lab1 BDA

Zijie Feng & Phillp Holscher

2019-03-30

0) Setup the environment for database

```
db <-dbConnect(RMySQL::MySQL(),host="localhost",
               dbname="jb",user="root",password="12345")

# db <-dbConnect(RMySQL::MySQL(),host="mariadb.edu.liu.se",
#                dbname="zijfe244",user="zijfe244",password="KYBAwFrp")
```

```
show tables;
```

Table 1: 10 records

Tables_in_jb
jbcity
jbdebit
jbdept
jbemployee
jbitem
jbparts
jbsale
jbstore
jbsupplier
jbsupply

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

```
select * from jbemployee;
```

Table 2: 25 records

id	name	salary	manager	birthyear	startyear
10	Ross, Stanley	15908	199	1927	1945
11	Ross, Stuart	12067	NA	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	NA	1920	1920
215	Collins, Joanne	7000	10	1950	1971
430	Brunet, Paul C.	17674	129	1938	1959

id	name	salary	manager	birthyear	startyear
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.

```
select name from jbdept order by name;
```

Table 3: 19 records

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)

```
select name from jbparts where qoh=0;
```

Table 4: 4 records

name
card reader
card punch
paper tape reader

name
paper tape punch

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

```
select name from jbemployee where salary between 9000 and 10000;
```

Table 5: 4 records

name
Edwards, Peter
Smythe, Carol
Williams, Judy
Thomas, Tom

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

```
select name, (startyear-birthyear) as age_started from jbemployee;
```

Table 6: 25 records

name	age_started
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"?

```
select id,name from jbemployee where name like '%son,%';
```

Table 7: 1 records

id	name
26	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.

```
select name from jbitem where supplier =
(select id from jbsupplier where name='Fisher-Price');
```

Table 8: 3 records

name
Maze
The 'Feel' Book
Squeeze Ball

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

```
select I.name, S.name from jbitem I, jbsupplier S
where S.name='Fisher-Price' and S.id=I.supplier;
```

Table 9: 3 records

name	name
Maze	Fisher-Price
The 'Feel' Book	Fisher-Price
Squeeze Ball	Fisher-Price

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

```
select DISTINCT(name) from jbcity
where id in (select city from jbsupplier);
```

Table 10: 15 records

name
Amherst
Boston
New York
White Plains
Hickville
Atlanta

name
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.)

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

Table 11: 4 records

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

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

```
select p.name,p.color from jbparts p, jbparts b
where p.weight>b.weight and b.name='card reader';
```

Table 12: 4 records

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

12) What is the average weight of black parts?

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

Table 13: 1 records

avg(weight)
347.25

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 s.name, sum(p.weight*d.quan)  from jbsupplier s,jbparts p,jbsupply d, jbcity c
where p.id=d.part  and d.supplier=s.id  and s.city=c.id  and c.state='mass' Group by s.name;
```

Table 14: 2 records

name	sum(p.weight*d.quan)
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!

```
# create an empty table with headers
create table jbinfo (id integer primary key, name varchar(25),
dept integer ,price integer , qoh integer , supplier integer );
```

```
insert into jbinfo (id, name, dept, price, qoh, supplier)
select id, name, dept, price, qoh, supplier from jbitem
where price<all (select avg(price) from jbitem);
```

```
select * from jbinfo;
```

Table 15: 14 records

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

```
drop table jbinfo;
```

15) Create a view that contains the items that cost less than the average price for items.

```
# or `create view item_view as select * from jbinfo`

create view item_view as
select * from jbitem
where price < (select avg(price) from jbitem);
```

```
select * from item_view;
```

Table 16: 14 records

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

```
drop view item_view;
```

16) 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?

Table is static, and view is dynamic and a vitual table derived from other tables or views. The static here means that some data are stored in such table actually, not just logistic `select` statement.

17) Create a view, using only the implicit join notation, i.e. only use where statements but no inner join, right join or left join statements, 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 total cost.

