# Big data

## Assignment 1

- Martynas Lukosevicius marlu207
- Alejo Perez Gomez alepe026

1) List all employees, i.e. all tuples in the jbemployee relation.
SQL: *select * from jbemployee*

| id | name | salary | manager | birthyear | startyear |
|---|---|---|---|---|---|
| 10 | Ross, Stanley | 15908 | 199 | 1927 | 1945 |
| 11 | Ross, Stuart | 12067 | 0 | 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 | 0 | 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 |

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.

SQL: `select name from jbdept order by name`

| 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 |

3) What parts are not in store, i.e. qoh = 0? (qoh = Quantity On Hand)
SQL: `select * from jbparts where 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) Which employees have a salary between 9000 (included) and 10000 (included)?
SQL: `select name from jbemployee where salary between 9000 and 10000;`

| name |
|---|
| Edwards, Peter |
| Smythe, Carol |
| Williams, Judy |

| |
|---|
| Thomas, Tom |

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

SQL: `select name, startyear-birthyear as age from jbemployee;`

| name | age |
|---|---|
| Ross, Stanley | 18 |
| Ross, Stuart | 1 |
| Edwards, Peter | 30 |
| Thompson, Bob | 40 |
| Smythe, Carol | 38 |
| Hayes, Evelyn | 32 |
| Evans, Michael | 22 |
| Raveen, Lemont | 24 |
| James, Mary | 49 |
| Williams, Judy | 34 |
| Thomas, Tom | 21 |
| Jones, Tim | 20 |
| Bullock, J.D. | 0 |
| Collins, Joanne | 21 |
| Brunet, Paul C. | 21 |
| Schmidt, Herman | 20 |
| Iwano, Masahiro | 26 |
| Smith, Paul | 21 |
| Onstad, Richard | 19 |
| Zugnoni, Arthur A. | 21 |
| Choy, Wanda | 23 |
| Wallace, Maggie J. | 19 |
| Bailey, Chas M. | 19 |
| Bono, Sonny | 24 |
| Schwarz, Jason B. | 15 |

6) Which employees have a last name ending with “son”?
SQL: `select name from jbemployee WHERE NAME LIKE '%son,%';`

| name |
|---|
| Thompson, Bob |

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.

SQL: *select name from jbitem where supplier in (select id from jbsupplier where name = 'Fisher-Price')*

| name |
|---|
| Maze |
| The 'Feel' Book |
| Squeeze Ball |

8) Formulate the same query as above, but without a subquery.
SQL: *select it.name from jbitem it, jbsupplier sup where it.supplier = sup.id and sup.name like 'Fish%'*

| name |
|---|
| Maze |
| The 'Feel' Book |
| Squeeze Ball |

9) Show all cities that have suppliers located in them. Formulate this query using a

subquery in the where-clause.

SQL: *select name from jbcity where id in (select city from jbsupplier);*

| name |
|---|
| Amherst |
| Boston |
| New York |
| White Plains |
| Hickville |
| Atlanta |
| Madison |
| Paxton |
| Dallas |
| Denver |
| Salt Lake City |
| Los Angeles |
| San Diego |

|  |
|---|
| San Francisco |
| Seattle |

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.)

SQL: *select name, color from jbparts where weight >= (select weight from jbparts where name = 'card reader');*

| name | color |
|---|---|
| disk drive | black |
| tape drive | black |
| line printer | yellow |
| card reader | gray |
| card punch | gray |

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

contain the weight as a constant.)

```
SQL: SELECT
    t1.name, t1.color
FROM
    jbparts t1
     JOIN jbparts t2
    ON t1.weight >= t2.weight
    AND t2.name  = 'card reader';
```

|  |  |
|---|---|
| 'disk drive' | 'black' |
| 'tape drive' | 'black' |
| 'line printer' | 'yellow' |
| 'card reader' | 'gray' |
| 'card punch' | 'gray' |

12) What is the average weight of black parts?

SQL: `select avg(weight) from jbparts where color = 'black';`

347,2500

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.

```
SQL: select sup.name, sum(suply.quan * parts.weight) total_weight
from jbsupply suply, jbsupplier sup, jbcity city, jbparts parts
where

suply.supplier = sup.id

and sup.city = city.id

and suply.part = parts.id

and city.state = 'Mass'

group by sup.name;
```

| name | total_weight |
|---|---:|
| DEC | 3120 |
| Fisher-Price | 1135000 |

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!

```
SQL: CREATE TABLE new_item (

    id INT,

    name VARCHAR(20),

    dept INT NOT NULL,

    price INT,
    qoh INT UNSIGNED ,

    supplier INT NOT NULL,

    CONSTRAINT pk_new_item PRIMARY KEY(id),

    CONSTRAINT fk_new_item_dept FOREIGN KEY (dept) REFERENCES jbdept(id),

    CONSTRAINT fk_new_item_supplier FOREIGN KEY (supplier) REFERENCES jbsupplier(id));


INSERT INTO new_item

select * from jbitem where

price > (select avg(price) from jbitem);
```

| id | name | dept | price | qoh | supplier |
|---|---|---|---|---|---|
| 52 | Jacket | 60 | 3295 | 300 | 15 |
| 101 | Slacks | 63 | 1600 | 325 | 15 |
| 115 | Gold Ring | 14 | 4995 | 10 | 199 |
| 121 | Queen Sheet | 26 | 1375 | 600 | 213 |
| 127 | Ski Jumpsuit | 65 | 4350 | 125 | 15 |
| 301 | Boy's Jean Suit | 43 | 1250 | 500 | 33 |