Uppsala University
Department of Information Technology
Kjell Orsborn

DATABASE DESIGN I - 1DL300

Assignment 3 - SQL queries, views and JDBC API

1 Part one – SQL queries and views

The purpose of this exercise is to practice writing both simple and more advanced queries in SQL, including the use of aggregate functions and views. This part of the assignment should be completed using the Mimer database management system.

1.1 Preparation

If needed install Mimer (not necessary if you are not using your own PC) and then set up the Jonson Brothers database. Instructions and scripts can be found at the assignment web page found from the course webpage.

- Elmasri and Navathe [EN10]: Chapter 4, 5 and 12.3.
- Padron-McCarthy and Risch [PMR05]: Chapter 7, 8, and 9.
- MIMER SQL documentation [M10].

There is an supervised introduction to the assignment in your schedule.

1.2 Assignment

In this assignment you should practice to use SQL to pose simple and more advanced queries towards The Jonson Brothers retail company database that you were introduced to in the previous assignment (i.e. Assignment 2). You will try several SQL features including for instance subqueries, aggregated queries and views.

1.3 Exercises

Use SQL to find the answers to the questions below towards your Jonson Brothers company database. Whenever a question requests information about entities that have both a number and a name, select both the number and the name to make your results more useful. For example, in Question (3) return both the number and the name of the parts that were asked for.

1. List all employees, i.e. all tuples in the `EMPLOYEE` relation.
2. List names of all departments, i.e. the `NAME` attribute for all tuples in the `DEPT` relation.
3. What parts are not in store, i.e. `QOH=0`? (QOH = Quantity On Hand).
4. Which employees have a salary between 9000 and 10000 (inclusive)?
5. What was the age of each employee (i.e. `STARTAGE`) when they started working here?
6. Which employees have a last name ending with "`son`" (in queries you can also use simple quotes: `'son'`)? Retrieve employee names and numbers.
7. Which items have been delivered by a supplier called "`Playskool`"? Formulate this query using a subquery in the where-clause.
8. Formulate the same query as above, but without a subquery.
9. What is the name and the color of the parts that are heavier than a black tape drive? Formulate this query using a subquery in the where-clause. (The SQL query should not contain the parts weight as a constant.)
10. Formulate the same query as above, but without a subquery. (Again, the query should not contain the weight as a constant.)
11. What is the average weight of black parts?

12. What is the total weight of all parts that each supplier in the state Massachusetts (i.e. "`Mass`") has delivered? Retrieve the total weight for each of these suppliers.

13. Create a new relation (a table) that contains the items that cost less than the average price for all items. List the contents of the new table, including item names.

14. Create a view that contains the items that cost less than the average price for all items. Query the view and show the result. What is the difference between (13) and (14)?

15. Create a view that calculates the total cost for all sales contained in each transaction by considering price and quantity of each bought item. (To be used for charging customer accounts). The view should return transaction number and total cost. Query the view and show the result.

16. Suddenly, an earthquake strikes. Explain, how you would proceed to remove all suppliers in Los Angeles from the table `SUPPLIERS`? (How would the corresponding SQL statements look like?) Do **not** execute!

17. A database manager in the company has tried to find out which suppliers have delivered items that are sold. He has created a help view and can find how many items are sold from each supplier of the items:

```
1 SQL> create view sale_supply(supplier, item, quantity) as
       select supplier.name, item.name, sale.quantity
       from supplier, item, sale
       where supplier.number = item.supplier and
             sale.item = item.number;
Ok

2 SQL> select supplier,sum(quantity) from sale_supply
       group by supplier;

6 rows selected
----------------------------------
|      SUPPLIER      |    SUM    |
----------------------------------
|       Cannon       |     6     |
|       Koret        |     1     |
|    Levi-Strauss    |     1     |
|     Playskool      |     2     |
|     White Stag     |     4     |
|     Whitmans       |     2     |
----------------------------------
```

Now he would also like to find out suppliers that have delivered some items none of which have been sold. Help him! Drop and redefine the `sale_supply` view in such a way that items that have been delivered by suppliers, but have never been sold, are considered as well. Repeat the

above query on your view.

