**Scenario 1:**

Employee Table:

|  |  |
| --- | --- |
| **id** | **employee\_name** |
| 1 | Homer Simpson |
| 2 | Ned Flanders |
| 3 | Barney Gumble |
| 4 | Clancy Wiggum |
| 5 | Moe Syzslak |

Department Table:

|  |
| --- |
| **department\_id** |
| 4 |
| 1 |
| 5 |
| 3 |
| NULL |

**Problem Statement:**

Do all Joins (Left, Right, Full outer and Inner join)

**Output:**

**Left Join:**

select \* from Employeetable E left join Department D on E.department\_id=D.department\_id;

A screenshot of a computer

Description automatically generated

**Right Join:**

select \* from Employeetable E Right join Department D on E.department\_id=D.department\_id;

A screenshot of a computer

Description automatically generated

**Full Outer Join:**

select \* from Employeetable E Full outer join Department D on E.department\_id=D.department\_id;

A screenshot of a computer

Description automatically generated

**Inner Join:**

select \* from Employeetable E inner join Department D on E.department\_id=D.department\_id;

A screenshot of a computer

Description automatically generated

**Scenario 2:**

**Salary Log Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **EMPLOYEE\_ID** | **NAME** | **SALARY** | **timestamp** |
| 100 | Jennifer | 4400 | 11-08-2022 |
| 100 | Jennifer | 4300 | 12-08-2022 |
| 101 | Michael | 13000 | 13-08-2022 |
| 101 | Michael | 13100 | 14-08-2022 |
| 101 | Michael | 13200 | 15-08-2022 |
| 102 | Pat | 6000 | 16-08-2022 |
| 102 | Pat | 6100 | 17-08-2022 |
| 103 | Den | 11000 | 18-08-2022 |

**Expected Output:**

|  |  |  |  |
| --- | --- | --- | --- |
| **EMPLOYEE\_ID** | **NAME** | **SALARY** | **timestamp** |
| 100 | Jennifer | 4300 | 12-08-2022 |
| 101 | Michael | 13200 | 15-08-2022 |
| 102 | Pat | 6100 | 17-08-2022 |
| 103 | Den | 11000 | 18-08-2022 |

**Observation:**

From above output table It is clear that to fetch latest log (timestamp) of each employee.

**Solution Query:**

select Employee\_Id,EmpName, salary ,timestamp from

( select Employee\_Id,EmpName, salary ,timestamp,

Row\_Number() over(partition by Employee\_Id order by timestamp desc) as rankOrder

from EmployeeSalaryLog order by employee\_id)temp

where rankOrder=1;

**Output:**

A screenshot of a computer

Description automatically generated

**Scenario 3:**

**Problem 1:**

Table A

|  |
| --- |
| TableA |
| ID |
| 1 |
| 2 |
| 3 |

Table B

|  |
| --- |
| TableB |
| ID |
| 3 |
| 4 |
| 5 |

**Problem Statement:**

By using any kind of JOIN -  fetch the data available in Table A but NOT in Table B

**solution Query 1:**

select A.id from A left join B on B.id=A.id where B.id is null; # solution 1

**solution Query 2:**

select A.id from A left join B on B.id=A.id minus select A.id from A join B on B.id=A.id ;

**Output:**

A screenshot of a computer

Description automatically generated

**Problem 2:**

Table ProjectA:

|  |  |
| --- | --- |
| Project | Project Cost |
| A | 5000 |
| B | 2000 |
| C | 2000 |
| A | 1000 |
| B | 3000 |
| C | 7000 |
| D | 6500 |

**Problem Statement:**

Project Names and their total cost whose total cost is >= 6000

**Solution Query:**

select Project,Sum(Project\_cost) as Total\_cost from projectA group by Project

having Sum(Project\_cost) >=6000 order by Project;

**Output:**

A screenshot of a computer

Description automatically generated

**Problem 3:**

|  |  |  |
| --- | --- | --- |
| Empid | EmpName | MgrId |
| E001 | Jennifer | E002 |
| E002 | Michael | E003 |
| E003 | John | NULL |
| E004 | Den | E003 |

**Problem Statement:**

Employee and their Manager Name, one who dont have manager should populate NULL

**Solution Query:**

select \* from

(select A.EmpName, B.EmpName as MgrName from Employee\_Manager A

left join Employee\_Manager B on A.MgrId=B.EmpId);

