732A54 - Database Lab 1

SQL Queries and Views

Ashraf Sarhan

2017-02-20

1. List all employees, i.e. all tuples in the jbemployee relation.

```
SELECT *
FROM jbemployee;
```

```
+------+--------------------+--------+---------+-----------+-----------+
| id   | name               | salary | manager | birthyear | startyear |
+------+--------------------+--------+---------+-----------+-----------+
|   10 | Ross, Stanley      |  15908 |     199 |      1927 |      1945 |
|   11 | Ross, Stuart       |  12067 |    NULL |      1931 |      1932 |
|   13 | Edwards, Peter     |   9000 |     199 |      1928 |      1958 |
|   26 | Thompson, Bob      |  13000 |     199 |      1930 |      1970 |
|   32 | Smythe, Carol      |   9050 |     199 |      1929 |      1967 |
|   33 | Hayes, Evelyn      |  10100 |     199 |      1931 |      1963 |
|   35 | Evans, Michael     |   5000 |      32 |      1952 |      1974 |
|   37 | Raveen, Lemont     |  11985 |      26 |      1950 |      1974 |
|   55 | James, Mary        |  12000 |     199 |      1920 |      1969 |
|   98 | Williams, Judy     |   9000 |     199 |      1935 |      1969 |
|  129 | Thomas, Tom        |  10000 |     199 |      1941 |      1962 |
|  157 | Jones, Tim         |  12000 |     199 |      1940 |      1960 |
|  199 | Bullock, J.D.      |  27000 |    NULL |      1920 |      1920 |
|  215 | Collins, Joanne    |   7000 |      10 |      1950 |      1971 |
|  430 | Brunet, Paul C.    |  17674 |     129 |      1938 |      1959 |
|  843 | Schmidt, Herman    |  11204 |      26 |      1936 |      1956 |
|  994 | Iwano, Masahiro    |  15641 |     129 |      1944 |      1970 |
| 1110 | Smith, Paul        |   6000 |      33 |      1952 |      1973 |
| 1330 | Onstad, Richard    |   8779 |      13 |      1952 |      1971 |
| 1523 | Zugnoni, Arthur A. |  19868 |     129 |      1928 |      1949 |
| 1639 | Choy, Wanda        |  11160 |      55 |      1947 |      1970 |
| 2398 | Wallace, Maggie J. |   7880 |      26 |      1940 |      1959 |
| 4901 | Bailey, Chas M.    |   8377 |      32 |      1956 |      1975 |
| 5119 | Bono, Sonny        |  13621 |      55 |      1939 |      1963 |
| 5219 | Schwarz, Jason B.  |  13374 |      33 |      1944 |      1959 |
+------+--------------------+--------+---------+-----------+-----------+
25 rows in set (0,00 sec)
```

2. List the name of all departments in alphabetical order. Note: by "name" we mean the name attribute for all tuples in the *jbdept* relation.

```
SELECT jbdept.name AS dept_name
FROM jbdept
ORDER BY dept_name;
```

```
+-------------------+
| dept_name         |
+-------------------+
| Bargain           |
| Book              |
| Candy             |
| Children's        |
| Children's        |
| Furniture         |
| Giftwrap          |
| Jewelry           |
| Junior Miss       |
| Junior's          |
| Linens            |
| Major Appliances  |
| Men's             |
| Sportswear        |
| Stationary        |
| Toys              |
| Women's           |
| Women's           |
| Women's           |
+-------------------+
19 rows in set (0,00 sec)
```

3. What parts are not in store, i.e. qoh = 0? (qoh = Quantity On Hand)

```
SELECT *
FROM jbparts
WHERE jbparts.qoh=0;
```

```
+----+-------------------+-------+--------+------+
| id | name              | color | weight | qoh  |
+----+-------------------+-------+--------+------+
| 11 | card reader       | gray  |    327 |    0 |
| 12 | card punch        | gray  |    427 |    0 |
| 13 | paper tape reader | black |    107 |    0 |
| 14 | paper tape punch  | black |    147 |    0 |
+----+-------------------+-------+--------+------+
4 rows in set (0,00 sec)
```

4. Which employees have a salary between 9000 (included) and 10000 (included)?

```
SELECT * FROM jbemployee
WHERE jbemployee.salary BETWEEN 9000 AND 10000;
```

```
+-----+----------------+--------+---------+-----------+-----------+
| id  | name           | salary | manager | birthyear | startyear |
+-----+----------------+--------+---------+-----------+-----------+
|  13 | Edwards, Peter |   9000 |     199 |      1928 |      1958 |
|  32 | Smythe, Carol  |   9050 |     199 |      1929 |      1967 |
|  98 | Williams, Judy |   9000 |     199 |      1935 |      1969 |
| 129 | Thomas, Tom    |  10000 |     199 |      1941 |      1962 |
+-----+----------------+--------+---------+-----------+-----------+
4 rows in set (0,00 sec)
```

