ values
- ▶ (“陳小胖” ,800,“台北市北投區”);

1-4-3INSERT新增資料

- ▶ 語法：
- ▶ insert into 資料表
- ▶ values
- ▶ (欄位1的值,欄位2的值,欄位3的值...)

- ▶ 我們使用insert指令將資料(1,'吳佳諺',98)插入student資料表中。
- ▶ INSERT INTO student VALUES (1,'吳佳諺',98);


- ▶ 我們使用delete子句就可以把資料列給刪除
- ▶ 語法：
- ▶ 這樣就可以把tutor資料庫中編號為321到323的資料給去刪除。
- ▶ delete from tutor
- ▶ where number<=323 and number >=321
- ▶ 我們如果使用delete指令而沒有限制條件，則會將所有的tutor資料刪除。
- ▶ delete from tutor

1-4-4DELETE刪除資料

- ▶ 如果我們要把資料列給更新則要用 update 子句
- ▶ 語法：
- ▶ update 資料表
- ▶ set 欄位 = 更新的資料
- ▶ where 條件

1-4-5UPDATE

Several white diagonal lines of varying lengths and thicknesses are positioned in the bottom right corner of the slide, creating a modern, abstract graphic element.

- ▶ 我們使用update total_tutor來更新資料編號為3的資料，並且將其depar欄位設為“ 資訊工程”。
 - ▶ update total_tutor
 - ▶ set depar="資訊工程"
 - ▶ where Number=3
- 
- A series of white diagonal lines of varying lengths and thicknesses, located in the bottom right corner of the slide.

- ▶ 這是還未update的資料，其depar欄位為中國文學系。

Number	name	sex	telephone	experience	address	depar	grade	trans	bestc	tcourse	jschedule	location	salary	email
3	吳佳謬	男	87323435	8	苗栗市	中國文學系	大學畢業	機車	"數學"	"程式設計"	都可	台北市	400	chaiyen@cm1.hinet.net

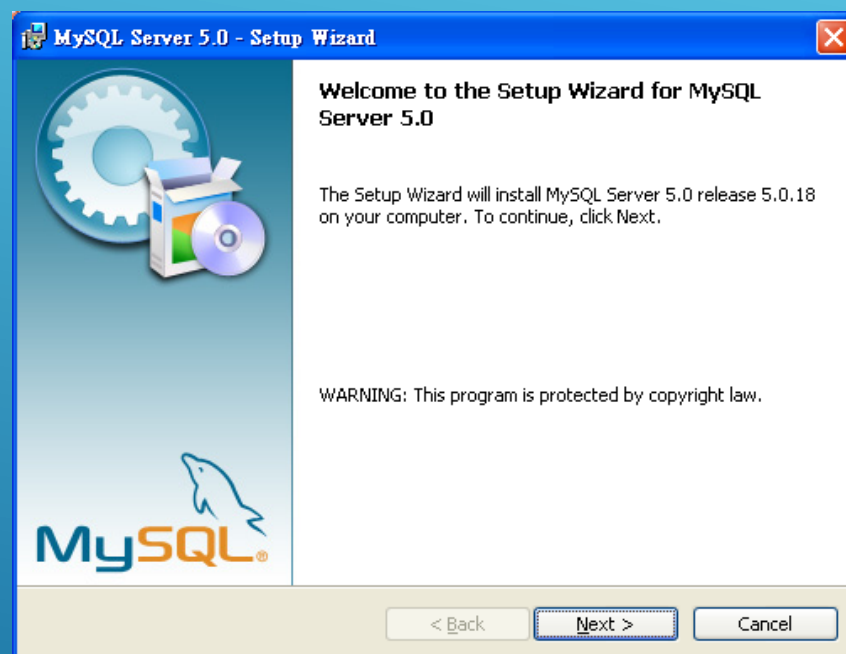
- ▶ 這是已經被更新的欄位資料，depar欄位為資訊工程。

- ▶ 我們要用Java來存取MySQL資料庫，我們必需使用JDBC介面來存取MySQL資料庫。我們可以從www.mysql.com/products/connector/j/來下載JDBC的介面mysql-connector-java-3.1.12。
- ▶ 使用mysql-connector-java-3.1.12來連接MySQL5資料庫，使用繁體中文字會產生亂碼。我們使用JDBC來連接Access資料庫則可以使用中文。我們可以使用多種JDBC來測試連接MySQL4或MySQL3的版本，來觀看中文支援的程度。我們可以到www.mysql.com/products/connector/j/來找MySQL資料庫相對應的连接介面。
- ▶ 我們然後將mysql-connector-java-3.1.12-bin.jar驅動程式套件放到C:\Program Files\Java\jdk1.***\jre\lib\ext目錄下，也就是我們Java安裝的jre\lib\ext目錄下。這樣JSP或Servlet就可以透過JDBC來連接我們資料庫了。我們也可以使用classpath來指定該套件的路徑。