**Output:**

A screenshot of a computer

Description automatically generated

**ASSIGNMENT – 2**

**Scenario 5:**

**Input table:**

|  |  |  |
| --- | --- | --- |
| session start | session end | session type |
| 11-08-2022 | 12-08-2022 | streamer |
| 13-08-2022 | 14-08-2022 | streamer |
| 16-08-2022 | 18-08-2022 | viewer |
| 19-08-2022 | 22-08-2022 | streamer |
| 26-08-2022 | 28-08-2022 | viewer |
| 29-08-2022 | 31-08-2022 | streamer |

**Expected Output:**

|  |  |  |  |
| --- | --- | --- | --- |
| session start | session end | session type | rank |
| 11-08-2022 | 12-08-2022 | streamer | 1 |
| 13-08-2022 | 14-08-2022 | streamer | 2 |
| 19-08-2022 | 22-08-2022 | streamer | 3 |
| 29-08-2022 | 31-08-2022 | streamer | 4 |
| 26-08-2022 | 28-08-2022 | viewer | 1 |
| 16-08-2022 | 18-08-2022 | viewer | 2 |

**Solution Query:**

select session\_start, session\_end, session\_type,

rank() over(partition by session\_type order by

case when session\_type='streamer' then session\_start end asc,

case when session\_type='viewer' then session\_start end desc ) as rank

from ChannelLog ;

**Output:**

A screenshot of a computer

Description automatically generated

**Scenario: 6**

**Input table :**

|  |  |  |
| --- | --- | --- |
| session start | session end | session type |
| 11-08-2022 | 12-08-2022 | streamer |
| 13-08-2022 | 14-08-2022 | streamer |
| 16-09-2022 | 18-09-2022 | viewer |
| 19-08-2022 | 22-08-2022 | streamer |
| 26-09-2022 | 28-09-2022 | viewer |
| 29-08-2022 | 31-08-2022 | streamer |

**Expected Output 1:**

|  |  |  |
| --- | --- | --- |
| session start | session end | session type |
| 26-09-2022 | 28-09-2022 | viewer |
| 29-08-2022 | 31-08-2022 | streamer |

**Solution Query:**

select \* from ChannelLog2 where session\_start in

(select max(session\_start) from ChannelLog2 group by Session\_type) ;

**Output:**

A screenshot of a computer

Description automatically generated

**Expected Output 2:**

|  |  |  |  |
| --- | --- | --- | --- |
| session start | session end | session type | priority |
| 11-08-2022 | 12-08-2022 | streamer | 1 |
| 13-08-2022 | 14-08-2022 | streamer | 2 |
| 16-09-2022 | 18-09-2022 | viewer | 5 |
| 19-08-2022 | 22-08-2022 | streamer | 3 |
| 26-09-2022 | 28-09-2022 | viewer | 6 |
| 29-08-2022 | 31-08-2022 | streamer | 4 |

**Solution Query :**

select session\_start, session\_end, session\_type,

row\_number() over(order by session\_start) as priority

from Channellog2;

**Output:**

A screenshot of a computer

Description automatically generated

**Scenario 4:**

**Input table:**

|  |  |  |  |
| --- | --- | --- | --- |
| policy | policy code | sequence | amount |
| 100 | 10 | 1 | 1000 |
| 101 | 10 | 2 | 4000 |
| 100 | 10 | 3 | 2000 |
| 101 | 20 | 1 | 5000 |
| 102 | 20 | 1 | 6000 |
| 102 | 20 | 2 | 7000 |
| 100 | 10 | 2 | 8000 |
| 101 | 10 | 1 | 9000 |
| 100 | 20 | 4 | 10000 |
| 102 | 10 | 3 | 11000 |
| 101 | 20 | 2 | 12000 |

**Expected Output:**

|  |  |  |  |
| --- | --- | --- | --- |
| policy | policy code | sequence | amount |
| 100 | 10 | 1 | 1000 |
| 100 | 20 | 4 | 10000 |
| 101 | 10 | 1 | 9000 |
| 101 | 20 | 1 | 5000 |
| 102 | 10 | 3 | 11000 |
| 102 | 20 | 1 | 6000 |

**Solution Query:**

select policy, policy\_code, sequence, amount from

(select policy, policy\_code, sequence, amount,

rank() over(partition by policy,policy\_code order by sequence) as rank1

from Policy )temp where rank1=1;

**Output:**

A screenshot of a computer

Description automatically generated