5. What was the age of each employee when they started working (startyear)?

```
SELECT emp.id, emp.name, (emp.startyear-emp.birthyear) AS age
FROM jbemployee AS emp;
```

```
+------+--------------------+------+
| id   | name               | age  |
+------+--------------------+------+
|   10 | Ross, Stanley      |   18 |
|   11 | Ross, Stuart       |    1 |
|   13 | Edwards, Peter     |   30 |
|   26 | Thompson, Bob      |   40 |
|   32 | Smythe, Carol      |   38 |
|   33 | Hayes, Evelyn      |   32 |
|   35 | Evans, Michael     |   22 |
|   37 | Raveen, Lemont     |   24 |
|   55 | James, Mary        |   49 |
|   98 | Williams, Judy     |   34 |
|  129 | Thomas, Tom        |   21 |
|  157 | Jones, Tim         |   20 |
|  199 | Bullock, J.D.      |    0 |
|  215 | Collins, Joanne    |   21 |
|  430 | Brunet, Paul C.    |   21 |
|  843 | Schmidt, Herman    |   20 |
|  994 | Iwano, Masahiro    |   26 |
| 1110 | Smith, Paul        |   21 |
| 1330 | Onstad, Richard    |   19 |
| 1523 | Zugnoni, Arthur A. |   21 |
| 1639 | Choy, Wanda        |   23 |
| 2398 | Wallace, Maggie J. |   19 |
| 4901 | Bailey, Chas M.    |   19 |
| 5119 | Bono, Sonny        |   24 |
| 5219 | Schwarz, Jason B.  |   15 |
+------+--------------------+------+
25 rows in set (0,00 sec)
```

6. Which employees have a last name ending with “son”?

```
SELECT emp.id, SUBSTRING_INDEX(emp.name , ',', 1) AS firstname,
SUBSTRING_INDEX(emp.name , ',', -1) AS lastname, emp.salary,
emp.manager, emp.birthyear, emp.startyear
FROM jb.jbemployee AS emp
WHERE SUBSTRING_INDEX(emp.name , ',', -1) LIKE '%son';
```

```
+------+-----------+----------+--------+---------+-----------+-----------+
| id   | firstname | lastname | salary | manager | birthyear | startyear |
+------+-----------+----------+--------+---------+-----------+-----------+
| 5219 | Schwarz   |  Jason   |  13374 |      33 |      1944 |      1959 |
+------+-----------+----------+--------+---------+-----------+-----------+
1 row in set (0,00 sec)
```

7. Which items (note items, not parts) have been delivered by a supplier called Fisher-Price? Formulate this query using a subquery in the where-clause.

```
SELECT *
FROM jbitem AS itm
WHERE itm.supplier = (SELECT sup.id
                      FROM jbsupplier as sup
                      WHERE sup.name = 'Fisher-Price');
```

```
+-----+-----------------+------+-------+------+----------+
| id  | name            | dept | price | qoh  | supplier |
+-----+-----------------+------+-------+------+----------+
|  43 | Maze            |   49 |   325 |  200 |       89 |
| 107 | The 'Feel' Book |   35 |   225 |  225 |       89 |
| 119 | Squeeze Ball    |   49 |   250 |  400 |       89 |
+-----+-----------------+------+-------+------+----------+
3 rows in set (0,00 sec)
```

8. Formulate the same query as above, but without a subquery.

```
SELECT itm.id, itm.name, itm.dept, itm.price, itm.qoh, itm.supplier
FROM jbitem AS itm, jbsupplier AS sup
WHERE itm.supplier = sup.id AND sup.name = 'Fisher-Price';
```

```
+-----+-----------------+------+-------+------+----------+
| id  | name            | dept | price | qoh  | supplier |
+-----+-----------------+------+-------+------+----------+
|  43 | Maze            |   49 |   325 |  200 |       89 |
| 107 | The 'Feel' Book |   35 |   225 |  225 |       89 |
| 119 | Squeeze Ball    |   49 |   250 |  400 |       89 |
+-----+-----------------+------+-------+------+----------+
3 rows in set (0,16 sec)
```

9. Show all cities that have suppliers located in them. Formulate this query using a subquery in the where-clause.

```
SELECT * FROM jbcity AS cty
WHERE cty.id IN (SELECT sup.city
                 FROM jbsupplier AS sup);
```

```
+-----+----------------+-------+
| id  | name           | state |
+-----+----------------+-------+
|  10 | Amherst        | Mass  |
|  21 | Boston         | Mass  |
| 100 | New York       | NY    |
| 106 | White Plains   | Neb   |
| 118 | Hickville      | Okla  |
| 303 | Atlanta        | Ga    |
| 537 | Madison        | Wisc  |
| 609 | Paxton         | Ill   |
| 752 | Dallas         | Tex   |
| 802 | Denver         | Colo  |
| 841 | Salt Lake City | Utah  |
| 900 | Los Angeles    | Calif |
| 921 | San Diego      | Calif |
| 941 | San Francisco  | Calif |
| 981 | Seattle        | Wash  |
+-----+----------------+-------+
15 rows in set (0,00 sec)
```

