Structured Query Language(SQL) and Database Basics

What is SQL?

- SQL stands for Structured Query Language
- SQL lets you access and manipulate databases
- SQL is an ANSI (American National Standards Institute) standard

What Can SQL do?

- SQL can execute queries against a database
- SQL can retrieve data from a database
- SQL can insert records in a database
- SQL can update records in a database
- SQL can delete records from a database
- SQL can create new databases
- SQL can create new tables in a database
- SQL can create stored procedures in a database

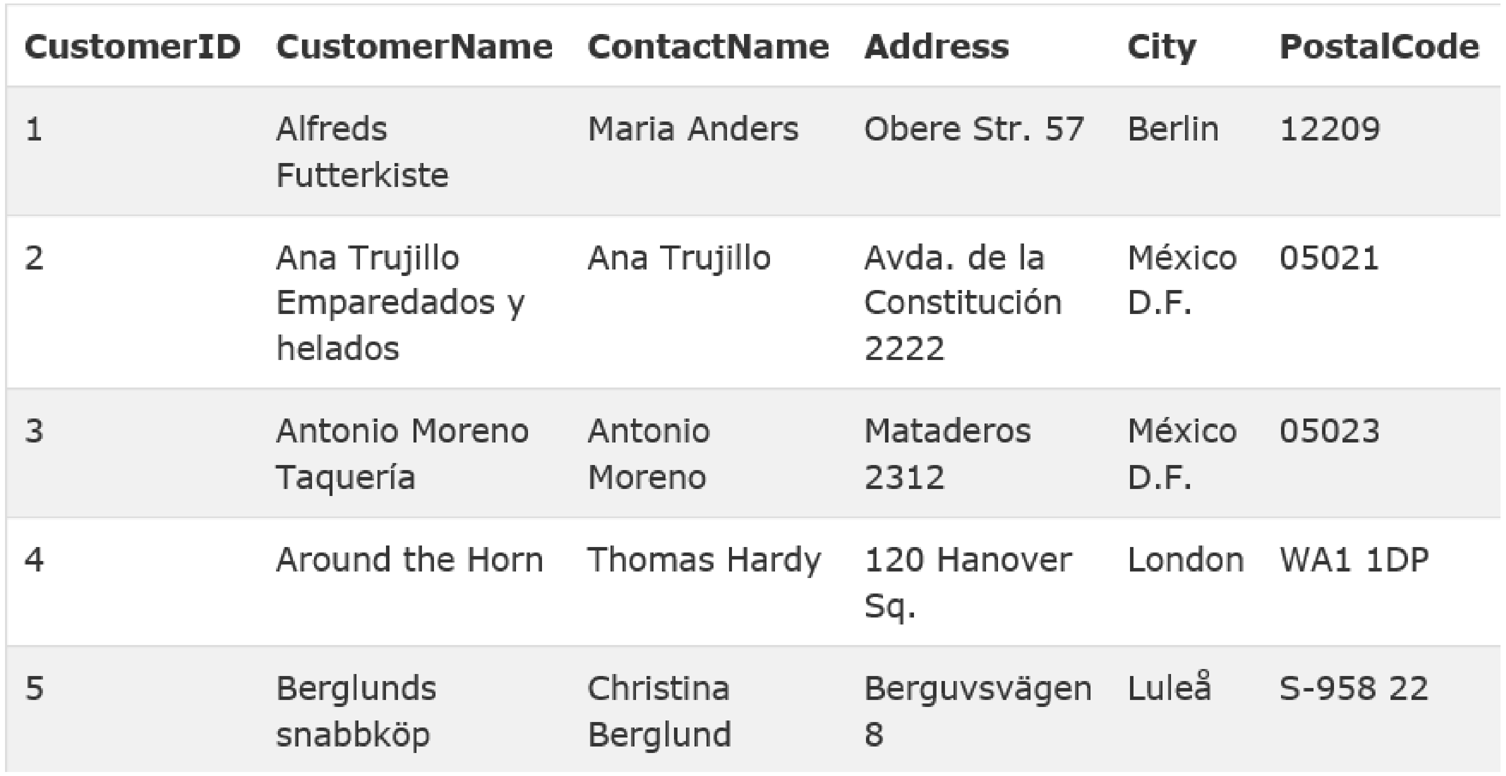
SQL is a Standard - BUT….

