LINKÖPING UNIVERSITY

DEPARTMENT OF COMPUTER AND INFORMATION SCIENCE (IDA)
DIVISION FOR DATABASE AND INFORMATION TECHNIQUES (ADIT)

Database Technologies
Assignment 3: Advanced SQL

Objective

The objective of this assignment is to practice using more advanced features of the database language SQL, including the use of aggregations and views.

Background Reading

Lecture material and book chapters about SQL. Note that small discrepancies might exist between some SQL interpreters and some books, as they follow slightly different SQL standards.

Introduction

This assignment is based again on the Jonson Brothers database as used in the first assignment. For an overview of this database, see the appendix of the document that describes the first assignment.

Tasks

Write an SQL statement for each of the following points.

1. List the name of all departments in alphabetical order. Note: by "name" we mean the name attribute in the *jbdept* relation.
2. Which items (note **items**, not parts) have been delivered by a supplier called *Fisher-Price*? Formulate this query by using a subquery in the WHERE clause.
3. List all cities that have suppliers located in them. Formulate this query using a subquery in the WHERE clause.
4. What is the name and the color of the parts that are heavier than a card reader? Formulate this query using a subquery in the WHERE clause. (The query must not contain the weight of the card reader as a constant; instead, the weight has to be retrieved within the query.)
5. Formulate the same query as above, but without a subquery. Again, the query must not contain the weight of the card reader as a constant.
6. What is the average weight of all black parts?
7. For every supplier in Massachusetts ("Mass"), retrieve the name and the total weight of all parts that the supplier has delivered? Do not forget to take the quantity of delivered parts into account. Note that one row should be returned for each supplier.

8. Create a new relation with the same attributes as the *jbitems* relation by using the CREATE TABLE command where you define every attribute explicitly (i.e., not as a copy of another table). Then, populate this new relation with all items that cost less than the average price for all items. Remember to define the primary key and foreign keys in your table!
9. Create a view that contains the items that cost less than the average price for items.
10. *What is the difference between a table and a view?* One is static and the other is dynamic. Which is which and what do we mean by static respectively dynamic?
11. Create a view that calculates the total cost of each debit, by considering price and quantity of each bought item. (To be used for charging customer accounts). The view should contain the sale identifier (debit) and the total cost. In the query that defines the view, capture the join condition in the WHERE clause (i.e., do *not* capture the join in the FROM clause by using keywords *inner join, right join* or *left join*).
12. Do the same as in the previous point, but now capture the join conditions in the FROM clause by using only *left, right* or *inner* joins. Hence, the WHERE clause must not contain any join condition in this case. Motivate why you use type of join you do (left, right or inner), and why this is the correct one (in contrast to the other types of joins).
13. Oh no! An earthquake!
    a) Remove all suppliers in Los Angeles from the *jbsupplier* table. This will not work right away. Instead, you will receive an error with error code 23000 which you will have to solve by deleting some other related tuples. However, do not delete more tuples from other tables than necessary, and do not change the structure of the tables (i.e., do not remove foreign keys). Also, you are only allowed to use “Los Angeles” as a constant in your queries, not “199” or “900”.
    b) Explain what you did and why.
14. An employee has tried to find out which suppliers have delivered items that have been sold. To this end, the employee has created a view and a query that lists the number of items sold from a supplier.

```
mysql> CREATE VIEW jbsale_supply(supplier, item, quantity) AS
 -> SELECT jbsupplier.name, jbitem.name, jbsale.quantity
 -> FROM jbsupplier, jbitem, jbsale
 -> WHERE jbsupplier.id = jbitem.supplier
 -> AND jbsale.item = jbitem.id;
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> SELECT supplier, sum(quantity) AS sum FROM jbsale_supply
 -> GROUP BY supplier;
+-------------+--------------+
| supplier    | sum(quantity) |
+-------------+--------------+
| Cannon      |            6 |
| Levi-Strauss |           1 |
| Playskool   |            2 |
| White Stag  |            4 |
| Whitman's   |            2 |
+-------------+--------------+
5 rows in set (0.00 sec)
```

Now, the employee also wants to include the suppliers that have delivered some items, although for whom no items have been sold so far. In other words, he wants to list all suppliers that have supplied any item, as well as the number of these items that have been sold. Help him! Drop and redefine the *jbsale_supply* view to also consider suppliers that have delivered items that have never been sold.

Hint: Notice that the above definition of *jbsale_supply* uses an (implicit) inner join that removes suppliers that have not had any of their delivered items sold.

Handing In

Hand in an executable SQL file that contains each of the SQL statements and, in comments, the output produced by executing the statement as well as the text answers for points 10, 12, and 13b. Additionally, the file should start with a comment that contains your names (yours and your lab partner’s) as well as your LiU IDs. Hence, this file should look something like the following (see next page).

```
/*Lab 3, Anders Andersson (andan123) and Björn Björnsson (bjobj456)*/

SOURCE company_schema.sql;
SOURCE company_data.sql;

/*Question 1: Print a message that says "hello world"*/
SELECT 'hello world!' AS 'message';

/*
+--------------+
|    message   |
+--------------+
| hello world! |
+--------------+
1 row in set (0.00 sec)
*/

/* Question 2: List all tables */
SHOW TABLES;

/*
+--------------------+
| Tables_in_andan123 |
+--------------------+
| jbcity             |
| jbdebit            |
| jbdept             |
| jbemployee         |
| jbitem             |
| jbparts            |
| jbsale             |
| jbstore            |
| jbsupplier         |
| jbsupply           |
+-------------------+
10 rows in set (0.00 sec)
*/

/* Question 3: What does the acronym SQL stand for? */
/* Answer: SQL stands for Structured Query Language. */
```