10. What is the name and color of the parts that are heavier than a card reader? Formulate this query using a subquery in the where-clause. (The SQL query must not contain the weight as a constant.)

```
SELECT prts.name, prts.color
FROM jbparts AS prts
WHERE prts.weight > (SELECT prts.weight
                     FROM jbparts AS prts
                     WHERE prts.name = 'card reader');
```

```
+--------------+--------+
| name         | color  |
+--------------+--------+
| disk drive   | black  |
| tape drive   | black  |
| line printer | yellow |
| card punch   | gray   |
+--------------+--------+
4 rows in set (0,00 sec)
```

11. Formulate the same query as above, but without a subquery. (The query must not contain the weight as a constant.)

```
SELECT prts.name, prts.color
FROM jbparts AS prts, jbparts AS prts2
WHERE prts2.name = 'card reader' AND prts.weight > prts2.weight;
```

```
+--------------+--------+
| name         | color  |
+--------------+--------+
| disk drive   | black  |
| tape drive   | black  |
| line printer | yellow |
| card punch   | gray   |
+--------------+--------+
4 rows in set (0,00 sec)
```

12. What is the average weight of black parts?

```
SELECT avg(prts.weight) AS avg_weight
FROM jbparts AS prts WHERE prts.color = 'black';
```

```
+------------+
| avg_weight |
+------------+
|   347.2500 |
+------------+
1 row in set (0,00 sec)
```

13. What is the total weight of all parts that each supplier in Massachusetts ("Mass") has delivered? Retrieve the name and the total weight for each of these suppliers. Do not forget to take the quantity of delivered parts into account. Note that one row should be returned for each supplier.

```
SELECT sup.name, SUM(prts.weight*sp.quan) AS total_weight
FROM jb.jbparts AS prts, jbsupplier AS sup, jbsupply AS sp, jbcity AS cty
WHERE prts.id = sp.part AND sup.id = sp.supplier
AND sup.city = cty.id AND cty.state = 'Mass'
GROUP BY sup.id;
```

```
+--------------+--------------+
| name         | total_weight |
+--------------+--------------+
| Fisher-Price |      1135000 |
| DEC          |         3120 |
+--------------+--------------+
2 rows in set (0,00 sec)
```

14. Create a new relation (a table), with the same attributes as the table items using the CREATE TABLE syntax where you define every attribute explicitly (i.e. not as a copy of another table). Then fill the table with all items that cost less than the average price for items. Remember to define primary and foreign keys in your table!

```
CREATE TABLE jbitem_replica
    (
    `id` INT(11),
    `name` VARCHAR(20),
    `price` INT(11),
    `qoh` INT(10) UNSIGNED,
    `dept` INT(11),
    `supplier` INT(11),
    PRIMARY KEY (`id`),
    FOREIGN KEY (`dept`) REFERENCES `jbdept`(`id`),
    FOREIGN KEY (`supplier`) REFERENCES `jbsupplier`(`id`)
    );
```

```
Query OK, 0 rows affected (0,47 sec)
```

```
INSERT INTO jb.jbitem_replica(`id`, `name`, `price`, `qoh`, `dept`, `supplier`)
    SELECT itm.id, itm.name, itm.price, itm.qoh, itm.dept, itm.supplier
    FROM jb.jbitem AS itm
    WHERE (itm.price) < (SELECT AVG(jb.jbitem.price) FROM jb.jbitem);
```

```
Query OK, 14 rows affected (0,14 sec)
Records: 14  Duplicates: 0  Warnings: 0
```

```
SELECT *
FROM jb.jbitem_replica;
```

```
+-----+-----------------+-------+------+------+----------+
| id  | name            | price | qoh  | dept | supplier |
+-----+-----------------+-------+------+------+----------+
|  11 | Wash Cloth      |    75 |  575 |    1 |      213 |
|  19 | Bellbottoms     |   450 |  600 |   43 |       33 |
|  21 | ABC Blocks      |   198 |  405 |    1 |      125 |
|  23 | 1 lb Box        |   215 |  100 |   10 |       42 |
|  25 | 2 lb Box, Mix   |   450 |   75 |   10 |       42 |
|  26 | Earrings        |  1000 |   20 |   14 |      199 |
|  43 | Maze            |   325 |  200 |   49 |       89 |
| 106 | Clock Book      |   198 |  150 |   49 |      125 |
| 107 | The 'Feel' Book |   225 |  225 |   35 |       89 |
| 118 | Towels, Bath    |   250 | 1000 |   26 |      213 |
| 119 | Squeeze Ball    |   250 |  400 |   49 |       89 |
| 120 | Twin Sheet      |   800 |  750 |   26 |      213 |
| 165 | Jean            |   825 |  500 |   65 |       33 |
| 258 | Shirt           |   650 | 1200 |   58 |       33 |
+-----+-----------------+-------+------+------+----------+
14 rows in set (0,00 sec)
```