1. Consider the following relational database schema consisting of the four relation schemas:

**passenger**( pid, pname, pgender, pcity)

**agency**( aid, aname, acity)

**flight**(fid, fdate, time, src, dest)

**booking**(pid, aid, fid, fdate)

Answer the following questions using relational algebra queries;

**Solution:**

|  |
| --- |
| **Relational algebra operators:**  σ – selection with conditions (It selects all tuples that satisfies the conditions. Shows entire table with respect to the structure)  Π – projection operator (It selects the attributes which are listed here)  ⨝ - natural join operator (Binary operator that join two relations on common attributes’ values)  -, ∪, and ∩ - set operators (difference, union and intersection) |

**Most of the following queries can be written in many different ways.**

**a) Get the complete details of all flights to New Delhi.**

σ *dest = “New Delhi”*(flight)

 -----------------------------------------------------------------------------------------------------

**b) Get the details about all flights from Chennai to New Delhi.**

σ *src = “Chennai” ^ dest = “New Delhi”* (flight)

-----------------------------------------------------------------------------------------------------

**c) Find only the flight numbers for passenger with pid 123 for flights to Chennai before 06/11/2020.**

Π*fid*(σ*pid = 123*(booking) ⨝ σ*dest = “Chennai” ^ fdate < 06/11/2020*(flight))

[**Hint:** ***Given conditions are pid, dest, and fdate. To get the flight id for a passenger given a pid, we have two tables flight and booking to be joined with necessary conditions. From the result, the flight id can be projected***]

 -----------------------------------------------------------------------------------------------------

**d) Find the passenger names for passengers who have bookings on at least one flight.**

Π*pname*(passenger ⨝ booking)

 -----------------------------------------------------------------------------------------------------

**e) Find the passenger names for those who do not have any bookings in any flights.**

Π*pname*((Π*pid*(passenger) - Π*pid*(booking)) ⨝ passenger)

[**Hint:** ***here applied a set difference operation. The set difference operation returns only pids that have no booking. The result is joined with passenger table to get the passenger names.***]

-----------------------------------------------------------------------------------------------------

**f) Find the agency names for agencies that located in the same city as passenger with passenger id 123.**

Π*aname*(agency ⨝*acity = pcity*(σ*pid = 123*(passenger)))

[**Hint:** ***we performed a theta join on equality conditions (equi join) here. This is done between details of passenger 123 and the agency table to get the valid records where the city values are same. From the results, aname is projected.***]

 -----------------------------------------------------------------------------------------------------

**g) Get the details of flights that are scheduled on both dates 01/12/2020 and 02/12/2020 at 16:00 hours.**

(σ*fdate = 01/12/2020 ^ time = 16:00*(flight)) ∩ (σ*fdate = 02/12/2020 ^ time = 16:00*(flight))

[**Hint:** ***the requirement is for flight details for both dates in common. Hence, set intersection is used between the temporary relations generated from application of various conditions.***]

-----------------------------------------------------------------------------------------------------

**h) Get the details of flights that are scheduled on either of the dates 01/12/2020 or 02/12/2020 or both at 16:00 hours.**

(σ*fdate = 01/12/2020 ^ time = 16:00*(flight)) ∪ (σ*fdate = 02/12/2020 ^ time = 16:00*(flight))

 -----------------------------------------------------------------------------------------------------

**i) Find the agency names for agencies who do not have any bookings for passenger with id 123.**

Π*aname*(agency ⨝ (Πaid(agency) – Πaid(σ*pid = 123*(booking)))

-----------------------------------------------------------------------------------------------------

**j) Find the details of all male passengers who are associated with Jet agency.**

Π*passengers.pid, pname, pcity*(σ*pgender = “Male” ^ aname = ‘Jet’*(passengers ⨝ booking ⨝ agency))

1. **Consider the following schema for institute library:** **(AKTU 2021-22)**

Student (RollNo, Name, Father\_ Name, Branch)

Book (ISBN, Title, Author, Publisher)

Issue (RollNo, ISBN, Date-of –Issue)

Write the following queries in SQL and relational algebra:

* List roll number and name of all students of the branch ‘CSE’.
* Find the name of student who has issued a book published by ‘ABC’ publisher.
* List title of all books and their authors issued to a student ‘RAM’.
* List title of all books issued on or before December 1, 2020.
* List all books published by publisher ‘ABC’.

