## Relational algebra

*P* → Pizza(<u>*code*</u>, *pname, size*)
*S* → Store(<u>*sname*</u>, *area, phone*)
*L* → Sells(<u>*sname, code*</u>, *price*)

1. Find the names of pizzas that come in a 10 inch size.

$$\pi_{pname}(\sigma_{size=10}(P))$$

2. Find the names of pizzas that come in a 10 inch or a 12 inch size.

$$\pi_{pname}(\sigma_{size=10 \vee size=12}(P))$$

3. Find the names of pizzas that come in both a 10 inch and a 12 inch size.

$$\pi_{pname}(\sigma_{size=10}(P)) \cap \pi_{pname}(\sigma_{size=12}(P))$$

4. Find the pairs of different codes of pizzas with the same name and the same size (is there any?).

$$\pi_{P1.code,P2.code}(\sigma_{P1.code \neq P2.code \wedge P1.name = P2.name \wedge P1.size = P2.size}\ (\rho(P1,P) \times \rho(P2,P)))$$

5. Find the names and phone numbers of the stores in "College Park" or "Greenbelt" that sell a 10 inch pizza named "pepperoni" for less than $8.

$\pi_{S.sname,phone}($

$\sigma_{size=10 \wedge pname=\text{`pepperoni'} \wedge price<8 \wedge (area=\text{`CollegePark'} \vee area=\text{`Greenbelt'})}(P \bowtie n\ S \bowtie n\ L))$

please note that the attribute names are changed, as given at the beginning of this answer sheet. Otherwise, we cannot use nature join here.

6. Find the codes of the most expensive pizzas assume the scheme of the database is reduced to a relation P → Pizza(<u>code</u>, price) to simplify.

$$\pi_{P1.code,P2.code}(\sigma_{P1.price \geq P2.price}(\rho(P1,P) \times \rho(P2,P)))/\pi_{code}(\rho(P1,P))$$

The intuition is: (i) Find all pairs (code1, code2) of pizza codes where the price of code1 is more or equal to the price of code2. (ii) For a specific c ∈ code1, if c is paired with all possible codes, it means that its price is more or equal to all prices. Therefore c is the most expensive pizza. Note that many such pizzas may exist (if all have the same high price).

Recall that, for two tables A(x, y), B(y), division is defined as:

$$A/B = \pi_x(A) - \pi_x((\pi_x(A) \times B) - A)$$

7. Find the names of the stores that sell all the pizzas.

$\pi_{sname,code}(L)/\pi_{code}(P)$

Observe. This is division. In this case:

$$A \equiv \pi_{sname,code}(L)$$

$$B \equiv \pi_{code}(P\ )$$

**SQL**

Find the names of pizzas that come in a 10 inch size

SELECT name FROM pizza WHERE size = 10

Find the names of pizzas that come in a 10 inch or a 12 inch size

SELECT name FROM pizza WHERE size = 10 OR size = 12

Find the names of pizzas that come in both a 10 inch and a 12 inch size

SELECT P1.name FROM pizza P1, pizza P2 WHERE P1.size = 10 AND P2.size = 12 AND P1.name=p2.name

Find the pairs of different codes of pizzas with the same name and the same size (is there any?)

SELECT T1.code, T2.code FROM pizza T1, pizza T2
WHERE T1.code <> T2.code AND T1.name = T2.name AND T1.size = T2.size

Find the names and phone numbers of the stores in "College Park" or "Greenbelt" that sell a 10 inch pizza named "pepperoni" for less than $8

SELECT T2.name, T2.phone FROM pizza T1, store T2, sells T3
WHERE T1.code= T3.code AND T2.name = T3.store_name
AND (T2.area = 'College Park ' OR T3.area = ' Greenbelt ')
AND T1.name = 'pepperoni' ∧ T1.size = 10

Find the codes of the most expensive pizzas – assume the scheme of the database is reduced to a relation pizza(code, price) to simplify

SELECT p.code FROM pizza p
WHERE p.price >=ALL(SELECT p1.price FROM pizza p1)

Find the names of the stores that sell all the pizzas

SELECT st.name FROM store st
WHERE NOT EXISTS (SELECT * FROM pizza p
WHERE NOT EXISTS (SELECT * FROM sells s
WHERE st.name=s.sname and p.code=s.code))