02 Queries

1. (c)

Query: $R \cup \rho_{S(A,B)}\, S$

Minimum = Max (r, s) i.e., is one relation is a result of the other.

Maximum = $r + s$ i.e., if relations are disjoint.

2 (c)

1. A consecutive selection condition can be broken up into a cascade / sequence of individual σ operations.
2. In a cascade (sequence) of π operations, all but the last one can be ignored.
3. If the selection condition c involves only those attributes $A_1, A_2, \ldots, A_n$ in the projection list, the two operations can be commuted.

3. (c)

In this query, we can explicitly show the sequence of operations, giving a name to each intermediate relation. 1st intermediate will have SSN of all the managers, 2nd intermediate will have SSN of all the employees with dependents. 3rd intermediate will have those department managers who don't have any dependent. Final result will give the name of those department managers who have no dependents.

4. (100)

$P \bowtie Q \Rightarrow$ Common attribute = C which is key for both P and Q.

$R_1 = P \bowtie Q \Rightarrow \min(200, 300) = 200$ tuples

$R_1 \bowtie R \Rightarrow$ Common attribute = E which is key for both R_1 and R.

$R_1 \bowtie R \Rightarrow \min(200, 100) = 100$ tuples

So, $|P \bowtie Q \bowtie R| = 100$ tuples

5. (d)

Both the statements are incorrect.

The select operation is commutative i.e. $\sigma_{c_1}(\sigma_{c_2}(R)) \Leftrightarrow \sigma_{c_2}(\sigma_{c_1}(R))$.

Ultimately only those tuples will be selected which satisfy both C_1 and C_2. Hence order does not matter.

But Π (projection) operation is not commutative.

$\Pi_{a_1}(\Pi_{a_2}(R)) = \Pi_{a_1}(R)$ if and only if a_1 is substring (or subset) of a_2, otherwise operation would be incorrect.

6. (240)

R contain 400 tuples and values for A are uniformly distributed in the interval [1, 25]

Number of values in [1, 25] = 25

Since numbers are uniformly distributed then

a number repeats = $\frac{400}{25} = 16$

$\therefore$ Each value in domain repeats for 16 times.

So number of tuples $\le 15 = 15$

Hence, $\sigma_{A \le 15}(R) = 15 \times 16 = 240$ tuples.

7. (6000)

Since 'P' in R_2 is not key, hence all value of 'P' may or may not be unique. Hence every entry under 'P' in R_2 will match with 'P' in R_1.

Hence maximum is 2000. But 'P' in R_2 is foreign key referencing 'P' in R_1.

Therefore minimum is also 2000.

$\therefore$ $X = 2000$,

$Y = 2000$;

$X + 2Y = 2000 + 4000$

$= 6000$

8. (a)

Let R = {(3, 4)} and S = {(1, 2)}

- Produce empty result, since no common attribute value is there for natural join.
- $\phi \times (1, 2) \rightarrow \phi$ {empty result}
- Produce (3, 2) as output

Hence query 3 is different.

9. (b)

(a) finds those drinkers who are frequents atleast one bar on same city where he lives.

10. (d)

Option (a) represent flight number that can not be piloted by any pilot whose salary is over $100000, since aircraft can be used for any flight provided it has sufficient range but here range is less than sufficient distance.

Option (b) and (c) represent find flno of flights that can be piloted by every pilot whose salary is over $100000.

11. (a)

HAVING is performed after GROUP BY. If you have to apply some conditions to get results. We need to use WHERE before group by.

12. (c)

Both query (a) and (b) represent the names of suppliers supplying some red part for less than 100 cost, but query2 is optimized variation of query1.

13. (a)

- Secondary index over key build based on unordered field so that it is dense.
- Clustering index based on non key may be sparse also so, clustering index may be dense if each cluster one record.
- Primary index on key with ordering may be dense or sparse.