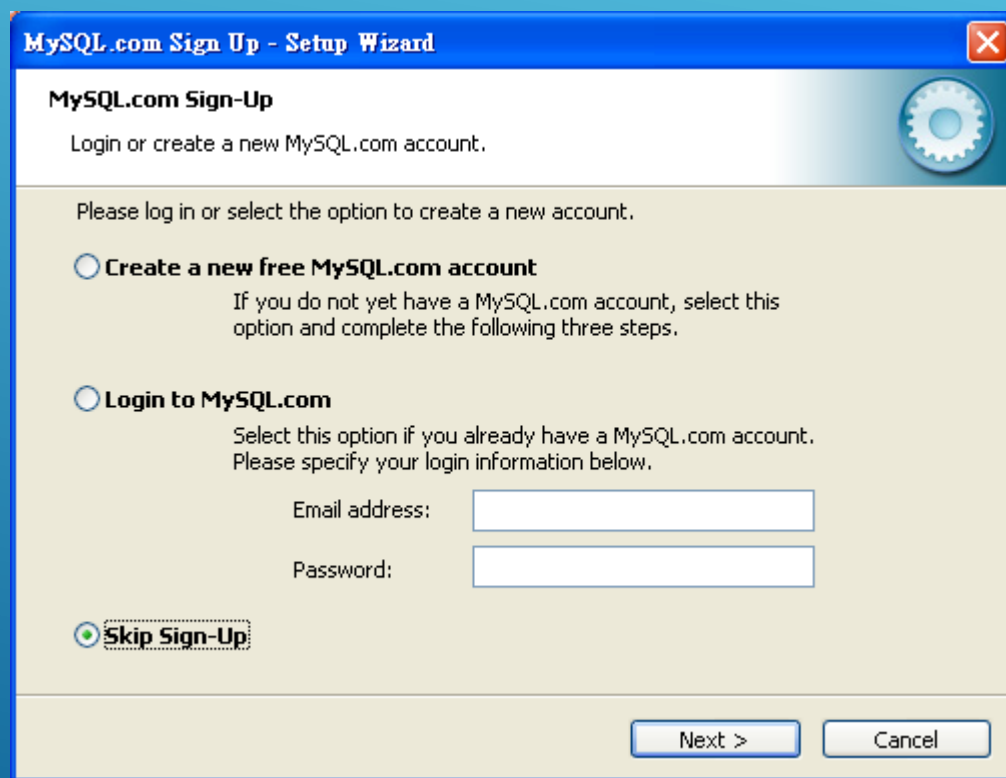
1-5MYSQL CONNECTOR/J 連接介面

1-6 MYSQL資料庫

- ▶ 我們安裝MySQL資料庫軟體，可以從光碟安裝。我們選取下一步Next。



我們可以不用到MYSQL公司的網站去，
我們選取跳過登錄SKIP SIGN-UP。我們
選取NEXT下一步。



The image shows a Windows-style dialog box titled "MySQL.com Sign Up - Setup Wizard". It has a blue title bar with a close button (X) in the top right corner. The main content area is light beige and contains the following elements:

- MySQL.com Sign-Up**: A sub-header with a gear icon to its right.
- Login or create a new MySQL.com account.
- Please log in or select the option to create a new account.
- ☐ **Create a new free MySQL.com account**

If you do not yet have a MySQL.com account, select this option and complete the following three steps.
- ☐ **Login to MySQL.com**

Select this option if you already have a MySQL.com account. Please specify your login information below.

Email address:

Password:
- ☒ **Skip Sign-Up**

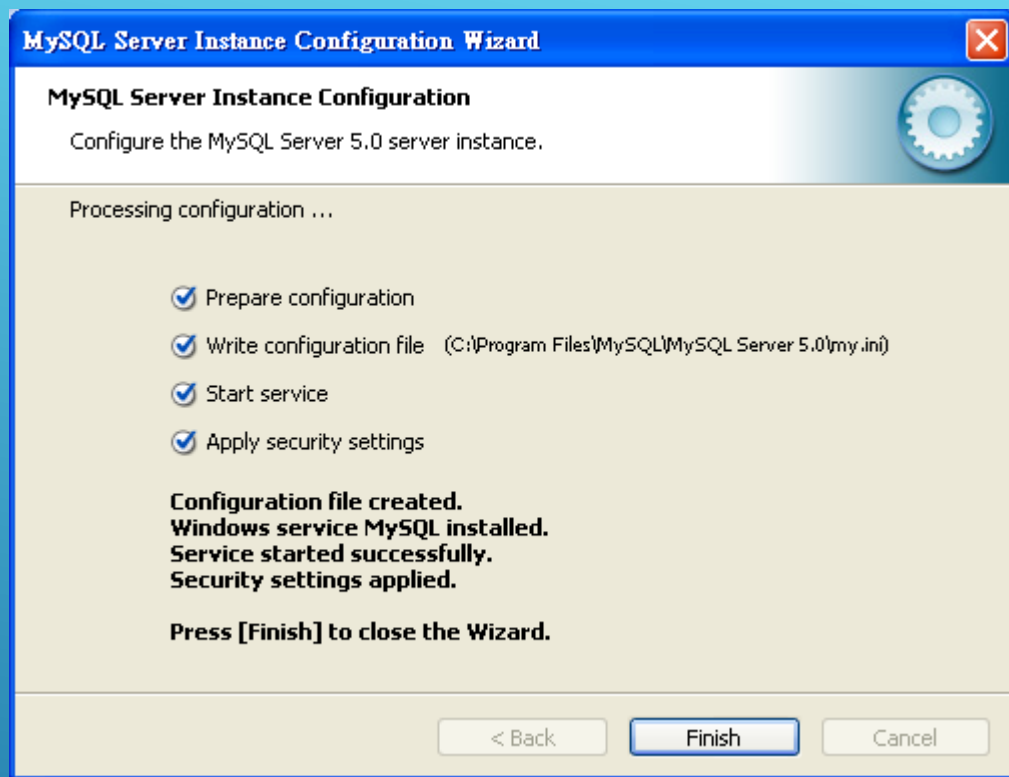
At the bottom right, there are two buttons: "Next >" and "Cancel".