Hint: The above definition of `sale_supply` uses an (implicit) inner join. An inner join removes suppliers that have not had any of their delivered items sold. Consider to replace the inner join with some other types of join.

1.4 Examination

The following should be handed in:

1. All SQL commands issued
2. All command results from the database server
3. Answers to questions, and explanations where appropriate

You should hand in the solutions and answers to questions as one report per group.

2 Part two – Java Database Connection using the JDBC API

Create and run a JDBC program using the client driver and Network Server. This part of the assignment demonstrates the ease to connect to a DBMS through a JDBC-based client/server connection between a Java-client and a DBMS Server.

2.1 Preparations (on Solaris)

1. Read tutorials and related material about JDBC provided on the course website.
2. Locate the Java DB/Derby database management system on your system and have a look at it. On Solaris workstations you will find it under /it/sw/java/JDK/6/db/.
3. Create a directory for Assignment 3 in your home directory, using this command in a terminal window:

   ```
   mkdir ~/assignment3
   ```

4. Enter the new directory:

   ```
   cd ~/assignment3
   ```

5. Create a file called `derby.properties` and put it in your Assignment 3 directory. As content in your `derby.properties` file, put the following row: `derby.drda.portNumber=xxxxx` (a port number). This number should be different than the port number used by your neighbors sitting on the other computers. Important! In order to avoid collisions please use a port number created like this 287gg, where gg is your group number. You can create the file with the following command (don't forget to replace `xx` with your group number):

   ```
   echo "derby.drda.portNumber=287xx" > derby.properties
   ```

6. Download the demo program `demoJDBCclient.java` from the course website and save it in the Assignment 3 directory. You can do it with the following command (or by downloading through your favorite browser):

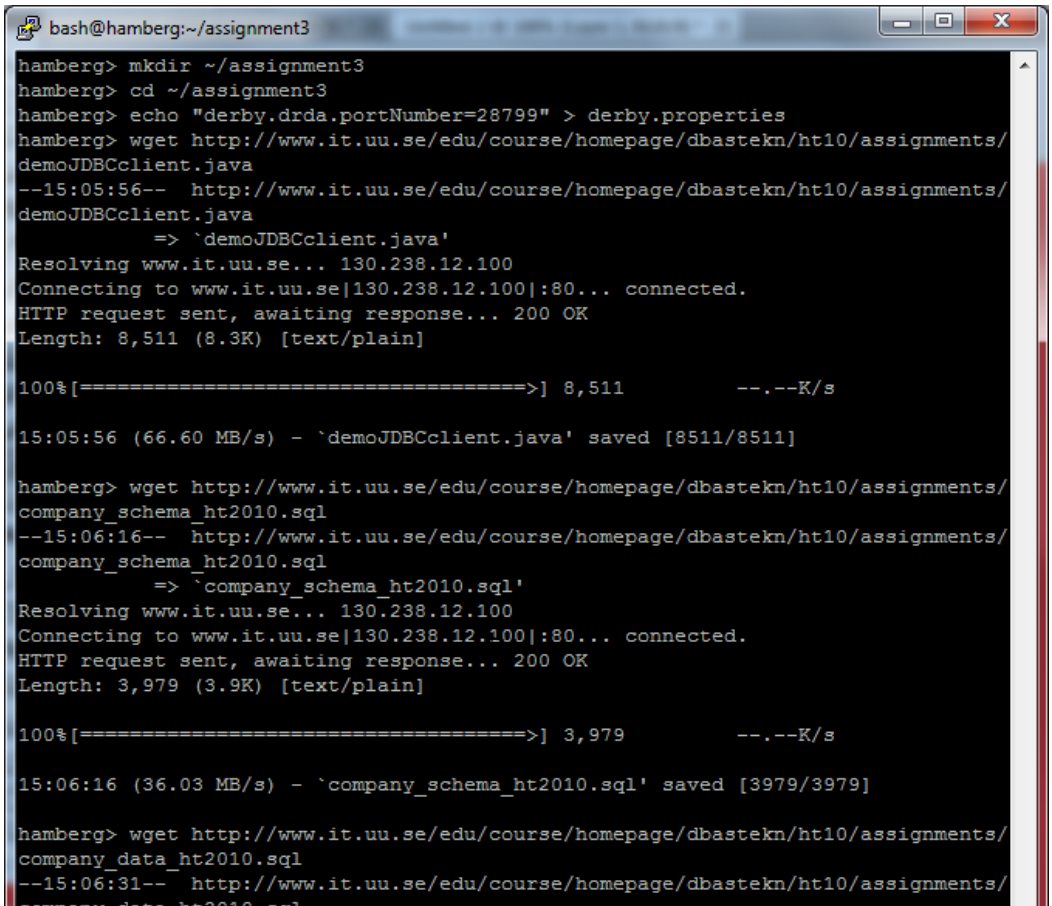
   ```
   wget http://www.it.uu.se/edu/course/homepage/dbastekn/ht10/assignments/demoJDBCclient.java
   ```