14. (2)

I. $\pi_{Eid}(\text{Employee}) = \{1, 2, 3, 4, 5\}$

II. $\pi_{Eid}(\text{Employee} \bowtie_{(Eid = Deid) \wedge (Dage \le Eage)} (\text{Dependent}))$ results employee id whose age is greater than equal to his/her dependent i.e. 1, 2, 3.

So, I – II = {1, 2, 3, 4, 5} – {1, 2, 3} = {4, 5}

15. (d)

- Maximum number of tuple for $R \cup S$ is $n + m$.
- Maximum number of tuple for $R \cap S$ is min (m, n).
- Maximum number of tuple for $\sigma_C(R) \times S$ is $m \times n$.
- Maximum number of tuple for $\sigma_L(R) - S$ is n.

16. (d)

(a) $\pi_{A1}(R_1 - R_2) \neq \pi_{A1}(R_1) - \pi_{A1}(R_2)$ because

R_1

A_1	A_2
2	4
3	4
2	5
3	5

R_2

A_1	A_2
2	4
2	5
3	5

LHS results:

A_1
3

RHS result:

A_1
Empty

(c) $\pi_{A1}(\sigma_{C1}(R_1)) \rightarrow \sigma_{C1}(\pi_{A1}(R_1))$ because LHS is always superset of RHS.

(d) $\pi_{A1}(\pi_{A2}(\sigma_{C1}(\sigma_{C2}(R_1)))) \rightarrow \pi_{A1}(\sigma_{C2}(\sigma_{C1}(R_1)))$ with condition $A_1 \subset A_2$ it gives the same results when LHS is replaced by RHS.

17. (d)

- Option (a) represent course in which proper subset of male student are enroll.
- Option (b) represent course in which only male student are enroll.
- Option (c) represent course in which only male student are enroll.

Hence both (b) and (c) are correct.

18. (5)

AB

W	Z
10	15
10	25
10	20
15	15
15	25
15	20

$-$

X	Z
10	15

$=$

WZ

W	Z
10	25
20	20
15	15
15	25
15	20

Total 5 tuples will be returned.

19. (d)

$\pi_B\left(R \underset{R.B \neq S.C}{\bowtie} S\right)$ which selects all foreign key values.

$\pi_C\left(R \underset{R.B \neq S.C}{\bowtie} S\right) - \pi_C(S) \equiv$ Always empty

All foreign keys values – All primary key values ≡ Always empty

20. (a)

Queries 1 will returns pairs of Sids such that the supplier with the first Sid charges more for some part than the supplier with the second Sid. Which is possible by the condition that is when two supplier id is different but part id is same and charges of first supplier is more than second supplier.

Queries 2 will returns empty set because we compare cost with Sid which is always returns empty set.

21. (500)

X_1 is the common attribute in between X and Y and is a primary key of X, therefore, all 100 values of X_1 in X are unique, since X contains 110 X_1 values and Y contains 500 values of X_1 then these 110 unique values could be associated with 500 values of X_1 in Y. But X_1 in Y can have multiple entries of the same. Hence, the maximum number of tuples in the result set of REQUIJOIN S will be 500.

22. (d)

In the A, B and C option, the relation between all the three relations R, S and T is preserved but in (d) the relationship between T and S is not preserved as it works as union between $(R \bowtie T)$ and $(R \bowtie S)$.

23. (d)

Left and right outer joints can yield result set of sizes both generator or smaller than each other depending on the data. Full outer join will always have a set size greater than (or equal to) the result of left or right joins.

24. (1000)

C is the common attribute in between relation R_1 and R_2 an is a primary key of R_1, therefore, all 500 values of C in R_1 are unique. Since R_1 contain 500 C values and R_2 contains 1000 values of C then these 500 unique values could be associated with 1000 values of C in R_2. Hence, the maximum number of tuples in the result set of NATURAL JOIN will be 1000.