我們選取視窗模式和CONSOLE
模式來操作MYSQL資料庫。我
們選取NEXT下一步。



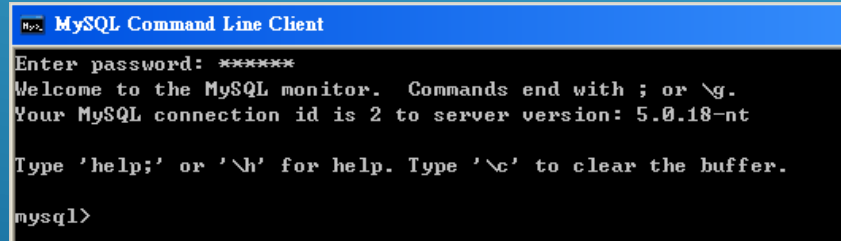
我們設定超級使用者ROOT的密碼。我們使用超級使用者ROOT的身份來操作MYSQL資料庫，我們按下一步NEXT。我們在這裏設定密碼為322739。



我們按下FINISH完成。

我們選取開始→所有程式→MYSQL→MYSQL
SERVER5.0→MYSQL COMMAND LINE CLIENT，命令模式。

- ▶ 我們輸入超級使用者root的密碼就可以登錄MySQL資料庫了。



```
MySQL Command Line Client
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2 to server version: 5.0.18-nt

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>
```

1-6-1 設定資料庫

- ▶ 我們現在要在MySQL資料庫中建立My_Books資料庫，並且在該資料庫中建立authors作者資料表和publisher出版社資料表，然後再使用insert指令將資料插入資料庫中的資料表。

- ▶ 這是My_Books.sql的檔案。

```
1 CREATE DATABASE IF NOT EXISTS My_Books;
2
3 USE My_Books;
4
5 DROP TABLE IF EXISTS authors;
6 DROP TABLE IF EXISTS publisher;
7
8 CREATE TABLE authors (
9     authorID INT NOT NULL AUTO_INCREMENT,
10    Name varchar (30) NOT NULL,
11    PRIMARY KEY (authorID)
12 ) TYPE=INNODB;
13
14 CREATE TABLE publisher (
15     Id INT NOT NULL AUTO_INCREMENT,
16    publisherName varchar (20) NOT NULL,
17    PRIMARY KEY (Id)
18 ) TYPE=INNODB;
19
20 insert into publisher (publisherName) values ('成大');
21 insert into publisher (publisherName) values ('Linux Company');
22 insert into authors (Name) values ('Justin wu');
23 insert into authors (Name) values ('Chaiyen wu');
24 insert into authors (Name) values ('Miss Chen');
```

```
C:\>mysql -u root -p < My_Books.sql  
Enter password: *****
```

我們將MY_BOOKS.SQL的SQL指令增加到MYSQL資料庫中。-U是指定使用者為超級使用者ROOT，-P是指定其密碼。<為輸入的符號。

- ▶ 在執行完輸入My_Books.sql後，我們可以看到在MySQL資料庫中，已經新增的my_books資料庫。

```
C:\>mysql -u root -p
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 17 to server version: 5.0.18-nt

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| my_books |
| mysql |
| test |
+-----+
4 rows in set (0.00 sec)
```

```
mysql> use my_books;  
Database changed  
mysql> show tables;  
+-----+  
| Tables_in_my_books |  
+-----+  
| authors             |  
| publisher           |  
+-----+  
2 rows in set (0.00 sec)
```

我們顯示了MY_BOOKS資料庫中，有AUTHORS資料表和PUBLISHER資料表。

- ▶ JDBC介面和類別是用來發展JAVA連接資料庫。下面是JAVA連接資料庫的步驟。
- ▶ (1).載入資料庫，我們使用
`Class.forName("JDBCDriverClass");`
- ▶ 驅動程式是實體類別，它實作了
`java.sql.Driver` 介面。下列是Access、MySQL和Oracle的驅動程式。

1-7操作資料庫使用JDBC

資料庫	驅動程式類別	驅動程式所在位置
Access 資料庫	<u>sun.jdbc.odbc.JdbcOdbcDriver</u>	已經內建在 JDK 中
MySQL 資料庫	<u>com.mysql.jdbc.Driver</u>	mysql-connector-java-3.1.12-bin.jar
Oracle	<u>oracle.jdbc.driver.OracleDriver</u>	Classes12.jar

- ▶ Access資料庫的JDBC-ODBC驅動程式已經內建在JDK中。MySQL資料庫的JDBC驅動程式則在mysql-connector-java-3.1.12-bin.jar中。Oracle資料庫的JDBC驅動程式則在classes12.jar中。我們可以到MySQL的網站和Oracle的網站去下載該JDBC驅動程式。
- ▶ 我們要將mysql-connector-java-3.1.12-bin.jar和classes12.jar的類別路徑classpath加入到DOS指令中，這樣才能使用MySQL資料庫和Oracle資料庫。預設是將mysql-connector-java-3.1.12-bin.jar套件和classes12.jar套件放入到c:\目錄下。

CLASSPATH=%CLASSPATH%;C:\ MYSQL-CONNECTOR-JAVA-3.1.12-BIN.JAR;C:\ CLASSES12.JAR

- ▶ `static final String JDBC_DRIVER = "com.mysql.jdbc.Driver";`
- ▶ `Class.forName(JDBC_DRIVER);`