SQL is an ANSI (American National Standards Institute) standard, there are different versions of the SQL language.

To be compliant with the ANSI standard, they all support at least the major commands (such as SELECT, UPDATE, DELETE, INSERT, WHERE) in a similar manner.

Database Tables

A database most often contains one or more tables. Each table is identified by a name (e.g. "Customers" or "Orders"). Tables contain records (rows) with data



CustomerID	CustomerName	ContactName	Address	City	PostalCode
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitución 2222	México D.F.	05021
3	Antonio Moreno Taquería	Antonio Moreno	Mataderos 2312	México D.F.	05023
4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP
5	Berglunds snabbköp	Christina Berglund	Berguvsvägen 8	Luleå	S-958 22

SQL Statements

Most of the actions you need to perform on a database are done with SQL statements

SELECT - extracts data from a database

UPDATE - updates data in a database

DELETE - deletes data from a database

```
SELECT * FROM Customers;
```

Selects all records all fields from the Customers table

INSERT INTO - inserts new data into a database

CREATE DATABASE - creates a new database

ALTER DATABASE - modifies a database

CREATE TABLE - creates a new table

ALTER TABLE - modifies a table

DROP TABLE - deletes a table

CREATE INDEX - creates an index (search key)

DROP INDEX - deletes an index

SQL SELECT Statement

The SELECT statement is used to select data from a database.

The result is stored in a result table, called the result-set

SELECT *column_name*,*column_name*
FROM *table_name*;

Or

Select * (selects all columns not just specified)

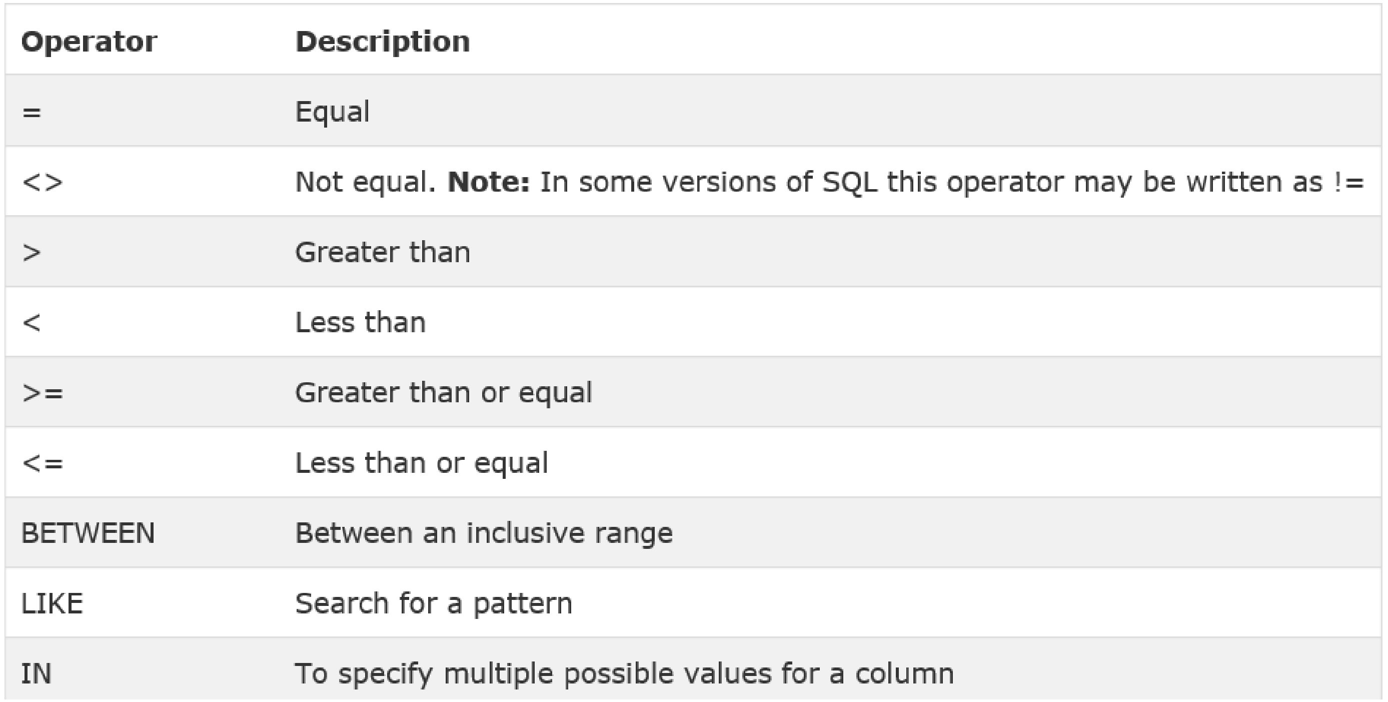
Both Commands return a result set that can be accessed via Java commands