25. (b)

$\sigma_{(P.Y = R.Y \wedge R.V = V2)}(P \times R)$

$\rightarrow$

X	Y	Z	V
X2	Y2	Z2	V2

$\pi_x(\sigma_{(P.Y = R.Y \wedge R.V = V2)}(P \times R)) \rightarrow$

X
X2

$\sigma_{(Q.Y = R.Y \wedge Q.T > 2)}(Q \times R)$

$\rightarrow$

X	Y	T	V
X1	Y1	6	V1
X1	Y2	5	V3
X1	Y2	5	V2

$\pi_x(\sigma_{(Q.Y = R.Y \wedge Q.T > 2)}(Q \times R)) \rightarrow$

Y
X1

S, final answer $\rightarrow$

X
X2

$-$

Y
X1

$= 1$

26. (d)

Course_id
CS-101
CS-347
PHY-101

–

Course_id
CS-101
CS-315
CS-311
FIN-201
HIS-351
MU-199

=

Course_id
CS-347
PHY-101

27. (a)

For the output to be same as relation A two conditions must be satisfied.

(i) **A has no duplicates:** If A has duplicates then output will not contain duplicates because of DISTINCT keyword (output will not be same as $A(a, b)$).

(ii) **B is non-empty:** If $B(c, d)$ is empty then cross product of A and B will result empty rows, so is the output (output will not be same as $A(a, b)$).

28. (c)

First query will successfully execute and after executing this statement, attribute Did of Employee table with ID = 1 becomes 'NULL' which is foreign key from Department's Did. Now, second statement will not execute, because primary key cannot be NULL.

29. (4)

The output Table will be

Dealer-No.	Color-id
D_2	C_2
D_7	C_3
D_2	C_5
D_7	C_6

30. (6)

SQL query will return the following output.

Emp_no.	Efloor	
1798	7	
1798	7	Duplicate Row is present in Hobby table
2369	7	
2581	6	
2643	5	
2643	5	Duplicate Row is present in Hobby table
2872	5	
2926	5	
2959	7	

Total 9 rows are selected

In Relation Algebra, only distinct values of Efloor are selected i.e. 5,6,7.

Therefore SQL row count RA row count = 9 – 3 = 6.

31. (a)

Query I returns vertices whose out degree is only 2. But Query II returns whose out degree is atleast 2.

32. (c)

S_1 : SQL's aggregation can not be expressed in relational algebra.

S_2 : Select clause is same as project in relation algebra.

S_3 : Arithmetic operation is return null if one of the value is null.

All S_1, S_2 and S_3 are correct.

33. (3)

The query will return "how many students took each exam" i.e. 3 rows (1, 3) (2, 4) and (3, 2).

34. (3)

1. Query: (SELECT Dept_name, avg (Salary) as avg_salary
   FROM Instructor
   GROUP by Dept_name)
   will return department name with their average salary.
2. Outside query return department name with average salary whose average salary more than 42000.
   i,e. IN, CS, CE.

35. (b)

Given query will result tupple where $A \geq 5$ (i.e. $A = 5, 6, 10, 8$) and $B \geq C$ (i.e. $B = 8$ because we can not compare Null values) or $C \geq 5$ (i.e. $C = 5, 6, 5$ because we can not compare Null values) so the resulting tupple are 2 i.e. $A = 6$, $A = 8$.

36. (b)

Condition E_1.salary < E_2.salary; will find all salaries accept highest. Aggregate function MAX(E_1.salary) will find maximum of all salaries which are less then highest salaries i.e. second highest salary.

37. (c)

Here, we are asked to COUNT all the course's NOT IN (NULL), this predicate evaluates to unknown because comparison with NULL can never result in either True or False, it always gives a third logical result, unknown. Hence, no rows will be printed i.e., 0.

38. (a)

39. (b)

Given a query throws an error, because every column that is specified in the group by clause should also be present in the SELECT clause. Here dept_id which is there in Group By Clause is not present in select clause. Therefore, it throws an error.

40. (d)

Since it is mentioned that the number of tuples in relation R can be n and n is greater than 0. And nothing is specifically given about the domain of the attribute A_1 and B. Therefore, we cannot tell the number of tuples returned by the given query.