7. Download both files with the company database schema and the company database data from the course website. Again, you can use the `wget` utility:

   ```
   wget http://www.it.uu.se/edu/course/homepage/dbastekn/ht10/assignments/company_schema_ht2010.sql
   wget http://www.it.uu.se/edu/course/homepage/dbastekn/ht10/assignments/company_data_ht2010.sql
   ```

8. Edit your copy of the `demoJDBCclient.java` (for instance in Xemacs) in order to fulfill the tasks in Section 2.2.

   In the end, your terminal will look similar to the following:



2.2 Exercises of part 2

It should be noted that this part of the assignment only concerns one table, the Employee table, of the Jonson Brothers database. It covers the use of the JDBC database API to create, connect, extract data and update data from the database (a Java DB / Derby database in this case). The tasks are the following:

1. Connect to the Jonson Brothers database, create and populate the Employee table. You use the same Jonson Brothers database that you have been using so far in the Assignments 2 and 3.
2. Use a query to retrieve the identity number, the name and the salary of the employees at Jonson Brothers. Print out the information.
3. Update the salary for one or several employees by your own criterion. Then use a query to extract the identity number, the name and the salary of the employees at Jonson Brothers again and print it out.
4. Delete the Employee table and close the connection. It is assumed that you have a working directory called assignment3 and have downloaded the `demoJDBCclient.java` file at that directory together with the schema and data files for The Jonson Brothers database. You can learn how to connect between a Java client-program and a database server (here Java DB) using JDBC by studying the demo-program `demoJDBCclient.java` and the material supplied on the course page.
   (a) You will get basic knowledge and experience on how to start and connect to the *network database server*.
   (b) You can make a copy of the demo client-program and use it as a basis for your client-program `TheBrosClient.java`. By editing this program you should make necessary changes to fulfill the requirements for a java client for the Jonson Brothers database.

2.3 Guide to run database server and client program (on Solaris)

1. Create your `TheBrosClient.java` file using the example file `demoJDBCclient.java`, put it in your `assignment3` catalog and compile it by typing the following command in a terminal window:

   ```
   javac TheBrosClient.java
   ```

2. Set up the client/server environment using the following steps:

(a) Open a new terminal window

(b) Change to the Assignment 3 directory:
`cd ~/assignment3`

(c) Start the *network database server* by typing:

```
java -classpath /it/sw/java/JDK/6/db/lib/derbyrun.jar \
   -jar /it/sw/java/JDK/6/db/lib/derbyrun.jar server start
```

You terminal will look similar to the following:

3. Run the client program using the following steps:

   (a) Open *another* terminal window and go to your assignment3 directory where you should have the class file received by compiling your java program

   (b) Run your java program:

   ```
   java -classpath /it/sw/java/JDK/6/db/lib/derbyclient.jar:. TheBrosClient
   ```

   Note! The -classpath option sets the classpath to include the location of the file `derbyclient.jar`. Important: Include the dot (.) at the end of the command, before the name of the java class, so that your current working directory is included in the classpath as well.

