

CSC343 Worksheet 4 Solution

June 18, 2020

1. a) $[(1, 0, 1), (5, 4, 9), (1, 0, 1), (6, 4, 16), (7, 9, 16)]$
- b) $[(1, 0), (3, 3), (3, 4), (4, 3), (1, 1), (4, 3)]$
- c) $[(0, 1), (0, 1), (2, 3), (2, 4), (3, 4)]$

Notes:

- $\tau_L(R)$ sorts tuples in order indicated by L .
 - e.g.

$\tau_{C,B}(R)$ in $R(A, B, C)$ orders the tuples of R by their values of C , and tuples with the same C -value are ordered by their B value.

- d) $[(0, 1), (0, 2), (2, 4), (2, 5), (3, 4), (3, 4)]$
- e) $[(0, 1), (2, 4), (2, 5), (3, 4), (0, 2)]$

Notes:

- $\delta(R)$ converts a bag into a set
 - e.g.

Let $R = [(1, 2), (3, 4), (1, 2), (1, 2)]$

$\delta(R(A, B)) = [(1, 2), (3, 4)]$

- f) $[(0, 2), (2, 7), (3, 4)]$

Notes:

- $\gamma_L(R)$ is an operator that groups a relation and/or aggregate some columns.
 - L in $\gamma_L(R)$ is either
 1. **Grouping attribute** or an attribute by which R will be grouped.

2. **Aggregated attribute** or an attribute where an aggregation operator is applied to.

Example:

$\gamma_{starName, MIN(year) \rightarrow minYear, COUNT(title) \rightarrow ctTitle} (StarsIn)$

studioName
Disney
Disney
Disney
MGM
MGM

groups by studioName

Figure 5.4: A relation with imaginary division into groups

- g) $[(0, 1.5), (2, 4.5), (3, 4)]$
 h) $[(0, 1), (0, 1), (2, 3), (2, 4), (3, 4)]$
 i) $\gamma_{A, MAX(C)}([(2, 3, 4), (2, 3, 4)]) \rightarrow [(2, 4)]$
 j) $[(0, 1, \perp), (2, 3, 4), (2, 3, 4), (0, 1, \perp), (2, 4, \perp), (3, 4, \perp)]$

Notes:

- \bowtie is an outerjoin operator
 - \bowtie_L means Natural Left Outer Join
 - \bowtie_R means Natural Right Outer Join
 - \bowtie means Natural Full Outer Join
 - \perp means null
- e.g. $U \bowtie V$

<i>A</i>	<i>B</i>	<i>C</i>
1	2	3
4	5	6
7	8	9

(a) Relation *U*

<i>B</i>	<i>C</i>	<i>D</i>
2	3	10
2	3	11
6	7	12

(b) Relation *V*

<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>
1	2	3	10
1	2	3	11
4	5	6	⊥
7	8	9	⊥
⊥	6	7	12

(c) Result $U \bowtie V$

- k) $[(\perp, 0, 1), (\perp, 2, 4), (\perp, 2, 5), (2, 3, 4), (\perp, 0, 2), (2, 3, 4)]$
 l) $[(0, 1, \perp), (2, 3, 4), (2, 3, 4), (0, 1, \perp), (2, 4, \perp), (3, 4, \perp),$
 $(\perp, 0, 1), (\perp, 2, 4), (\perp, 2, 5), (2, 3, 4), (\perp, 0, 2), (2, 3, 4)]$
 m) $(0, 1) : \{(2, 4), (2, 5), (3, 4), (3, 4)\}$

But, $\{(2, 3), (2, 4), (3, 4)\}$ from *R* and $\{(0, 1), (0, 2)\}$ in *S* don't match. So,

$[(0, 1, 2, 4), (0, 1, 2, 5), (0, 1, 3, 4), (0, 1, 3, 4), (0, 1, 2, 4), (0, 1, 2, 5), (0, 1, 3, 4), (0, 1, 3, 4),$
 $(2, 3, \perp, \perp), (2, 4, \perp, \perp), (3, 4, \perp, \perp), (\perp, \perp, 0, 1), (\perp, \perp, 0, 2)]$

Notes:

- $R \bowtie_C S$ is equivalent form of $\sigma_C(R \times S)$ but instead of filtering, the unmatched tuples are filled with null.