這是MySQL資料庫的驅動程式。

```
▶ Class.forName("sun.jdbc.odbc.JdbcOdbcDriver"  
);
```

這是ACCESS資料庫的驅動程式。

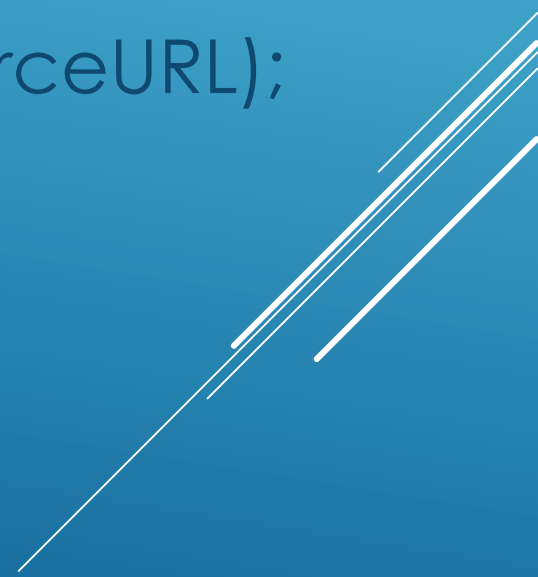
- ▶ 我們然後將mysql-connector-java-3.1.12-bin.jar驅動程式套件放到C:\Program Files\Java\jdk1.5.0\jre\lib\ext目錄下，也就是我們Java的jre\lib\ext目錄下。這樣JSP或 Servlet就可以透過JDBC來連接我們資料庫了。

- ▶ Connection為我們的連接物件，它會連接Java到指定的資料庫。
- ▶ 我們使用DriverManager類別的getConnection()靜態方法來連接資料庫。這是MySQL資料庫的連接。
- ▶ jdbc:mysql://localhost/My_Books是我們連接本地端的My_Books資料庫。
- ▶ root是我們MySQL資料庫的管理者，而他登錄的密碼是設定為322739。
- ▶

```
static final String DATABASE_URL =  
"jdbc:mysql://localhost/My_Books";
```
- ▶

```
Connection connection =  
DriverManager.getConnection( DATABASE_URL, "root",  
"322739" );
```

(2)建立連接

- ▶ 我們使用DriverManager類別的getConnection()靜態方法來連接資料庫。這是Access資料庫的連接。
 - ▶ jdbc.odbc:books是我們連接JDBC-ODBC的Access資料庫books。
 - ▶ String sourceURL = "jdbc:odbc:books";
 - ▶ Connection databaseConnection = DriverManager.getConnection(sourceURL);
- 

資料庫	URL 範本
Access 資料庫	<u>jdbc:odbc:資料庫來源</u>
MySQL 資料庫	<u>jdbc:mysql://主機名稱/資料庫名稱</u>
Oracle 資料庫	<u>jdbc:oracle:thin:@hostname:port#:oracleDBSID</u>

DATABASE_URL是資料庫在
網路上的識別名稱。

- ▶ 這是建立敘述物件statement。
Statement物件將我們的SQL敘述傳送到資料庫或從資料庫接收資料。當我們建立完連接物件Conection後，我們就要建立statement物件。使用createStatement()方法。

(3)建立敘述

- ▶ 這是執行敘述。executeQuery()會執行SQL的資料定義DDL或更新敘述。查詢的結果是以ResultSet的方式來回傳。我們在這裏將authors資料表的資料回傳給ResultSet資料集。
- ▶


```
ResultSet resultSet =  
statement.executeQuery( "SELECT authorID,  
Name FROM authors" );
```

(4) 執行敘述

- ▶ ResultSet是一個資料集，也可以看作是資料表，它是由許多資料列所集合而成。我們可以使用next()函數來移動到下一列資料，next()函數可以在ResultSet中作移動。我們可以使用ResultSet類別的各種get()函數來得到ResultSet資料集的資料。
- ▶ authorResults.getString("authid")的getString()得到資料集中每一列的authid欄位資料。
authorResults.getString("firstname")的getString()得到資料集中每一列的firstname欄位資料。
authorResults.getString("lastname")的getString()得到資料集中的每一列的lastname欄位資料。

(5) 處理結果資料集

- ▶ `ResultSet authorResults =`
`statement.executeQuery(queryWildcard);`
- ▶ `int row = 0;`
- ▶ `while(authorResults.next()) {`
- ▶ `System.out.println("Row " + (++row) + ") "+`
- ▶ `authorResults.getString("authId")+ " " +`
- ▶ `authorResults.getString("Name");`
- ▶ `}`