41. (c)

It is not a valid insertion as the sequence of attribute name mentioned is different from the sequence of values inserted. Therefore, the value will be an error due to type mismatch. Hence it is not valid.

42. (c)

Both (a) and (b) will result in $R \cap P$. In (a) It is easily visible that it is intersection and in (b) nested query 2 will return all P tuples (SELECT * FROM P) and nested query 1 will return tuples of r which is not similar to tuples return by nested query 2 (SELECT * FROM R WHERE (a, b, c) NOT IN (nested query 2)) and finally main query will return all those tuples which are common in both the relations.

43. (5)

The inner query (SELECT age FROM customer WHERE age > 21) will return an empty set and we know that ALL returns true if all of the subquery values meet the condition.
Hence, all the distinct cid will be there in the result set. Since there are 5 cid and all of them are distinct. Hence, the result set will contains 5 tuples.

44. (d)

A manager must inherit all the rights in order to set the access rights.

45. (d)

Except clause returns all records present in the first table and absent in the second one but this clause eliminate duplicates.
Except ALL returns all records from the first table which are not present in the second table, leaving the duplicate as it is

Output of I query:

Y
4
5

Output of II query:

Y
2

46. (0)

The inner query will return the output as (5, NULL, NULL) Now this compared by the output of the outer query which is (5, NULL, 4).

We need to get (5, NULL, 4) NOT IN (5, NULL, NULL) = 0. As we know, No two NULL values could be compared. Hence we can only check for value $T_1 = 5$. Since it is present in the set of inner query output. Therefore number of tuples given as output is equal to zero.

47. (c)

The query creates a virtual table Named as "Number" and select the Movie_id's which have the largest number of actors.

The inner query return the count * of all the cast members in movie, while the outer query will print all those Movie_id whose that employed the largest number of actors.

The second part of the query returns the attribute, Movie_id and title of the movie from the relation Number and Movies. Since there is the use of ORDER-BY clause over Movie_id.

48. (c)

The query groups the tuples based on their addresses and count (*) will have the number of rows for each address group. Having a clause will put the condition for the result set such that only those group of addresses which have count more one will be present in the result set. The whole data is ordered in descending order based on the count value. Hence the output of the query will be the list of addresses of each house that was sold more than once along with the count of the number of sales of those properties. The result of which is ordered by the number of sales in decreasing order.

49. (a)

50. (d)

Inner query: For a given company say "ABC", this returns all the average salary of all the employees of ABC.

Outer query: This return the name of the employees whose salary is greater than the average salary of his company (returned by inner query).

51. (b)

I. From clause only performs cross product.

II. The RA expression is equivalent to the SQL expression

52. (c)

53. (d)

1. $\rho(R_1, \pi_{sid}(\pi_{cid}(\sigma_{branch = 'CS'}(Course)) \bowtie Enrols))$

   $\rho(R_2, \pi_{sid}(\pi_{cid}(\sigma_{branch = 'IT'}(Course)) \bowtie Enrols))$

   $R_1 \cap R_2$

   Find the Sid who enrolled atleast one course of CS branch then find the Sid who enrolled atleast one course of IT branch. Then take inter-section both Sid.

2. $\{T \mid \exists T_1 \in enrols\ (\exists x \in courses\ (x.branch = 'CS' \wedge x.cid = T_1.cid) \wedge \exists T_2 \in Enrols\ (\exists y \in courses\ (y.branch = 'IT' \wedge y.cid = T_2.cid) \wedge T_2.sid = T_1.sid) \wedge T.sid = T_1.sid)\}$

   Find the Sid who enrolled atleast one course of CS branch then find the Sid who enrolled atleast one course of IT branch with same Sid. Then return Sid.

