CS2102 Tutorial 1

1. a) superkey is a subset of attributes that uniquely identifies a tuple in a relation

A	X	0	BC	X	12
B	X	0, 1	BD	X	10
C	X	2	CD	X	20
D	X	0	ABC	✓	no duplicates
AB	X	00	ABD	✓	no duplicates
AC	✓	no duplicates	ACD	✓	no duplicates
AD	✓	no duplicates	BCD	X	120
			ABCD	✓	no duplicates

supersets of superkeys/keys are also superkeys

superkeys of R are $\{A, C\}$, $\{A, D\}$, $\{A, B, C\}$, $\{A, B, D\}$, $\{A, C, D\}$, $\{A, B, C, D\}$

b) keys are superkeys that are minimal
candidate keys are the set of all keys for a relation

If $\{A, C\}$ is a superkey, any superkey which is a superset of $\{A, C\}$ cannot be minimal, hence cannot be a candidate key

only $\{A, C\}$, $\{A, D\}$, $\{A, B, D\}$ remain

since $\underline{\{A, C\}}$ is a superkey and is also minimal, it must be a candidate key

If $\{A, D\}$ is a candidate key, then $\{A, B, D\}$ cannot

2. A foreign key is a set of attributes that refer to the primary key of another relation

each foreign key in referencing relation must

① appear as primary key in referenced relation OR

② null value

S. _ → R.A

W	✓
X	X 4 does not map to R.A
Y	✓
Z	✓ null ok

W, Y, Z are possible foreign keys

3. a) Equivalent

attribute A available for select and project in both queries

order of operation does not impact result

b) Not Equivalent

Q_2 is invalid as attribute C is no longer available after projection

selection condition refers to non existent attribute

c) Equivalent

both queries have $|S|*|T|$ entries and columns D, Y

d) Equivalent

projection reorders both columns to be the same

e) Equivalent

cross product in relational algebra is associative

order does not affect result

f) Equivalent

Both projection and union removes duplicates

g) Not equivalent

eg A tuple (a, c) in R and (a, d) in S

Q_1: (a, c) still remains after $R - S$ hence (a) in $\pi_A(R-S)$

Q_2: $\left.\begin{array}{l} \pi_a(R) \Rightarrow \{(a)\} \\ \pi_a(S) \Rightarrow \{(a)\} \end{array}\right\}$ set difference $= \emptyset$

4. a) Find all pizzas Alice likes but not liked by Bob

set difference between set of pizzas Alice likes and set of pizzas Bob likes

$$\boxed{\pi_{pizza}(\sigma_{cname='Alice'}(Likes)) - \pi_{pizza}(\sigma_{cname='Bob'}(Likes))}$$

b) Find all customer-restaurant pairs (C, R) where C and R in the same area

C likes some pizza sold by R

① To find (C, R) pairs in the same area

$Q_1 = Customers \bowtie Restaurants$ (natural join over common area attribute)

② To find pizzas C likes

$Q_2 = Q_1 \bowtie Likes$ (natural join over cname)

③ To find pizzas sold by R liked by C

$Q_3 = Q_2 \bowtie sells$ (natural join over rname and pizza)

④ project the (C, R) pairs

$\pi_{cname, rname}(Q_3)$

$$\boxed{\pi_{cname, rname}(Customers \bowtie Restaurants \bowtie Likes \bowtie Sells)}$$

equivalent to

$$\left(\pi_{c_1, r_1}\left(\sigma_{(a1=a2) \wedge (c1=c2) \wedge (r1=r2) \wedge (p_1=p_2)}\left(\begin{array}{c} \rho(c1, a1)(Customers) \\ \times \\ \rho(r1, a2)(Restaurants) \\ \times \\ \rho(c2, p_1)(Likes) \\ \times \\ \rho(r2, p2, price)(Sells) \end{array}\right)\right)\right)$$

c) Find pizzas each customer dislikes

If (cname, pizza) not in Likes, customer dislikes

dislikes is the set difference between all customer-pizza pairs and Likes

$$\boxed{\pi_{cname,\, pizza}(\text{Customers} \times \text{Pizza}) - \text{Likes}}$$

d) Find all customer pairs (c_1, c_2) such that c_1 likes some pizza that c_2 dislikes

natural join on pizza attribute of Likes and DisLikes

$$\boxed{\pi_{cname,\, cname2}(\text{Likes} \bowtie \rho_{cname2 \leftarrow cname}(\text{Dislikes}))}$$

e) Find customer pairs $(C1, C2)$ such that $C1 < C2$ and they like the same pizzas

customer pairs who like at least 1 pizza + ordered

$$Q_1 = \pi_{cname,\, cname_2}(\sigma_{cname < cname_2}(\text{Likes} \times \rho_{cname2 \leftarrow cname}(\text{Likes})))$$

set difference with Like Dislikes to remove pairs who do not like same pizza

$$Q_2 = \text{Like Dislike} \cup \pi_{cname_2,\, cname}(\text{Like Dislike})$$

$$Q_1 - Q_2$$

f) Find most expensive for each restaurant $\quad$ Max

$Q_1 = \pi_{rname, price}(\text{sells})$

$\Rightarrow$ there exists a price that is higher $=$ not max

$Q_2 = \pi_{rname, price}\left(\sigma_{(rname = rname2) \wedge (price < price2)}\left(Q_1 \times \rho_{(rname2, price2)} Q_1\right)\right)$

$Q_1 - Q_2$

set difference between all (restaurant - price) pairs and price which are not max for each restaurant

g) pizzas sold and their areas

$Q_1 = \text{sells} \bowtie \text{Restaurants}$ $\quad$ natural join over rname

$Q_2 = \pi_{cname, pizza}(\text{Customers} ⟕ Q_1)$ $\quad$ natural join over area include dangling customers where no pizza sold in their area

5.

R_1 : All pizzas Maggie likes

R_2 : All (restaurant x pizza which Maggie likes) pairs

R_3 : Restaurants that do not sells pizzas Maggie likes

R_4 : Restaurants that sells pizzas Maggie likes

R_5 : All pizzas Ralph likes

R_6 : Restaurants that sell pizzas Ralph likes

R_7 : Restaurants that sells pizzas Maggie likes but not Ralph