- ▶ 範例：Authors.java
- ▶ 第一行是輸入java.sql.Connection連接類別Connection。
- ▶ 第二行是輸入java.sql.Statement敘述類別Statement。
- ▶ 第三行是輸入java.sql.DriverManager管理驅動程式類別DriverManager。
- ▶ 第四行是輸入java.sql.ResultSet資料集類別ResultSet。
- ▶ 第五行是輸入java.sql.ResultSetMetaData資料集資料類別ResultSetMetaData。
- ▶ 第六行是輸入java.sql.SQLException，SQL例外類別SQLException。
- ▶ 第二十一行載入MySQL的驅動程式，使用Class.forName()來載入驅動程式。
- ▶ 第二十二行和第二十三行是建立連接到資料庫，使用Connection類別的物件connection。
- ▶ 第二十四行是建立查詢資料庫的敘述，使用Statement的類別物件statement。
- ▶ 第二十六行到第二十七行是查詢資料庫使用statement類別物件的executeQuery()函數。
- ▶ 第二十八行到第三十九行是處理查詢結果。
- ▶ 第二十八行ResultSetMetaData類別的metadata物件，它包含了ResultSet資料集的內容。
- ▶ 第二十九行的metadata.getColumnCount()可以得到ResultSet資料表的欄位個數。
- ▶ 第三十二行的metadata洗ColumnName()可以得到ResultSet資料表的欄位名稱。
- ▶ 第三十七行的ResultSet.getObject()函數可以得到ResultSet的特定欄位資料。
- ▶ 第四十九行是關閉敘述物件statement。
- ▶ 第五十行是關閉連接物件connection。

```
1 import java.sql. Connection;
2 import java.sql. Statement;
3 import java.sql. DriverManager;
4 import java.sql. ResultSet;
5 import java.sql. ResultSetMetaData;
6 import java.sql. SQLException;
7
8 public class Authors
9 {
10     // JDBC 驅動程式的名子和資料庫的位址
11     static final String JDBC_DRIVER = "com.mysql.jdbc.Driver";
12     static final String DATABASE_URL = "jdbc:mysql://localhost/My_Books";
13
14     public static void main( String args[] )
15     {
16         Connection connection = null;
17         Statement statement = null;
18
19         // 連接資料庫和查詢
20         try{
21             Class.forName( JDBC_DRIVER );
22             connection =
23                 DriverManager.getConnection( DATABASE_URL, "root", "322739" );
24             statement = connection.createStatement();
```

```
25
26      ResultSet resultSet = statement.executeQuery(
27          "SELECT authorID, Name FROM authors" );
28      ResultSetMetaData metaData = resultSet.getMetaData();
29      int numberOfColumns = metaData.getColumnCount();
30      System.out.println( "書籍作者的名子:" );
31      for ( int i = 1; i <= numberOfColumns; i++ )
32          System.out.printf( "%-8s\t", metaData.getColumnName( i ) );
33      System.out.println();
34      while ( resultSet.next() )
35      {
36          for ( int i = 1; i <= numberOfColumns; i++ )
37              System.out.printf( "%-8s\t", resultSet.getObject( i ) );
38          System.out.println();
39      }
40  }catch ( SQLException sqlException ){
41      sqlException.printStackTrace();
42      System.exit( 1 );
43  }catch ( ClassNotFoundException classNotFound ){
44      classNotFound.printStackTrace();
45      System.exit( 1 );
46  }finally{
47      try
48      {
```

```
49         statement.close();
50         connection.close();
51     }
52     catch ( Exception exception )
53     {
54         exception.printStackTrace();
55         System.exit( 1 );
56     }
57 }
58 }
59 }
```

1-7-1 編譯AUTHORS.JAVA

```
C:\>javac Authors.java
```

- ▶ 當我們寫好程式後，我們編譯該Authors.java程式。

- ▶ 我們執行Authors程式時要指定JDBC的類別路徑，
E:\Java2\database\example\mysql\mysql-connector-java-3.1.12\mysql-connector-java-3.1.12-bin.jar。
mysql-connector-java-3.1.12-bin.jar是我們的MySQL資料庫的JDBC連接套件。我們要指定放置它的路徑，在這邊的範例中，我們是將它放到E:\的目錄下，我們可以從光碟拷背出來，也可以在MySQL的網站下載。Access資料庫則已經內建在JDK中，就不用再設定路徑了。

1-7-2執行

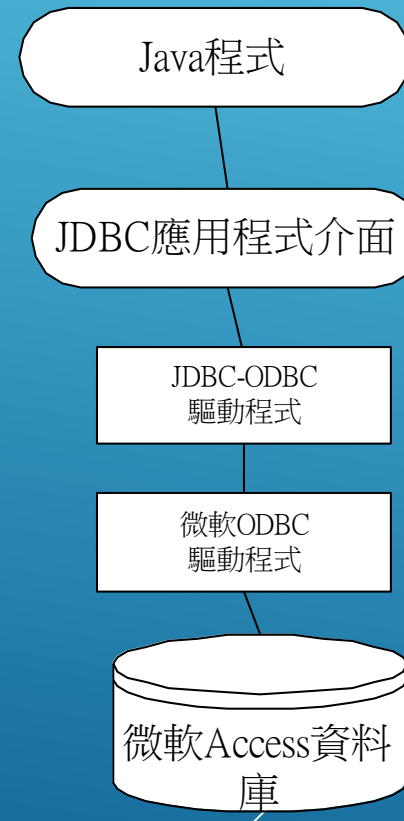

```
C:\>java -classpath E:\Java2\database\example\mysql-connector-java-3.1.12\mysql-connector-java-3.1.12\mysql-connector-java-3.1.12-bin.jar;. Authors
```

書籍作者的名子:

authorID	Name
1	Justin wu
2	Chaiyen wu
3	Miss Chen
4	Miss Lin
5	Good man

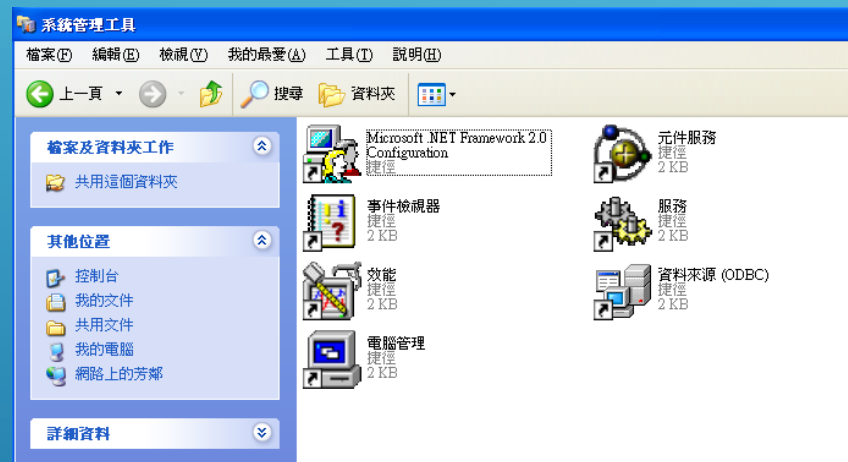
- ▶ 我們Java的程式，透過JDBC和ODBC介面來存取在Windows作業系統上的Access資料庫。