**Ans:**

* **SQL**

SELECT RollNo, Name

FROM Student

WHERE Branch = ‘CSE’;

**RA**

Result ← **π** RollNo, Name (σ branch = ‘CSE’(Student))

* **SQL**

SELECT S.Name

FROM Student S, Issue I, Book B

WHERE S.RollNo = I.RollNo AND I.ISBN  = B.ISBN

AND B.Publisher = ‘ABC’;

**RA**

R ← ((Student \* Issue) \* Book)

Result ← **π** Name (σ Publisher = ‘ABC’(R))

* **SQL**

SELECT B.Title, B.Author

FROM Student S, Issue I, Book B

WHERE S.RollNo = I.RollNo **AND** I.ISBN  = B.ISBN

**AND** S.Name = ‘RAM’;

**RA**

R ← ((Student \* Issue) \* Book)

Result ← **π** Title, Author (σ Name = ‘RAM’(R))

* **SQL**

SELECT B.Title

FROM Issue I, Book B

WHERE I.ISBN  = B.ISBN

AND I.Date-of-Issue <= ’01-DEC-2020’;

**RA**

R ← (Book \* Issue)

Result ← **π** Title (σ Date-of-Issue <= ’01-DEC-202’(R))

* **SQL**

SELECT \* FROM Book WHERE Publisher = ‘ABC’;

Result ← σ Publisher = ‘ABC’(Book)

**EmployeeInfo Table:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **EmpID** | **EmpFname** | **EmpLname** | **Department** | **Project** | **Address** | **DOB** | **Gender** |
| 1 | Sanjay | Mehra | HR | P1 | Hyderabad(HYD) | 01/12/1976 | M |
| 2 | Ananya | Mishra | Admin | P2 | Delhi(DEL) | 02/05/1968 | F |
| 3 | Rohan | Diwan | Account | P3 | Mumbai(BOM) | 01/01/1980 | M |
| 4 | Sonia | Kulkarni | HR | P1 | Hyderabad(HYD) | 02/05/1992 | F |
| 5 | Ankit | Kapoor | Admin | P2 | Delhi(DEL) | 03/07/1994 | M |

**EmployeePosition Table:**

|  |  |  |  |
| --- | --- | --- | --- |
| **EmpID** | **EmpPosition** | **DateOfJoining** | **Salary** |
| 1 | Manager | 01/05/2022 | 500000 |
| 2 | Executive | 02/05/2022 | 75000 |
| 3 | Manager | 01/05/2022 | 90000 |
| 2 | Lead | 02/05/2022 | 85000 |
| 1 | Executive | 01/05/2022 | 300000 |
|  |  |  |  |

### Write a query to fetch the number of employees working in the department ‘HR’

SELECT COUNT(\*) FROM EmployeeInfo WHERE Department = 'HR';

**b) Write a query to find the names of employees that begin with ‘S’**

SELECT \* FROM EmployeeInfo WHERE EmpFname LIKE 'S%';

**c) Write a query to find all the employees whose salary is between 50000 to 100000.**

SELECT \* FROM EmployeePosition WHERE Salary BETWEEN '50000' AND '100000';

**d) Write a query to fetch the EmpFname from the EmployeeInfo table in upper case and use the ALIAS name as EmpName.**

SELECT UPPER(EmpFname) AS EmpName FROM EmployeeInfo;

**Write a query to retrieve the EmpFname and EmpLname in a single column as “FullName”. The first name and the last name must be separated with space.**

SELECT CONCAT(EmpFname, ' ', EmpLname) AS 'FullName' FROM EmployeeInfo;

**Write a query to fetch the EmpFname from the EmployeeInfo table in the upper case and use the ALIAS name as EmpName.**

SELECT UPPER(EmpFname) AS EmpName FROM EmployeeInfo;

**Write a query to fetch top N records.**

By using the TOP command in SQL Server:

|  |  |
| --- | --- |
| 1 | SELECT TOP N \* FROM EmployeePosition ORDER BY Salary DESC; |

By using the LIMIT command in MySQL:

|  |  |
| --- | --- |
| 1 | SELECT \* FROM EmpPosition ORDER BY Salary DESC LIMIT N; |

**Q10. Write a query to retrieve the EmpFname and EmpLname in a single column as “FullName”. The first name and the last name must be separated with space.**

|  |  |
| --- | --- |
| 1 | SELECT CONCAT(EmpFname, ' ', EmpLname) AS 'FullName' FROM EmployeeInfo; |

**Q11. Write a query find number of employees whose DOB is between 02/05/1970 to 31/12/1975 and are grouped according to gender**

**Write a query to fetch all the records from the EmployeeInfo table ordered by EmpLname in descending order and Department in the ascending order.**