3. Select Sid
   From courses P, Enrols C
   where P.branch='CS' AND P.cid = C.cid
   AND EXISTS (Select Sid
   From courses P2, Enrol C2
   where P2.branch = 'IT' AND C2.sid = C.sid
   AND P2.cid = C2.cid)

   Find the Sid who enrolled atleast one course of CS branch then find the same Sid enrolled for atleast one course of IT branch and return it.

54. (a)

```
Select Eid              ⇐ employee age max than 30
From Emp E
where age > 30 and
      not Exists  (select Pid      ⇐ all the project id whose project name is database
                   From project P
                   where Pname = 'database' and
                         not exist (select Pid      ⇐ the P.id where Eid is not in work relation
                                    from works W
                                    where W.Eid = E.Eid
                                    and W.Pid = P.Pid))
```

55. (a)

Here the 'with' clause defines the temporary relation dept_total (dept_name, value) with column name department name and value.
Second with clause is used to define temporary relation dept_total_avg (value) with column name value which contain average marks of all department.
At last we select dept_name where marks in department greater than equal to average marks of all department.

56. (a)

We need to find all those courses that are taught in both the Third 2015 semester and First 2017 semester.

Q_1: It will first select the course_id of the course that is taught in First 2017 semester and then check whether it is also there in Third 2015 semester.

Q_2: It will first select the course_id of the course that is taught in First 2017 semester and check the membership for it in present in Third 2015 semester also.

57. (b)

S_1 : Retrieves Students who gets less marks than every Student of dept 'CS'.

S_2 : Retrieves Students who gets more marks than some Student of dept 'CS'.

58. (c)

Given SQL query produces:
Displays Teachers who got a grade greater than equals to 9 in all course in madeeasy.

59. (b)

This query counts the number of tuples with salary > minimum salary. All the tuples will have unique salary. Hence only one tuple with minimum salary. Therefore 999 will be the result of the query (999 records will have salary greater than minimum salary).

60. (a)

S_1 and S_2 are same as 'ANY' and 'IN' conditions will give the same result.

61. (a)

Q_1 : Retrieves *A* which are more than some *C*.

Q_2 :
Retrieves *A* which are more than some *C*.

Q_3 : Retrieves *A* which are more than every *C*.

62. (4)

Because inner query result empty ALL clause condition true for all records of outer Query.

63. (b)

It retrieves employee eid whose age more than 20 and work for every project name data science.

64. (b)

SQL query return the pid's of parts that are supplied by atleast five suppliers.
Inner query returns the count value and outer query checks that the count is greater than 4 or not. If it is greater the tuple will be printed.

65. (b)

In the SQL query inner query finds all budgets of Delhi, than outer query compare every tuple of project and if the budget greater than or equal to inner query budget otherwise reject and because of NOT SQL query gives the name of projects whose budget is less than every project in Delhi.

66. (5)

Select $T_2 \cdot$ sid, count (*)
FROM Enroll T_1, Student T_2
Where $T_2 \cdot$ age ≥ ALL (select age FROM Student where age > 25) GROUP BY $T_2 \cdot$ sid;
⇓
empty
empty relation is always true in ALL
Thus unique sid in student table i.e. 5.

67. (0)

Outer query results Select Professors who have Rating > ALL(inner query).
None of tuple will satisfied the condition.

68. (c)

The first subquery (which is not correlated with the outer query) selects all projects controlled by department 5, and the second subquery (which is correlated) selects all projects that the particular employee being considered works on.
If the set difference of the first subquery result MINUS (EXCEPT) the second subquery result is empty, it means that the employee works on all the projects and is therefore selected.

69. (c)

Q_1 : Fails if two or more white parts.
If two or more white parts then (= ALL) false for every records.
Q_2 : Retrieves Sid who not enrolled every white part.
So both Q_1 and Q_2 is incorrect.

70. (2)

The given query result "the sids of suppliers who supply some red part and some green part". So the out tuples are sid 1 and sid 3.

71. (b)

Q_1 : Results set of A values which are more than every value of S.
Q_2 : Results set of A values which are more than or equal to all values of S.
Q_3 : Results set of A values which are more than every values of S.