2. a) SELECT model FROM PC WHERE speed > 3.0;
 b) SELECT DISTINCT maker FROM Products NATURAL JOIN Laptops WHERE hd >= 100;

```

c)      SELECT model, price FROM (
2         (SELECT model, price FROM PC NATURAL JOIN Products)
3         UNION
4         (SELECT model, price FROM Laptop NATURAL JOIN Products)
5         UNION
6         (SELECT model, price FROM Printer INNER JOIN Products ON
Printer.model = Product.model)
7     );
8

```

d) SELECT model FROM Printer WHERE color;

e)

```
(SELECT DISTINCT makers FROM Products WHERE type='laptops') -  
(SELECT DISTINCT makers FROM Products WHERE type='pc');
```

f)

```
SELECT hd FROM PC WHERE EXISTS (  
    SELECT hd, COUNT(model) FROM PC GROUP BY hd  
    HAVING COUNT(model) > 2  
);
```

3. a) SELECT class, country FROM classes WHERE bore >= 16;

b) SELECT * FROM Ships WHERE launched < 1921;

c) SELECT * FROM Outcomes WHERE result='sunk';

d) SELECT * FROM Classes NATURAL JOIN Ships WHERE displacement > 35000;

e)

```
SELECT name, displacement, numGuns FROM Classes NATURAL JOIN (  
    SELECT * FROM Ships INNER JOIN Outcomes ON Ships.name =  
    Outcome.ship  
);
```

f)

```
(SELECT name FROM Ships)  
UNION  
(SELECT ship AS name FROM Outcomes);
```

g)

```
SELECT class, COUNT(class) FROM Ships  
GROUP BY Class  
HAVING COUNT(class) = 1;
```

h)

```
(SELECT countries FROM Classes WHERE type='bb')  
INTERSECT  
(SELECT countries FROM Classes WHERE type='bc');
```

i) Current attempt:

```
(SELECT Table1.name FROM Outcomes AS Table1 INNER JOIN Ships ON  
    Outcomes.ship = Ships.name)
```

Took too much time. Omitted for now.

4. a) SELECT AVG(speed) FROM PC;

b) SELECT AVG(speed) FROM Laptop HAVING price > 1000;

c) SELECT AVG(price) FROM PC NATURAL JOIN Product HAVING maker = 'A';

d)

```

1  SELECT AVG(price) FROM (
2      (SELECT model, price FROM PC NATURAL JOIN Product WHERE maker
3      = 'D')
4      UNION
5      (SELECT model, price FROM Laptop NATURAL JOIN Product WHERE
6      maker = 'D')
7  );

```

e) SELECT speed, AVG(price) FROM PC GROUP BY speed;

f) SELECT maker, AVG(screen) FROM Laptop NATURAL JOIN Product GROUP BY maker;

g)

```

1  SELECT maker, COUNT(model) FROM Products GROUP BY maker HAVING
2  COUNT(model) >= 3;
3

```

h)

```

1  SELECT maker, MAX(price) FROM PC
2  NATURAL JOIN Products GROUP BY maker;
3

```

i)

```

1  SELECT speed, AVG(price) FROM PC GROUP BY speed
2  HAVING speed > 2.0;
3

```

5. a) SELECT COUNT(class) FROM Classes;

Correct Solution:

```
SELECT COUNT(class) FROM Classes HAVING type='bb';
```

b) SELECT class, AVG(numGuns) FROM Classes GROUP BY class;

Correct Solution:

```
SELECT class, AVG(numGuns) FROM Classes GROUP BY class
HAVING type='bb';
```

c) SELECT AVG(numGuns) FROM Classes HAVING type='bb';

d) SELECT class, MIN(launched) FROM Ships GROUP BY class;

e)

```

1  SELECT class, COUNT(Ships.name) FROM Ships INNER JOIN Outcomes
2  GROUP BY class ON Ships.name = Outcomes.ship
3  HAVING result='sunk';
4

```

6.

```

1  SELECT name, MIN(movieYear) FROM MovieStar INNER JOIN StarsIn ON
2  StarsIn.starName = MovieStar.name GROUP BY name HAVING COUNT(name)
3  >= 3;

```

7. Yes. It is possible

- Rename aggregate columns using ρ
- Use σ around $\rho(\gamma(\dots))$

8. a) Notes:

- Insertion
 - **Syntax:** INSERT INTO R(A_1, A_2, \dots, A_n) VALUES (v_1, \dots, v_n)
 - Inserts a tuple with values into relation R