SQL WHERE Clause

The WHERE clause is used to extract only those records that fulfill a specified criterion

```
SELECT column_name,column_name
FROM table_name
WHERE column_name operator value;
```

```
SELECT * FROM Customers
WHERE Country='Germany‘
```



Operator	Description
=	Equal
<>	Not equal. **Note:** In some versions of SQL this operator may be written as !=
>	Greater than
<	Less than
>=	Greater than or equal
<=	Less than or equal
BETWEEN	Between an inclusive range
LIKE	Search for a pattern
IN	To specify multiple possible values for a column

SQL AND & OR Operators

The AND operator displays a record if both the first condition AND the second condition are true.

The OR operator displays a record if either the first condition OR the second condition is true

```
SELECT * FROM Customers
WHERE Country='Germany'
AND City='Berlin';
```

The SQL LIKE Operator

The LIKE operator is used to search for a specified pattern in a column

SELECT *column_name(s)*
FROM *table_name*
WHERE *column_name* LIKE *pattern*

```
SELECT * FROM Customers
WHERE City LIKE 's%';
```

```
SELECT * FROM Customers
WHERE City LIKE '%s';
```

```
SELECT * FROM Customers
WHERE Country NOT LIKE '%land%';
```

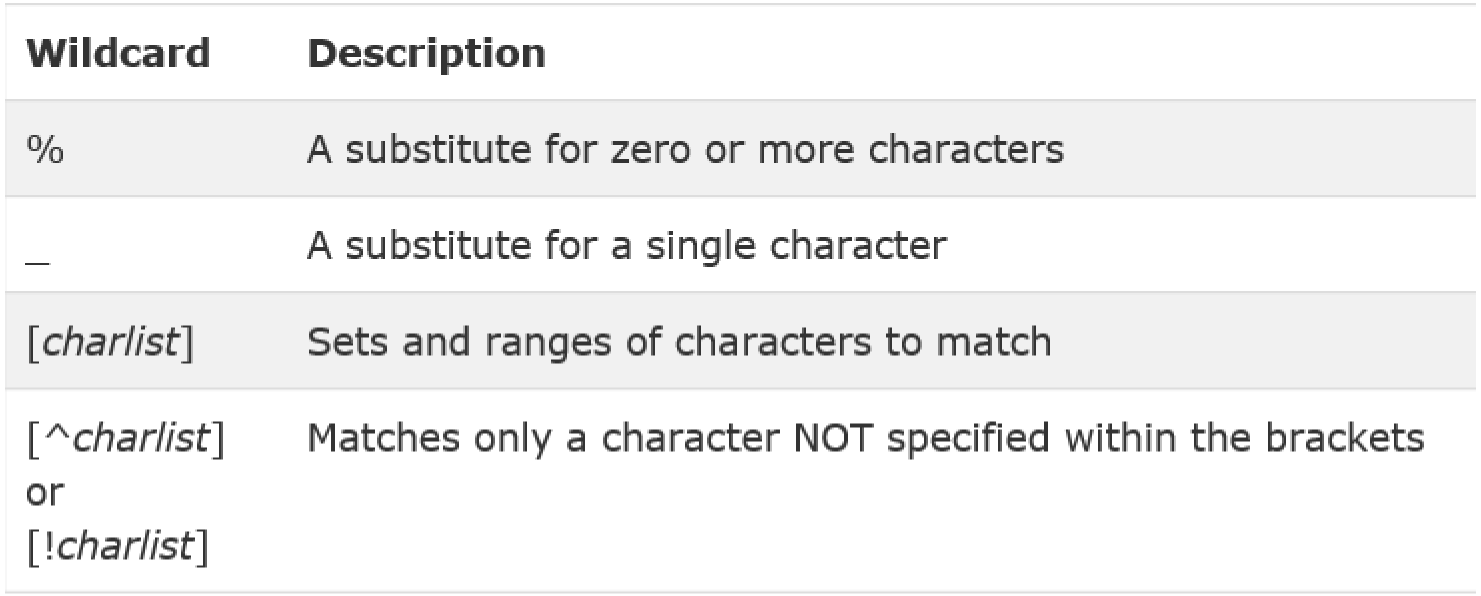
SQL Wildcards

In SQL, wildcard characters are used with the SQL LIKE operator.

SQL wildcards are used to search for data within a table.

```
SELECT * FROM Customers
WHERE City LIKE 'ber%';
```

```
SELECT * FROM Customers
WHERE City LIKE '[a-c]%';
```



Wildcard	Description
%	A substitute for zero or more characters
_	A substitute for a single character
[*charlist*]	Sets and ranges of characters to match