To order the records in ascending and descnding order, you have to use the [ORDER BY statement in SQL](https://www.edureka.co/blog/order-by-in-sql).

|  |  |
| --- | --- |
| 1 | SELECT \* FROM EmployeeInfo ORDER BY EmpFname desc, Department asc; |

**Q13. Write a query to fetch details of employees whose EmpLname ends with an alphabet ‘A’ and contains five alphabets.**

To fetch details mathcing a certain value, you have to use the [LIKE operator in SQL](https://www.edureka.co/blog/like-in-sql/).

|  |  |
| --- | --- |
| 1 | SELECT \* FROM EmployeeInfo WHERE EmpLname LIKE '\_\_\_\_a'; |

**Q14. Write a query to fetch details of all employees excluding the employees with first names, “Sanjay” and “Sonia” from the EmployeeInfo table.**

|  |  |
| --- | --- |
| 1 | SELECT \* FROM EmployeeInfo WHERE EmpFname NOT IN ('Sanjay','Sonia'); |

**Write a query to fetch details of employees with the address as “DELHI(DEL)”.**

|  |  |
| --- | --- |
| 1 | SELECT \* FROM EmployeeInfo WHERE Address LIKE 'DELHI(DEL)%'; |

**Q16. Write a query to fetch all employees who also hold the managerial position.**

|  |  |
| --- | --- |
| 1  2  3 | SELECT E.EmpFname, E.EmpLname, P.EmpPosition  FROM EmployeeInfo E INNER JOIN EmployeePosition P ON  E.EmpID = P.EmpID AND P.EmpPosition IN ('Manager'); |

**Q17.** **Write a query to fetch the department-wise count of employees sorted by department’s count in ascending order.**

|  |  |
| --- | --- |
| 1  2  3 | SELECT Department, count(EmpID) AS EmpDeptCount  FROM EmployeeInfo GROUP BY Department  ORDER BY EmpDeptCount ASC; |

**Q18. Write a query to calculate the even and odd records from a table.**

To retrieve the even records from a table, you have to use the MOD() function as follows:

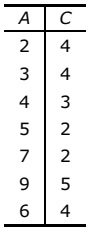
|  |  |
| --- | --- |
| 1 | SELECT EmpID FROM (SELECT rowno, EmpID from EmployeeInfo) WHERE MOD(rowno,2)=0; |

Similarly, to retrieve the odd records from a table, you can write a query as follows:

|  |  |
| --- | --- |
| 1 | SELECT EmpID FROM (SELECT rowno, EmpID from EmployeeInfo) WHERE MOD(rowno,2)=1; |

**Q19.** **Write a SQL query to retrieve employee details from EmployeeInfo table who have a date of joining in the EmployeePosition table.**

|  |  |
| --- | --- |
| 1  2  3 | SELECT \* FROM EmployeeInfo E  WHERE EXISTS  (SELECT \* FROM EmployeePosition P WHERE E.EmpId = P.EmpId); |



The following table has two attributes A and C where A is the primary key and C is the foreign key referencing A with on-delete cascade.

A C

-----

2 4

3 4

4 3

5 2

7 2

9 5

6 4

The set of all tuples that must be additionally deleted to preserve referential integrity when the tuple (2,4) is deleted is:  
**(A)** (3,4) and (6,4)  
**(B)** (5,2) and (7,2)  
**(C)** (5,2), (7,2) and (9,5)  
**(D)** (3,4), (4,3) and (6,4)Top of Form

Bottom of Form

Top of Form

here A is behaving like parent and C is  behaving like child bcoz A is PK and C is FK referencing to A.when (2,4) is deleted then it force to delete all entries in which C contains 2 bcoz there is no 2 remains in parent so child not able to access 2. so delete (5,2) (7,2) which force to delete all entries in which C contains 5or7 bcoz there is no 5 or 7 remain in parent so child not able to access 5 or 7.so delete (9,5) which force to delete all entries in which C contains 9 .

so ans should be **C**

Bottom of Form

C references A. this means C will only have values that A consists or null. so if you delete (2,4) then every 2 in C need to be deleted and this goes on. until there is no cascading deletion. (5,3) (7,2) will be deleted, then 5 is removed from A, them again (9,5) will be removed from relation. there is no 7 in C so no relation on that value.  
  
Read more at: <https://edurev.in/question/2115514/The-following-table-has-two-attributes-A-and-C-where-A-is-the-primary-key-and-C-is-the-foreign-key-r>

Consider the given ER Diagram. Lodging is many to many relationships. Rent, Payment to be made by person Occupying different hotel room should be added as an attribute to which entity and why?

s