1-8使用JDBC來存取ACCESS資料庫



1-8-1 設定WINDOWS XP上的資料庫

- ▶ 我們在資料來源上選取資料來源ODBC。我們按兩下資料來源(ODBC)。





我們在使用者資料來源名稱
選取新增。

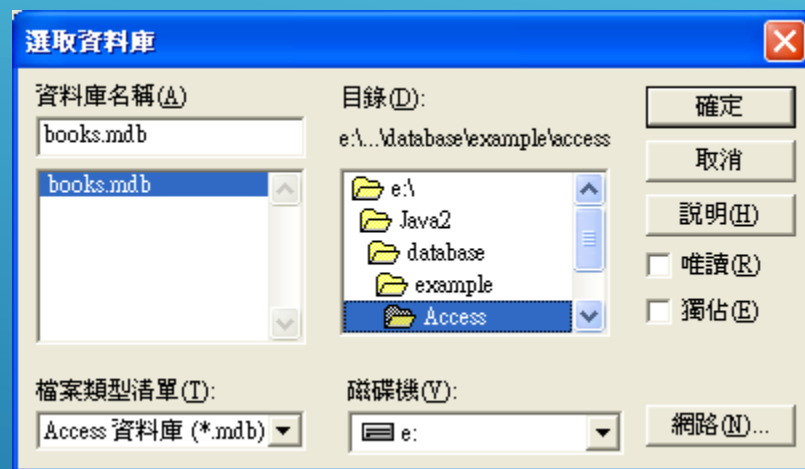


我們選取MICROSOFT ACCESS DRIVER(*.MDB)，我們再按下完成。

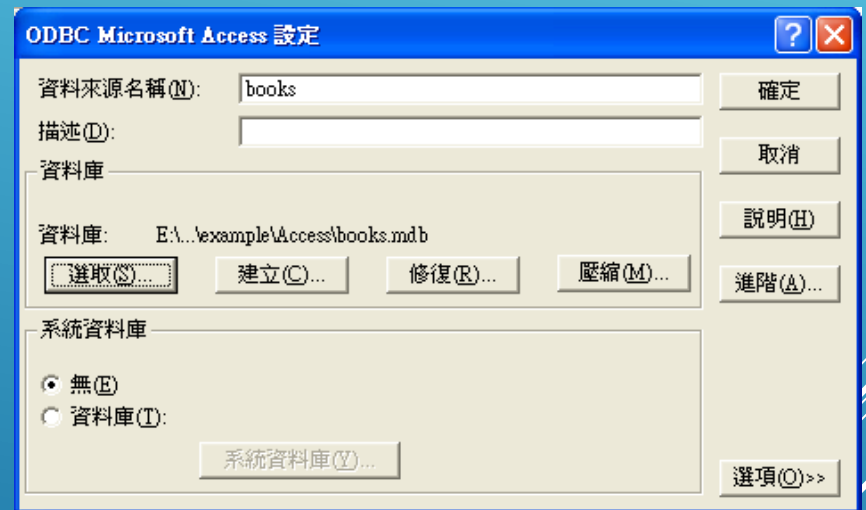


我們輸入資料來源名稱BOOKS，
我們選取所要連接的資料庫。
我們要選取資料庫，我們選取“
選取”。

- ▶ 我們選取books.mdb資料庫。我們可以將光碟範例中的Access資料庫放置到特定地方，再選取資料庫中來選取它。



- 這是我們選取連接資料庫後的情況，這樣就連接Windows Xp上的ODBC介面來連接Access資料庫。





這是我們已經透過ODBC介面來連接我們資料庫的情況。

- ▶ 我們在這裏使用JDBC-ODBC的橋接介面 `sun.jdbc.odbc.JdbcOdbcDriver` 來存取微軟的Access資料庫 `books.mdb`。我們在前面過程中，已經設定微軟的Access資料庫的ODBC驅動程式來連接 `books.mdb` 了。

- ▶ 這是要載入驅動程式
sun.jdbc.odbc.JdbcOdbcDriver，它是Access
資料庫的驅動程式。Class.forName()會載入
ODBC驅動程式。我們可以使用Class類別的靜態
函數forName()來載入驅動程式。假如驅動程式類
別找不到，forName()函數可能會丟出
ClassNotFoundException例外。當驅動程式被載
入時，驅動程式將建立它自己的實體，然後自動的
呼叫DriverManager類別方法來登錄該實體。在
DriverManager類別的方法都是靜態的。

- ▶ // 載入驅動類別

- ▶

```
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
```


1-8-2管理驅動程式

- ▶ DriverManager類別的靜態方法getConnection()會回傳Connection連接物件。sourceURL是定義Access資料庫books所在的位置。
- ▶ // 定義驅動程式的資料來源
- ▶ String sourceURL = "jdbc:odbc:books";
- ▶ // 使用DriverManager 來建立連接
- ▶ Connection databaseConnection = DriverManager.getConnection(sourceURL);

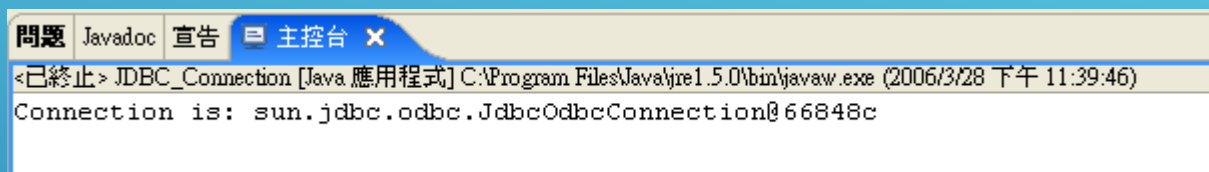
1-8-3建立連接資料來源

- ▶ // 定義驅動程式的資料來源
- ▶ String sourceURL = "jdbc:odbc:books";
- ▶ // 使用DriverManager 來建立連接
- ▶ Connection databaseConnection = DriverManager.getConnection(sourceURL,帳號,密碼);