[^*charlist*] or [!*charlist*]	Matches only a character NOT specified within the brackets

SQL ORDER BY Keyword

The ORDER BY keyword is used to sort the result-set by one or more columns.

The ORDER BY keyword sorts the records in ascending order by default. To sort the records in a descending order, you can use the DESC keyword

SQL INSERT INTO

The INSERT INTO statement is used to insert new records in a table

It is possible to write the INSERT INTO statement in two forms.

The first form does not specify the column names where the data will be inserted, only their values:

INSERT INTO *table_name*
VALUES (*value1*,*value2*,*value3*,...);

INSERT INTO *table_name* (*column1*,*column2*,*column3*,...)
VALUES (*value1*,*value2*,*value3*,...);

INSERT INTO Customers (CustomerName, City, Country)
VALUES ('Cardinal', 'Stavanger', 'Norway');

SQL UPDATE

The UPDATE statement is used to update existing records in a table

```
UPDATE table_name
SET column1=value1,column2=value2,...
WHERE some_column=some_value
```

Example:

```
UPDATE Customers
SET ContactName='Alfred Schmidt', City='Hamburg'
WHERE CustomerName='Alfreds Futterkiste';
```

Update Warning!

Be careful when updating records

An omitted where clause will update all records.

It is also easy to specify a where clause that is broader than you intended

You can **TEST** your were clause by using it in a **SELECT** statement

SQL DELETE

the DELETE statement is used to delete rows in a table

DELETE FROM *table_name*
WHERE *some_column=some_value*;

Example:

SQL Joins

DELETE FROM Customers
WHERE CustomerName='Alfreds Futterkiste' AND ContactName='Maria Anders';

Delete Warning!

