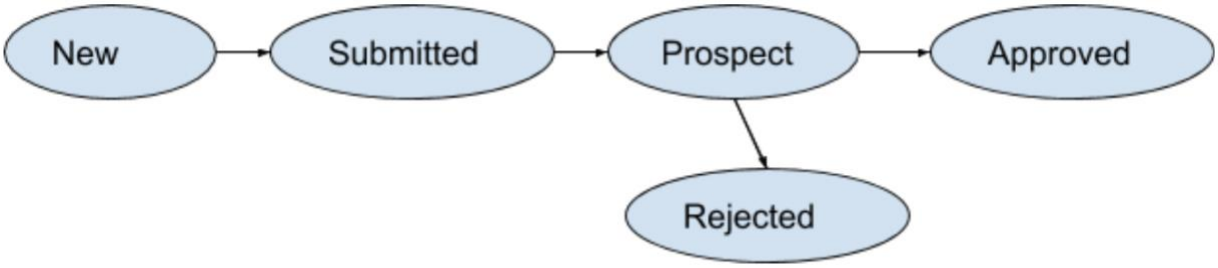
An object's state transition diagram is like this



Given a table with this schema that stores the above

ProjectID (Number) PK
ProjectName (string)
State (Number) PK
StateEntryTime (DateTime)

A given project may have gone through multiple states and as a result have multiple states stored with a different StateEntryTime.

Q1) Write a query to get the following
How many projects in each of the above states at its current final state.

New -- ??
Submitted -- ??
Prospect --??
Approved -- ??
Rejected -- ??

ANSWER:

// Pick the item with latest StateEntryTime for each ProjectID

```
SELECT State,COUNT(T1.ProjectID) as StateCount
FROM MOCK_DATA T1
INNER JOIN (
  SELECT ProjectID, MAX(StateEntryTime) as MaxDate
  FROM MOCK_DATA
  GROUP BY ProjectID
) T2 ON T1.StateEntryTime = T2.MaxDate
GROUP BY State
```

```
;
```

// Can further add this ORDER BY with CASE clauses to produce State in order

```
ORDER BY
  CASE when State="Rejected" Then 5 END,
  CASE when State="Approved" Then 4 END,
  CASE when State="Prospect" Then 3 END,
  CASE when State="Submitted" Then 2 END,
  CASE when State="New" Then 1 END
;
```

OUTPUT:

State	StateCount
New	10
Submitted	3
Prospect	1
Approved	4
Rejected	2

Q2) How would you **simplify** the query if only Approved and Rejected are needed ?
Approved -- ??
Rejected -- ??

ANSWER:

```
SELECT State, COUNT(ProjectID) as Count
FROM MOCK_DATA
WHERE State="Approved" OR State="Rejected"
GROUP BY State
;
```

// If there are multiple distinct types its preferable to use IN clause

```
SELECT State, COUNT(ProjectID) as Count
FROM MOCK_DATA
WHERE State IN ("Rejected", "Approved")
```

```
GROUP BY State
;
```

OUTPUT:

State	Count
Approved	4
Rejected	2

Q3) Write a query to get the following result

ID	New	Submitted	Prospect	Approved	Rejected
1	2019-01-01	2019-01-11	2019-01-21	2019-01-31	
2	2019-03-03	2019-04-05	2019-04-06		2019-04-10
3	2019-04-04	2019-04-04	2019-04-05		
4	2019-05-18	2019-05-20			
5	2019-05-21				

ANSWER:

```
-- Using PIVOT function from SQL Server
SELECT
  ProjectID,
  isnull(CAST([New] AS varchar),''),
  isnull(CAST([Submitted] AS varchar),''),
  isnull(CAST([Prospect] AS varchar),''),
  isnull(CAST([Approved] AS varchar),''),
  isnull(CAST([Rejected] AS varchar),'')
FROM MOCK_DATA
PIVOT (
  MAX (StateEntryTime)
  FOR State
  IN (New, Submitted, Prospect, Approved, Rejected)
)
AS PIVOTTABLE
ORDER BY ProjectID
```

```
;
```

OUTPUT:

ProjectID	New	Submitted	Prospect	Approved	Rejected
1	1999-11-29	2017-04-17	1998-01-18		2003-04-11
2	2010-01-08	2003-07-22	2011-03-17	2009-02-08	
3	2004-11-10	2002-03-11	2002-08-09		
4	1999-10-08	2015-08-14			
5	1996-03-03				
6	1997-07-09	2002-07-10	2005-07-03		
7	2004-11-09	2012-07-23	2012-02-22		2007-01-20
8	2003-12-22	2003-06-21	2001-10-12	2000-10-23	

```
-- Using CASE statements from MySQL
SELECT
 ProjectID as ID,
 CASE when State="New" Then StateEntryTime ELSE "" END as New,
 CASE when State="Submitted" Then StateEntryTime ELSE "" END as Submitted,
 CASE when State="Prospect" Then StateEntryTime ELSE "" END as Prospect,
 CASE when State="Approved" Then StateEntryTime ELSE "" END as Approved,
 CASE when State="Rejected" Then StateEntryTime ELSE "" END as Rejected
FROM (
 SELECT ProjectID, State, StateEntryTime
 FROM MOCK_DATA
) as S_Count
;
```