除了連接ACCESS資料庫的位址，這是要登錄ACCESS資料庫的帳號和密碼。

- ▶ 範例：JDBC_Connection.java
 - ▶ 第一行是輸入sql套件中的連接類別Connection。
 - ▶ 第二行是輸入sql套件中的管理驅動程式類別DriverManager。
 - ▶ 第三行是輸入sql套件中的SQLException例外類別。
 - ▶ 第八行到第二十行為try...catch子句。我們將載入驅動程式放入到try子句中。
 - ▶ 第十行是載入JDBC-ODBC驅動程式。
 - ▶ 第十四行使用DriverManager類別的getConnection()函數來建立連接資料庫。
- 

```
1 import java.sql. Connection;
2 import java.sql. DriverManager;
3 import java.sql. SQLException;
4
5 public class JDBC_Connection{
6     public static void main(String[] args) {
7         // 載入驅動程式
8         try {
9             // 載入驅動類別
10            Class.forName(" sun. jdbc. odbc. JdbcOdbcDriver");
11            // 定義驅動程式的資料來源
12            String sourceURL = " jdbc:odbc:books";
13            // 使用DriverManager 來建立連接
14            Connection databaseConnection = DriverManager.getConnection(sourceURL);
15            System.out.println("Connection is: "+databaseConnection);
16        } catch(ClassNotFoundException cnfe) {
17            System.err.println(cnfe);
18        } catch(SQLException sqle) {
19            System.err.println(sqle);
20        }
21    }
22 }
```

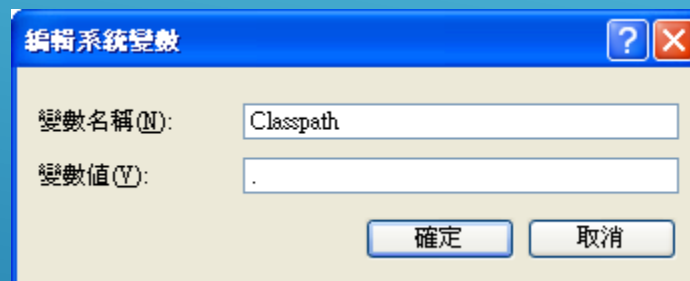


The screenshot shows the Eclipse IDE's console window. The title bar includes tabs for '問題' (Problems), 'Javadoc', '宣告' (Declarations), and '主控台' (Console). The console output displays a message indicating the termination of a Java application and the successful establishment of a JDBC connection.

```
<已終止> JDBC_Connection [Java 應用程式] C:\Program Files\Java\jre1.5.0\bin\javaw.exe (2006/3/28 下午 11:39:46)  
Connection is: sun.jdbc.odbc.JdbcOdbcConnection@66848c
```

這是使用ECLIPSE執行的結果。

- ▶ 我們選取開始→控制台→效能及維護→系統。我們在系統變數選取新增，新增變數名稱為Classpath類別路徑，並且變數值為.逗點。也就是設定目前所在位置為類別路徑。



```
E:\Java2\database\example\Access>javac JDBC_Connection.java  
E:\Java2\database\example\Access>java JDBC_Connection  
Connection is: sun.jdbc.odbc.JdbcOdbcConnection@66848c
```

這是執行的結果。

- ▶ JDBC-ODBC介面是包含在JDK裏面，它的介面是“`sun.jdbc.odbc.JdbcOdbcDriver`”，它讓Java的應用程式能從驅動程式來存取資料。JDBC-ODBC介面將JDBC方法轉移給ODBC函數呼叫。