Be careful when updating records

An omitted where clause will Delete all records.

It is also easy to specify a where clause that is broader than you intended

You can **TEST** your were clause by using it in a **SELECT** statement

SQL CREATE DATABASE Statement

The CREATE DATABASE statement is used to create a database

Example

```
CREATE DATABASE my_db;
```

SQL CREATE TABLE

The CREATE TABLE statement is used to create a table in a database.

Tables are organized into rows and columns; and each table must have a name.

```
CREATE TABLE table_name
(
column_name1 data_type(size),
column_name2 data_type(size),
column_name3 data_type(size),
....
);
```

Note the Format
-Makes copying easier

The column_name parameters specify the names of the columns of the table.

The data_type parameter specifies what type of data the column can hold (e.g. varchar, integer, decimal, date, etc.).

The size parameter specifies the maximum length of the column of the table.

SQL PRIMARY KEY

The PRIMARY KEY constraint uniquely identifies each record in a database table.

Primary keys must contain UNIQUE values.

A primary key column cannot contain NULL values.

Most tables should have a primary key, and each table can have only ONE primary key

Example: **Note different Database Management Systems (DBMS) will have slightly different syntax**

```
CREATE TABLE Persons
(
P_Id int NOT NULL PRIMARY KEY,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Address varchar(255),
City varchar(255)
)
```