72. (c)

Both query results students who never got a grade above 4.0 but 1st query also give those students who never enrolled any course.

73. (d)

The inner query is returning column $T_2 y$ of all the tuples from the table formed by cross product $T_1 \times T_2$ where columns $T_1 y$ and $T_2 y$ are not equal.
Table T_1 has 3 tuples and T_2 has 2 tuples so $T_1 \times T_2$ will have $2 \times 3 = 6$ tuples.
In these 6 tuples, values of attributes ($T_1 u$, $T_2 y$) will be (1, 1) (1, 2) (2, 1) (2, 2) (3, 1) (3, 2). Out of these 6 pairs, 4 of them are not having equal ($T_1 y$, $T_2 y$ value. Since, the sub-query will return 4 tuples, the NOT EXISTS construct will return the false for each of the 3 tuples in T_1. So 0 tuples will be there in output.

74. (d)

1. $\rho(R_1, \pi_{sid}(\pi_{cid}(\sigma_{branch = 'CS'}(Course)) \bowtie Enrols))$
   $\rho(R_2, \pi_{sid}(\pi_{cid}(\sigma_{branch = 'IT'}(Course)) \bowtie Enrols))$
   $R_1 \cap R_2$

   Find the Sid who enrolled atleast one course of CS branch then find the Sid who enrolled atleast one course of IT branch. Then take inter-section both Sid.
2. $\{T \mid \exists T_1 \in enrols\ (\exists x \in courses\ (x.branch = 'CS' \wedge x.cid = T_1.cid) \wedge \exists T_2 \in Enrols\ (\exists y \in courses\ (y.branch = 'IT' \wedge y.cid = T_2.cid) \wedge T_2.sid = T_1.sid) \wedge T.sid = T_1.sid)\}$

   Find the Sid who enrolled atleast one course of CS branch then find the Sid who enrolled atleast one course of IT branch with same Sid. Then return Sid.
3. Select Sid
   From courses P, Enrols C
   where P.branch='CS' AND P.cid = C.cid
   AND EXISTS (Select Sid

   From courses P2, Enrol C2

   where P2.branch = 'IT' AND C2.sid = C.sid

   AND P2.cid = C2.cid)

   Find the Sid who enrolled atleast one course of CS branch then find the same Sid enrolled for atleast one course of IT branch and return it.

75. (b)

$\{A \mid \exists S \in Sailors\ \exists R \in Reserves\ (R.sid = S.sid \wedge R.bid = 210 \wedge A.sname = S.sname)\}$ gives the name of sailors who have reserved boat 210.

76. (b)

In option (b)

When Student e with all A grade, for enrollment tuples not having her roll number, LHS is false.

For enrollment tuples having her/his roll number, LHS is true, RHS also true so the implication is true for all e tuples.

When Student e with some non-S grades, for enrollment tuples not having her roll number, LHS is false.

For enrollment tuples having her roll number, LHS is true, but RHS is false for at least one tupple.

77. (c)

$\{t \mid \exists r \in student\ (r[ID] = t[ID]) \wedge (\forall u \in course\ (u[dept\_name] = \text{“CS”} \Rightarrow \exists s \in takes\ (t[ID] = s[ID] \wedge s[course\_id] = u[course\_id]))\}$ will results all students who have taken all courses offered in the CS department. Since we know that $P \Rightarrow Q \equiv \text{not } P \vee Q$, so option (b) is also true.

78. (c)

The given query cannot be expressed in RA, and TRC (only in extended RA and TRC we are able to use to the aggregate functions) but could be expressed in SQL.

79. (a)

In the given tuple relational query, the tuple t has two attributes X and Y with $t \cdot X$ and $t \cdot Y$ as 50 and 90 respectively. So, the tuples having both X and Y is nothing but all the tuples having $X = 50$ and $Y = 90$.

Hence, only statement-I is correct.

■■■■