CS3550

Final- Exam

T. Akash
ES18BTECH11019.

(1)

```
SELECT *
FROM E, D
WHERE dept = dnr.
```

(a) We can write the given SQL using "σ" and "$\times$" relational operators. We don't need a "Π" operator as we should select all the attributes from the Cartesian Product of those relations.

$\therefore$ $\boxed{\sigma_{dept = dnr}(E \times D)}$

(b) Resulting Tuples are :

enr	ename	dept	dnr	dname
1	Bill	A	A	Marketing
2	Sarah	C	C	Legal
3	John	A	A	Marketing.

(2) R_1, R_2 → Relations containing N_1 and N_2 tuples

$N_2 > N_1 > 0$

(a) $R_1 \cup R_2$:

Assumptions: R_1 and R_2 must be Union Compatible.

For the Set Operations to be Valid, We require two Conditions,

(1) The domains of i^{th} Attribute of R_1 and i^{th} attribute of R_2 must be same for all i.

(2) R_1 and R_2 must have same number of attributes.

Minimum: N_2 ways

↓

Possible when $R_1 \subseteq R_2$

Maximum: $(N_1 + N_2)$ ways

↓

Possible when R_1 and R_2 are disjoint.

(b) $R_1 \cap R_2$:

Assumptions: R_1 and R_2 must be Union Compatible and the assumptions are same as of "$R_1 \cup R_2$".

Minimum: 0 → Possible when R_1 and R_2 are disjoint.

Maximum: N_1 → Possible when $R_1 \subseteq R_2$.

(c) $R_1 - R_2$:

Assumption : R_1 and R_2 must be Union Compatible and assumptions are Same as "$R_1 \cup R_2$".

Minimum: 0 → Possible if $R_1 \subseteq R_2$

Maximum: N_1 → Possible When R_1, R_2 are disjoint.

$$R_1 - R_2 = R_1 - \underbrace{(R_1 \cap R_2)}_{0}$$

$$= N_1$$

(d) $R_1 \times R_2$: No assumptions.

Minimum: $N_1 * N_2$

Maximum: $N_1 * N_2$

} Cartesian Product only depend on dimensions of the Relations not on their properties such as disjoint or subset.

(e) $\sigma_{a=5}(R_1)$:

Assumption: R_1 has an attribute named "a"

Minimum: 0 → Possible When "No Tuple" has an attribute named "a" Whose Vallue is '5'.

Maximum: N_1 → Possible when all tuples have an attribute named "a" and equal to '5'.

③

```
SELECT *
FROM EMPLOYEE
WHERE Salary =
    (
      SELECT MIN(Salary)
      FROM EMPLOYEE
      WHERE Salary IN (
          SELECT DISTINCT TOP(10) Salary
          FROM EMPLOYEE
          ORDER BY Salary DESC
      )
    )
```

Explanation:

① Inner most Query finds the employees With top 10 distinct Salaries

② The next query finds the lowest Salary among the Salaries fetched by the above query. This will give us 10th highest Salary

③ Final query selects details of the employee who has Salary fetched by above query.

④ update TBL

set Nmbr = case

when Nmbr = 0 then Nmbr + 2

else Nmbr + 3

end

⑤ $\left(\prod_{sid}\left(\left(\sigma_{color='red'}(Parts) \bowtie \left(\sigma_{cost<100}(Catalog)\right) \bowtie Suppliers\right)\right)\right.$

$\cap$

$\left.\left(\prod_{sid}\left(\left(\sigma_{color='green'}(Parts)\right) \bowtie \left(\sigma_{cost<100}(Catalog)\right) \bowtie Suppliers\right)\right)\right)$

⑥ ρ_{E_1} (Employees)

ρ_{E_2} (Employees)

$\rho_{E_3}\left(\prod_{E_2.eid}\left(E_1 \bowtie_{(E_1.salary > E_2.salary)} E_2\right)\right)$

$\left(\prod_{eid}(E_1)\right) - E_3$

→ Finding all the employees who donot have the highest Salary. Subtracting these from the Original list of employees. We are left with highest paid employees.

(7) → First finding the highest paid employees as in question (6)

→ Removing the highest paid employees from the original list.

We are now left with second highest paid employees along with rest of employees.

→ Now we select the highest paid employees in this new list.

$$\rho_{E_1}(\text{Employees})$$

$$\rho_{E_2}(\text{Employees})$$

$$\rho_{E_3}\left(\Pi_{E_2.eid}\left(E_1 \bowtie_{E_1.Salary > E_2.Salary} E_2\right)\right)$$

$$\rho_{E_4}(E_2 \bowtie E_3)$$

$$\rho_{E_5}(E_2 \bowtie E_3)$$

$$\rho_{E_6}\left(\Pi_{E_5.eid}\left(E_4 \bowtie_{E_4.Salary > E_5.Salary} E_5\right)\right)$$

$$\left(\Pi_{eid} E_3\right) - E_6$$

(8)

```
SELECT  P.pname, MAX(C.cost)
FROM    Suppliers as S, Parts as P, Catalog as C
WHERE   P.pid = C.pid AND C.sid = S.sid
GROUP BY  S.sname, S.sid
HAVING  ANY(P.Color = 'red') AND ANY(P.Color = 'green')
```

(9)

```
SELECT U.user_id, username, training_id, training_date,
       COUNT(user_training_id)
FROM   users as U, training_details as T
WHERE  T.user_id = U.user_id
GROUP BY U.user_id, username, training_id, training_date
HAVING COUNT(user_training_id) > 1
ORDER BY training_date DESC.
```

(10)

```
SELECT sum(case
       when x>0 then x
       else 0
       end) as sum_pos,
       Sum(case
       when x<0 then x
       else 0
       end) as sum_neg
FROM A.
```