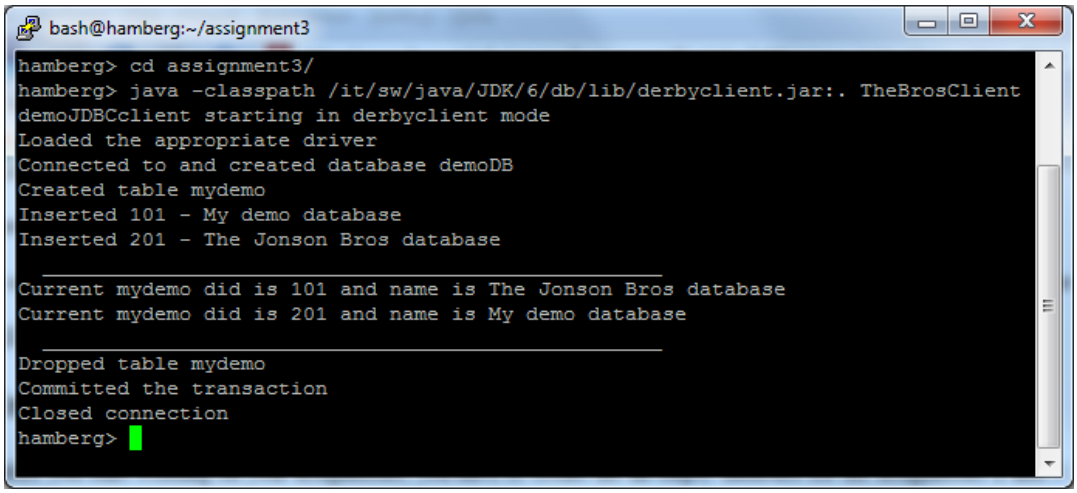
   You terminal will look similar to the following:

4. You can shut down the server by sending a shutdown command to the *network database server* (from the client shell):

```
java -classpath /it/sw/java/JDK/6/db/lib/derbyrun.jar \
  -jar /it/sw/java/JDK/6/db/lib/derbyrun.jar server shutdown
```

2.4 Notes on client-server environments

In a client-server environment, the client program is often used from other computers on the network. Whenever a system accepts connections from other



computers, there is a chance of abuse. To maintain security, the Derby Network Server defaults to accepting connections only from clients running on the local machine (localhost). Before this or any other Derby client program can access the Network Server from another machine, additional steps should be taken to secure the Network Server environment. Once secured, the Network Server can be safely configured to accept connections from other machines. Refer to the Network Server security and Running the Network Server under the security manager sections of the Java DB Server and Administration Guide for important information on securing the Network Server and enabling network connections.

With the Network Server started, you can run the client program simultaneously in multiple windows. To demonstrate this, open two terminal windows and perform the sub-steps of the Run the client program step in each terminal window. Both clients will operate without a problem. In contrast, it would not be possible for a program that uses the JavaDB embedded driver to access the database until the database or the Network Server is shut down.

You may have noticed that the client program does not shut down the database. This is because the database is a shared resource in a client-server environment and, in most cases, should be shut down only when the Network Server is shut down. If multiple clients are accessing the database and one client shuts down the database, the remaining clients will encounter a failure the next time they attempt an SQL command.

2.5 Examination of part 2

Hand in the source code of your Java program to your appointed assistant and also include a printout from executing the program. Your java program has to complete all four tasks in Section 2.2.

References

[EN10] Elmasri, R. and Navathe, S. B.: Fundamentals of Databases, 6th Edition, Addison-Wesley, 2010 (available e.g. at Akademibokhandeln).

[PMR05] Padron-McCarthy, T. and Risch, T.: Databasteknik, Studentlitteratur, 2005 (available e.g. at Akademibokhandeln).

[M10] Mimer documentation, Version 9.2 (html) (include User's Manual, Reference Manual and Programmer's Manual), `http://developer.mimer.com/documentation/html_92/Mimer_SQL_Engine_DocSet/Mimer_SQL_Engine.htm`.