CS2102 Tutorial 4

1. a) Query A is valid

allowed to SELECT 1 in query

correlated subquery also valid

for tuples (2,20) and (3,30) in bar,

there exists f.a = b.a and f.f >100

returns

a	b
2	20
3	30

b) Query B is invalid

table alias foo f can only be used in the query it was declared or subqueries.

f is referenced outside => invalid

c) Query C is valid

a in a = b.a references f.a as attributes are resolved

from inner to outer queries => same as Query A

a	b
2	20
3	30

d) Query D is valid

a = a => f.a = f.a => all tuples valid

a	b
1	10
2	20
3	30
4	40

e) Query E is valid

b > 20 resolves to b.b after definition found in outer query.

only (3,30) satisfies WHERE clause

a | b

2.

a) natural join of Likes and Sells over pizza attribute and additional contraint S.rname = 'Corleone Corner'

```
SELECT DISTINCT cname
FROM likes L , sells S
WHERE S.pizza = L.pizza
AND S.rname = 'Corleone Corner';
```

b)

```
(SELECT cname
 FROM customers )

EXCEPT

(SELECT DISTINCT C.cname
FROM customers C, likes L, sells S
WHERE S.pizza = L.pizza
AND S.rname = 'Corleone Corner'
AND C.cname = L.cname);
```

set difference of all customer name with customer who likes pizzas sold by Corleone Corner

left outer join ?

c)

```
SELECT DISTINCT S1.rname
FROM sells S1, sells S2
WHERE S1.rname <> 'Corleone Corner'
AND S2.rname = 'Corleone Corner'
AND S1.price > S2.price;
```

compare all sell x sell pairs & filter fitting condition

d) Highest price pizza = all pizza - pizza that is not max

```
(SELECT rname, pizza, price
 FROM sells )

EXCEPT

(SELECT S1.rname, S1.pizza, S1.price
 FROM sells S1, sells S2
 WHERE S1.rname = S2.rname   AND S1.price < S2.price   AND S2.price
                                                        IS NOT NULL);
```

not max

3. a)

```
(SELECT pizza
 FROM Likes
 WHERE cname = 'Alice')

EXCEPT

(SELECT pizza
 FROM Likes
 WHERE cname = 'Bob');
```

b) exclude any pizzas not sold by restaurants => only select pizza from sells.

set difference with

pizzas sold in same area by 2 different restaurants

```
(SELECT DISTINCT pizza
 FROM Sells)

EXCEPT

(SELECT DISTINCT S1.pizza
 FROM (Sells Natural Join Restaurants) S1, (Sells Natural Join Restaurants) S2
 WHERE S1.pizza = S2.pizza
 AND S1.area = S2.area
 AND S1.rname <> S2.rname);
```

OR using NOT EXISTS (see .txt)

c) price <= ALL (pizza in same area)

d) price >= ALL (pizza in same area)

natural join min and max subqueries

4.

$Q_1 = (\text{Sells} \bowtie \sigma_{area='East'}(\text{Restaurants}))$

Let Sells = S
Restaurants = R
$\sigma_{area='East'} = \sigma_E$

$Q_2 = \sigma_{area='East'}(\text{Sells} \bowtie \text{Restaurants})$

To prove $S \bowtie \sigma_E(R) \equiv \sigma_E(S \bowtie R)$

we can split R into 2 partitions $\sigma_E(R) \cup (R - \sigma_E(R))$

$$S \bowtie R \equiv S \bowtie \sigma_E(R) \cup S \bowtie (R - \sigma_E(R))$$

$$\sigma_E(S \bowtie R) \equiv \sigma_E(S \bowtie \sigma_E(R)) \cup \sigma_E(\underbrace{S \bowtie (R - \sigma_E(R))}_{\text{select nothing}})$$

$$\equiv S \bowtie \sigma_E(R)$$

Q_1 and Q_2 are equivalent

5.

$Q_1 = ((S \bowtie \sigma_E R) \cup (S \bowtie \sigma_L(L)))$

$Q_2 = (\sigma_R \wedge \sigma_E) \vee (\sigma_P \wedge \sigma_L)(S \times R \times L)$

$\equiv \sigma_R \wedge \sigma_E (S \times R \times L) \cup \sigma_P \wedge \sigma_L (S \times R \times L)$

$\equiv \sigma_E(S \bowtie R) \cup \sigma_L(S \bowtie L)$ (π_{rname} eliminates duplicates)

$\equiv S \bowtie \sigma_E(R) \vee S \bowtie \sigma_L(L)$ (Q4)

equivalet ?

6. create temporary office / assume already in database since did not give new office_id

```
UPDATE Employees
SET office_id =   ( SELECT office_id
                    FROM Offices
                    WHERE building = 'Tower 1'
                    AND room_number = 11
                    AND level = 5 )
WHERE office_id = 123;
```

7. a)

X	A	Y	B	Z	C	D
30	8	0	5	1	60	500
60	4	1	3	3	30	100

need include A, B for both R & S ✱

b)

X	A	Y	R.B	Z	S.B	C	D
30	8	0	5	1	5	60	500
60	4	1	3	3	2	40	200
60	4	1	3	3	3	30	100

c)

inner join

X	R.A	Y	R.B	Z	S.A	S.B	C	D
30	8	0	5	1	8	5	60	500
60	4	1	3	3	4	2	40	200
60	4	1	3	3	4	3	30	100
0	10	0	9	2	NULL	NULL	NULL	NULL
0	0	0	4	5	NULL	NULL	NULL	NULL

d)

	X	R.A	Y	R.B	Z	S.A	S.B	C	D
3 rows, inner join									
	NULL	NULL	NULL	NULL	NULL	17	1	20	100

e)

	X	R.A	Y	R.B	Z	S.A	S.B	C	D	
3 rows inner join										
	NULL	NULL	NULL	NULL	NULL	17	1	20	100	dangling right
	0	10	0	9	2	NULL	NULL	NULL	NULL	dangling left
	0	0	0	4	5	NULL	NULL	NULL	NULL	