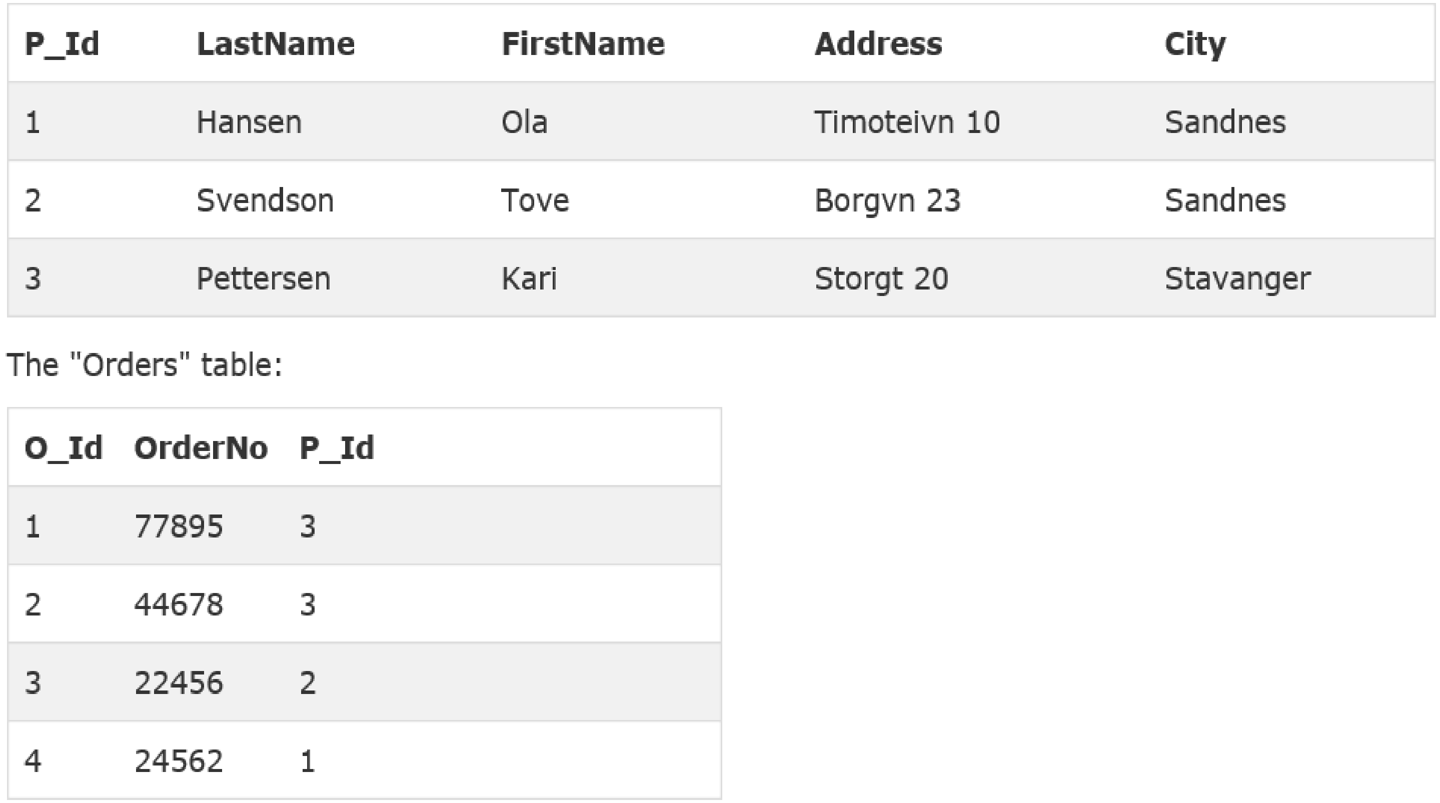
SQL FOREIGN KEY

A FOREIGN KEY in one table points to a PRIMARY KEY in another table.

Note that the "P_Id" column in the "Orders" table points to the "P_Id" column in the "Persons" table.

The "P_Id" column in the "Persons" table is the PRIMARY KEY in the "Persons" table.

The "P_Id" column in the "Orders" table is a FOREIGN KEY in the "Orders" table.



P_Id	LastName	FirstName	Address	City
1	Hansen	Ola	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

The "Orders" table:

O_Id	OrderNo	P_Id
1	77895	3
2	44678	3
3	22456	2
4	24562	1

SQL Joins

SQL joins are used to combine rows from two or more tables

An SQL JOIN clause is used to combine rows from two or more tables, based on a common field between them.

The most common type of join is: **SQL INNER JOIN (simple join)**. An SQL INNER JOIN return all rows from multiple tables where the join condition is met

INNER JOIN: Returns all rows when there is at least one match in BOTH tables

LEFT JOIN: Return all rows from the left table, and the matched rows from the right table

RIGHT JOIN: Return all rows from the right table, and the matched rows from the left table

FULL JOIN: Return all rows when there is a match in ONE of the tables

Exercise 1: QuoteData.java, P. 294-6

Yahoo! offers a downloadable spreadsheet link on its home page for each ticker symbol. This link can be found at http://quote.yahoo.com/q?s=fb&d=v1 Below the price chart, there is a download data link. The file has the data stored in a single line of data of the latest close information. The data is in order by ticker symbol. It contains the symbol, closing price, date, time, price change since last close, daily low, daily high, daily open and the volume.

The application we are about to develop, QuoteData.java, uses each of these fields except one, the time, to load our stock data.

Exercise 1: QuoteData.java

The following actions take place:

A stock ticker symbol is used as a command-line argument

A QuoteData object is created with the ticker symbol as an instance variable called ticker