```
create view debit_view as
select D.id, sum(I.price*S.quantity) as total_cost
from jbdebit D, jbitem I, jbsale S
where I.id = S.item and D.id = S.debit group by S.debit;
```

```
select * from debit_view
```

Table 17: 6 records

id	total_cost
100581	2050
100582	1000
100586	13446
100592	650
100593	430
100594	3295

18) Do the same as in (17), using only the explicit join notation, i.e. using only left, right or inner joins but no where statement. Motivate why you use the join you do (left, right or inner), and why this is the correct one (unlike the others).

```
select D.id, sum(I.price*S.quantity) as total_cost
from (jbsale S left join jbdebit D on S.debit = D.id
left join jbitem I on S.item = I.id) group by S.debit
```

Table 18: 6 records

id	total_cost
100581	2050
100582	1000
100586	13446
100592	650
100593	430
100594	3295

```
drop view debit_view
```

19) Oh no! An earthquake!

- a) Remove all suppliers in Los Angeles from the table jbsupplier. This will not work right away (you will receive 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, remember that you are only allowed to use "Los Angeles" as a constant in your queries, not "199" or "900".

```
# show keys from jbsupplier;
select TABLE_NAME,COLUMN_NAME,CONSTRAINT_NAME,REFERENCED_TABLE_NAME,REFERENCED_COLUMN_NAME
from information_schema.KEY_COLUMN_USAGE where REFERENCED_TABLE_NAME='jbcity';
```

Table 19: 2 records

TABLE_NAME	COLUMN_NAME	CONSTRAINT_NAME	REFERENCED_TABLE_NAME	REFERENCED_
jbstore	city	fk_store_city	jbcity	id
jbsupplier	city	fk_supplier_city	jbcity	id

```
delete jbsale
from jbsale
right join jbitem on jbitem.id=jbsale.item
left join jbsupplier on jbsupplier.id=jbitem.supplier
left join jbcity on jbcity.id=jbsupplier.city
where jbcity.name='Los Angeles';
```

```
delete jbitem
from jbitem
left join jbsupplier on jbsupplier.id=jbitem.supplier
left join jbcity on jbcity.id=jbsupplier.city
where jbcity.name='Los Angeles';
```

```
delete jbsupplier
from jbsupplier
left join jbcity on jbcity.id=jbsupplier.city
where jbcity.name='Los Angeles';
```

```
select * from jbsupplier;
```

Table 20: 15 records

id	name	city
5	Amdahl	921
15	White Stag	106
20	Wormley	118
33	Levi-Strauss	941
42	Whitman's	802
62	Data General	303
67	Edger	841
89	Fisher-Price	21
122	White Paper	981
125	Playskool	752
213	Cannon	303
241	IBM	100
440	Spooley	609
475	DEC	10
999	A E Neumann	537

- b) Explain what you did and why.

The *id* in jbitem is the foreign key of jbsale (item), and the *id* in jbsupplier is the foreign key of jbitem (supplier). If we want to delete the tuples in jbsupplier, we do have to remove all the correlated tuples firstly.

20) An employee has tried to find out which suppliers that have delivered items that have been sold. He has created a view and a query that shows the number of items sold from a supplier.

```
CREATE VIEW jbsale_supply(supplier, item, quantity) AS
SELECT jbsupplier.name, jbitem.name, jbsale.quantity
FROM jbsale
right join jbitem on jbitem.id=jbsale.item
inner join jbsupplier on jbsupplier.id = jbitem.supplier
```

```
SELECT supplier, sum(quantity) AS sum FROM jbsale_supply
GROUP BY supplier;
```

Table 21: 6 records

supplier	sum
Cannon	6
Fisher-Price	NA
Levi-Strauss	1
Playskool	2
White Stag	4
Whitman's	2

```
drop view jbsale_supply
```