Assignment

Thomas D. Vinther & Jens Kristian R. Nielsen

24. februar 2019

1

1.Find the names of beverages that come in ‘maxi’ size.
We choose the name of the beverage from BEVERAGE, where size = ’maxi’, pretty straight forward.

```
SELECT DISTINCT name
FROM BEVERAGE
WHERE size = ’maxi’
```

2. Find the names of beverages that come in ‘maxi’ or ‘medium’ size.
We simply add an OR statement to the previous query.

```
SELECT DISTINCT name
FROM BEVERAGE
WHERE size = ’maxi’ OR size = ’medium’;
```

3. Find the names of beverages that come in both a ‘maxi’ and a ‘medium’ size.
We "create"two identical tables, and compare the entires where the names are identical and where in one table the size is ’maxi’ and in the other is ’medium’, we return the names of these entries.

```
SELECT DISTINCT b1.name
FROM BEVERAGE b1, BEVERAGE b2
WHERE b1.name = b2.name AND b1.size = ’maxi’ AND b2.size = ’medium’ ;
```

4. Find the names and phone numbers of the stores in “Randers” or “Horsens” that sell a ‘maxi’ beverage named “cordusio” for no more than DKK 45.
Straight forward once again, we simply use all three tables, check the entries where the store name is equal to the store name in SELLS, the codes are equal from SELLS and BEVERAGE, the name of the beverage is ’Cordusio’, the size is maxi and the area is either Rander or Horsens, from STORE.

```
SELECT s.name, s.phone
FROM STORE s, BEVERAGE b, SELLS se
WHERE s.name = se.store_name AND se.code = b.code AND se.price <= 45 AND
b.name = ’Cordusio’ AND b.size = ’maxi’ AND (s.area = ’Randers’ OR s.area = ’Horsens’);
```

5. Find the code(s), name(s), and name(s) of store(s) selling the least expensive beverage(s). We find the entries where the code is equal from BEVERAGE to SELLS, the store name from SELLS and STORE, we then choose the SELLS price to be the minimum of all the prices, and thereby get all entries with the cheapest beverage.

```
SELECT b.code, b.name , s.name
FROM STORE s, BEVERAGE b, SELLS se
WHERE se.code = b.code AND se.store_name = s.name AND se.price =

(SELECT MIN(price)
FROM sells);
```

6. For each store, give its name and the code(s) of the least expensive beverage(s) it sells. [hint: pay attention to the fact that one store's lowest price may be another store's ordinary price]
We repeat the procedure from 5, but in the minimum we add minimum for each store and choose the distinct entries so as not to get duplicates.

```
SELECT DISTINCT s.name, b.code
FROM STORE s, BEVERAGE b, SELLS se
WHERE se.code = b.code AND s.name = se.store_name AND se.price =

(SELECT MIN(price)
FROM SELLS sel
WHERE sel.store_name = s.name);
```

7. For each store, give its name and the price of the least expensive beverage(s) it sells. Answer once again, but now (i) include the name(s) of such beverage(s) and (ii) do not use aggregation operations like MIN, GROUP BY, ORDER BY, etc.
We choose distinct name from SELLS so as to only get each store once. We then choose the entries with the name of the Stores where the price is less to or equal to all the prices for that particular store.

```
SELECT DISTINCT se.store_name, se.price
FROM STORE s, SELLS se
WHERE s.name = se.store_name AND se.price <=  ALL (

SELECT price
FROM SELLS
WHERE store_name = se.store_name);
```

7(II)
We repeat the above procedure but add the name of the Beverage by comparing the code from BEVERAGE and SELLS.

```
SELECT DISTINCT se.store_name, se.price, b.name
FROM STORE s, SELLS se, BEVERAGE b
WHERE b.code = se.code AND s.name = se.store_name AND se.price <=  ALL (

SELECT price
FROM SELLS
WHERE store_name = se.store_name);
```

8. For each beverage, give its name, size, and its highest price across all stores. Answer once again, but now (i) include the name(s) of store(s) selling that beverage at the highest price, and (ii) do not use aggregation operations like MAX, GROUP BY, ORDER BY, etc.
We more or less repeat the procedure from 7. This time we just choose from BEVERAGE and SELLS as this is adequate as we simply compare the entires with equal code, and where the price is highest for each code.

```
SELECT DISTINCT b.name, b.size, se.price
FROM SELLS se, BEVERAGE b
WHERE b.code = se.code AND  se.price >=  ALL (

SELECT price
FROM SELLS
WHERE code = b.code);
```

8(II)
We repeat the procedure but add the name of the store where it is sold at the price, which is the highest.

```
SELECT DISTINCT b.name, b.size, se.price, se.store_name
FROM SELLS se, BEVERAGE b
WHERE b.code = se.code AND se.price >=  ALL (

SELECT price
FROM SELLS
WHERE code = b.code);
```

9. Find the names of the stores that sell all beverages in the database; do not use COUNT. Here we struggled a little bit, because it would probably be quite easy to just use a count function on BEVERAGE, and then as well on SELLS comparing distinct store names. But luckily we got a great hint from the lectures, using a form of double negation.

```
SELECT s.name
FROM STORE s
WHERE NOT EXISTS(
SELECT *
FROM BEVERAGE b
WHERE NOT EXISTS(SELECT *
FROM SELLS se
WHERE se.store_name = s.name AND se.code = b.code));
```

10.Find the codes of beverages that are sold in all stores in the database; do not use COUNT. We rinse and repeat but choose the code instead.

```
SELECT b.code
FROM BEVERAGE b
WHERE NOT EXISTS(
SELECT *
FROM STORE s
WHERE NOT EXISTS(SELECT *
FROM SELLS se
WHERE se.store_name = s.name AND se.code = b.code));
```