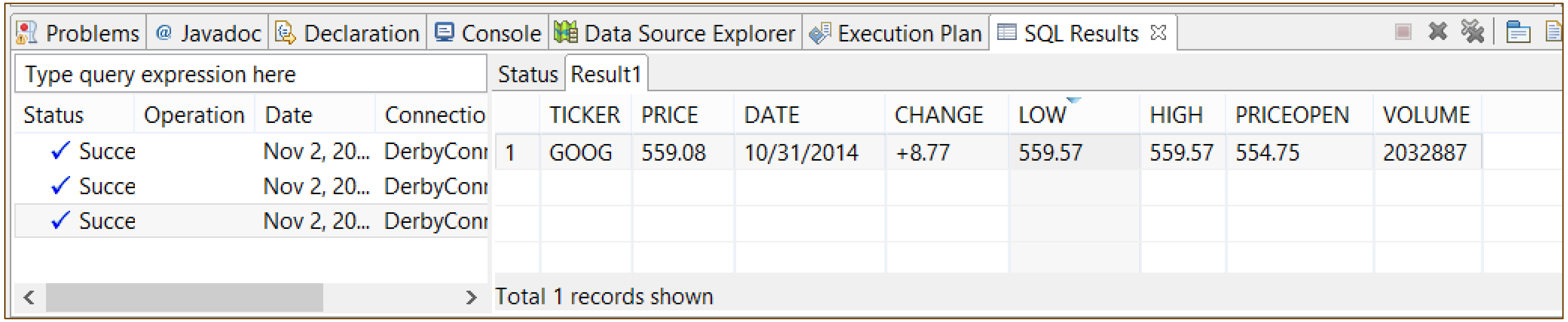
The object’s retrieveQuote() method is called to download the stock data from Yahoo! and returns it as a String

The object’s storeQuote() method is called with that String as an argument. It saves the data to a database.

Before we can begin, we must create a table in our database to hold the data. Please run the AddATable.java file in your student files to create the STOCKS table under the derbyDB for the USER1 schema.

Once we have the STOCKS table, code the following module and test it accordingly.

output



Problems | Javadoc | Declaration | Console | Data Source Explorer | Execution Plan | SQL Results

Type query expression here

Status	Operation	Date	Connectio
✓ Succe		Nov 2, 20...	DerbyConı
✓ Succe		Nov 2, 20...	DerbyConı
✓ Succe		Nov 2, 20...	DerbyConı

Status | Result1

	TICKER	PRICE	DATE	CHANGE	LOW	HIGH	PRICEOPEN	VOLUME
1	GOOG	559.08	10/31/2014	+8.77	559.57	559.57	554.75	2032887

Total 1 records shown

Let’s examine the code

This application stores data but does not produce output per se. We can use the extract feature in Luna to see that GOOG was entered in the database table STOCKS.

Lines 13 – 34: Download the Yahoo! data and save it as a string

Lines 36 – 69: Use SQL to store the data to the database. We use StringTokenizer to split the data into its tokens on the comma character that is between each token. We then store this in a String array with nine elements. The array contains the same elements as the table we created.

Lines 42 – 46: Connect to the database

Lines 49 – 53: Prepare our statements – an insert statement is used to store the data

Lines 54 – 61: Put the elements of the String array into the prepared statement using a series of setString() method. They must be put in the statement in the order of the database columns

Line 62: Execute the statements after the prepared statement has been filled using the executeUpdate() – it will either add it to the database or throw a SQL error.

Line 40: Strip any excess quotes from the data using the stripQutoes() private method

Moving through a result set

Since we used prepared statements we have the following methods available to us to help navigate the set. This is because the result set's policies have been specified as arguments to the connection's **createStatement()** and **prepareStatement()** methods. The **ResultSet** class includes many class variables to give us more flexibility when working with sets.

- afterLast() – go to the place immediately after the last record

- beforeFirst() – go to the place immediately before the first record

- first() – go to the first record

- last() – go to the last record

- previous() – go to the previous record in the set

Summary

We’ve learned to read and write database records using classes that work with the most popular relational database products. The only difference is that the database driver must be specific to the product we are using.

We also received a very brief view into database technologies and how a program communicates with it to retrieve and store data.

Independent exercises, P. 297

A. Modify the SysTableReporter application to pull fields from another table in SYS. You will have to make modifications to the connection strings and the drivers for this application. The Access database is in the student files.

B. Write an application that reads and displays records from Yahoo! Stock quote database.

SQL Resources

Tutorials

http://www.w3schools.com/SQl/default.asp

http://beginner-sql-tutorial.com/sql.htm

MySQL

http://www.mysql.com/