1-8-4URLS和JDBC

- ▶ 範例：EssentialJDBC.java
- ▶ 第八行到第八十一行我們定義了EssentialJDBC類別。
- ▶ 第十四行到第二十行為我們主程式main()。
- ▶ 第十七行會呼叫SQLException物件的getResultsByColumnName()函數。
- ▶ 第十八行會呼叫SQLException物件的getResultsByColumnPosition()函數。
- ▶ 第十九行會呼叫SQLException物件的closeConnection()函數。
- ▶ 第二十一行到第三十一行為EssentialJDBC()建構子，它會連接books資料庫。
- ▶ 第二十五行是使用connection.createStatement()函數來建立statement敘述物件。
- ▶ 第三十二行到第四十九行使用欄位名稱來得到資料集的資料。
- ▶ 第三十四行使用statement物件的executeQuery()函數來執行SQL查詢。這回傳資料集的結果是以java.sql.ResultSet型態物件回傳。
- ▶ 第三十七行authorResults.next()是會求取下一筆資料列。
- ▶ 第三十九行authorResults.getString("authid")會得到books資料表中authid欄位的資料。
- ▶ 第四十三行authorResults.close()函數會關閉資料集。
- ▶ 第五十行到第六十七行的getResultsByColumnPostion()函數會得到查詢以欄位位置回傳的資料。
- ▶ 第五十八行authorResults.getString(i)會得到books資料表中，第i欄位的資料。
- ▶ 第六十九行到第八十行是關閉連接closeConnection()。
- ▶ 第七十二行是連接物件的關閉connection.close()。

```
1 import java.sql. Connection;
2 import java.sql. Statement;
3 import java.sql. DriverManager;
4 import java.sql. ResultSet;
5 import java.sql. ResultSetMetaData;
6 import java.sql. SQLException;
7
8 public class EssentialJDBC {
9     private Connection connection;
10    private Statement statement;
11    private String sourceURL = "jdbc:odbc:books";
12    private String queryIDAndName = "SELECT authid, firstname, lastname FROM authors";
13    private String queryWildcard = "SELECT * FROM authors";
14    public static void main (String[] args) {
15        //建立應用程式物件
16        EssentialJDBC SQLExample = new EssentialJDBC();
17        SQLExample.getResultsByColumnName();
18        SQLExample.getResultsByColumnPosition();
19        SQLExample.closeConnection();
20    }
21    public EssentialJDBC() {
22        try {
23            Class.forName("sun. jdbc. odbc. JdbcOdbcDriver");
24            connection = DriverManager.getConnection(sourceURL);
25            statement = connection.createStatement();
26        } catch(SQLException sqle) {
27            System.err.println("Error creating connection");
```

```

28     } catch(ClassNotFoundException cnfe) {
29         System.err.println(cnfe.toString());
30     }
31 }
32 void getResultByColumnName() {
33     try {
34         ResultSet authorResults = statement.executeQuery(queryWildcard);
35         int row = 0;
36
37         while(authorResults.next()) {
38             System.out.println("Row " + (++row) + ") " +
39                               authorResults.getString("authid") + " " +
40                               authorResults.getString("firstname") + " , " +
41                               authorResults.getString("lastname"));
42         }
43         authorResults.close();
44     } catch (SQLException sqle) {
45         System.err.println ("\nSQLException-----\n");
46         System.err.println ("SQLState: " + sqle.getSQLState());
47         System.err.println ("Message : " + sqle.getMessage());
48     }
49 }
50 void getResultByColumnPosition() {
51     try {
52         ResultSet authorResults = statement.executeQuery(queryIDAndName);
53
54         int row = 0;

```

```

55     while (authorResults.next()) {
56         System.out.print("\nRow " + (++row) + " ) ");
57         for(int i = 1 ; i<=3 ; i++) {
58             System.out.print((i>1?" , ":" ") + authorResults.getString(i));
59         }
60     }
61     authorResults.close();
62 } catch (SQLException ex) {
63     System.err.println("\nSQLException-----\n");
64     System.err.println("SQLState: " + ex.getSQLState());
65     System.err.println("Message : " + ex.getMessage());
66 }
67 }
68 //關閉連接
69 void closeConnection() {
70     if(connection != null) {
71         try {
72             connection.close();
73             connection = null;
74         } catch (SQLException ex) {
75             System.out.println("\nSQLException-----\n");
76             System.out.println("SQLState: " + ex.getSQLState());
77             System.out.println("Message : " + ex.getMessage());
78         }
79     }
80 }
81 }

```

```
問題 Javadoc 宣告 主控台 x
<已終止> EssentialJDBC [Java 應用程式] C:\Program Files\Java\jre1.5.0\bin\javaw.exe (2006/3/28 下午 11:43:03)
Row 1) 8 陳, 惠君
Row 2) 7 陳, 依鈴
Row 3) 6 江, 程邦
Row 4) 5 黃, 哲君
Row 5) 4 劉, 建宏
Row 6) 3 陳, 美玉
Row 7) 2 林, 小明
Row 8) 1 吳, 佳諺

Row 1) 8, 陳, 惠君
Row 2) 7, 陳, 依鈴
Row 3) 6, 江, 程邦
Row 4) 5, 黃, 哲君
Row 5) 4, 劉, 建宏
Row 6) 3, 陳, 美玉
Row 7) 2, 林, 小明
Row 8) 1, 吳, 佳諺
```

這是執行的結果。


```

1 import java.sql. Connection;
2 import java.sql. DriverManager;
3 import java.sql. SQLException;
4 import java.sql. Statement;
5 import java.sql. ResultSet;
6
7 public class MakingAStatement {
8     public static void main(String[] args) {
9         // 載入驅動程式
10        try {
11            // 載入驅動程式類別
12            Class.forName("sun. jdbc. odbc. JdbcOdbcDriver");
13            // 定義驅動程式的資料來源
14            String sourceURL = new String(" jdbc:odbc:books");
15            // 使用DriverManager來建立連接
16            Connection databaseConnection = DriverManager.getConnection(sourceURL);
17            Statement statement = databaseConnection.createStatement();
18            ResultSet authorNames = statement.executeQuery("SELECT firstname,
19                                                         lastname FROM authors");
20            // 輸出結果
21            while(authorNames.next()) {
22                System.out.println(authorNames.getString("firstname")+
23                                   authorNames.getString("lastname"));
24            }
25        } catch(ClassNotFoundException cnfe) {
26            System.err.println(cnfe);
27        } catch(SQLException sqle) {
28            System.err.println(sqle);
29        }
30    }
31 }

```

範例：MAKINGASTATEMENT.JAVA
 第十八行到第十九行是執行緒述使用
 STATEMENT物件的EXECUTEQUERY()函數來
 執行SQL敘述。

```
<已終止> MakingAStatement [Java 應用程式] C:\Program Files\Java\jre1.5.0\bin\javaw.exe (2006/3/28 下午 11:46:42)
陳          惠君
陳          依鈴
江          程邦
黃          哲君
劉          建宏
陳          美玉
林          小明
吳          佳謬
```

這是在ECLIPSE執行的結果。

```
E:\Java2\database\example\Access>javac MakingAStatement.java  
E:\Java2\database\example\Access>java MakingAStatement  
陳 惠君  
陳 依鈴  
江 程邦  
黃 哲君  
劉 建宏  
陳 美玉  
林 小明  
吳 佳謬
```

這是在命令提示字元下執行的情況。