OUTPUT:

ID	New	Submitted	Prospect	Approved	Rejected
1	1999-11-29				
1		2017-04-17			
1			1998-01-18		
1					2003-04-11

2	2010-01-08				
2		2003-07-22			
2			2011-03-17		
2				2009-02-08	

Q4) Given a table with 4 cols (Emp_id, emp_name, date_of_joining, dept_id)

a) write a query to show the number of employees in each department that have more than 50 employees sorted desc on biggest size of employees.

ANSWER:

```
SELECT dept_id, COUNT(Emp_id) AS "Employee Count"
FROM table_name
GROUP BY dept_id
HAVING COUNT(dept_id) > 50
ORDER BY COUNT(dept_id) DESC;
```

OUTPUT:

dept_id	Employee Count
3	18
4	10
2	10
1	8

// In output above COUNT(dept_id) > 5 was used

b) Generate a serial number for this result set as part of the query

ANSWER:

// Using variables in MySQL

```
SET @row_number = 0;
SELECT
  (@row_number:=@row_number + 1) AS `Serial #`,
  dept_id AS `Department ID`,
  `Employee Count`
```

```
FROM (
  SELECT dept_id, COUNT(Emp_id) AS `Employee Count`
  FROM MOCK_DATA_2
  GROUP BY dept_id
  HAVING COUNT(dept_id) > 50
  ORDER BY COUNT(dept_id) DESC
  ) as R
```

OUTPUT:

Serial #	Department ID	Employee Count
3	3	18
4	2	10
2	4	10
5	1	8
1	5	4

// Using row_number() function in SQL Server

```
-- Using SQL Server
SELECT
  ROW_NUMBER() OVER(ORDER BY dept_id DESC) AS Serial_No,
  dept_id AS DepartmentID,
  EmployeeCount
FROM (
  SELECT dept_id, COUNT(Emp_id) AS EmployeeCount
  FROM MOCK_DATA_2
  GROUP BY dept_id
  HAVING COUNT(dept_id) > 2
  ) as R
ORDER BY EmployeeCount DESC
;
```

OUTPUT:

Serial_No	DepartmentID	EmployeeCount

3	3	18
4	2	10
2	4	10
5	1	8
1	5	4

c) The records now have a **repeated field called Project, Role** where both are enums. Enhance the query to find only those departments where number of managers is more than 10.

For example, Employee Yvonne, ITdept, SalesProject:Developer, OrderProject:Analyst, QuotingProject:Manager.

Table Emp:

- Emp_id,
- name,
- date_of_joining,
- Dept_id,
- ProjectRoleEnum
  - Project,
  - Role

ANSWER:

// Video for BigQuery Nested and Repeated Fields
// https://www.youtube.com/watch?v=STo98QUKDS8
// https://cloud.google.com/bigquery/docs/legacy-nested-repeated

Given Table Example:

Row	Emp_id	name	date_of_joining	Dept_id	ProjectRoleEnum.Project	ProjectRoleEnum.Role
1	1	Yvonne	2000-12-01	123	Sales	Develepor
					Order	Analyst
					Quoting	Manager

After Unnesting:

Row	Emp_id	name	date_of_joining	Dept_id	ProjectRoleEnum.Project	ProjectRoleEnum.Role
1	1	Yvonne	2000-12-01	123	Sales	Develepor
2	1	Yvonne	2000-12-01	123	Order	Analyst
3	1	Yvonne	2000-12-01	123	Quoting	Manager

```
SELECT Dept_id, Count(*) AS ManagerCount
FROM (
  SELECT Dept_id, Role
  FROM MOCK_DATA_4C
```

```
   CROSS JOIN UNNEST(ProjectRoleEnum) as ProjectRoleEnum
)
WHERE Role = "Manager"
GROUP BY Dept_id
Having Count(*) > 10
;
```

OUTPUT:

Dept_id	ManagerCount
10	24
7	20
3	19
4	18
11	12

// Above output is NOT tested on BigQuery

Q5) Given two tables:
Table 1: Projects (Project_ID, Manager, Project_Name, Status)
Table 2: ProjectsRisks (Project_ID, Risk_Name, Date)

Write a query that lists each manager with their latest risk per project.

ANSWER:

Assumptions:

- A manager can have several projects under him.
- The solution picks the latest RISK_NAME for EACH Manager.
- Only one project is picked for every manager which has the latest RISK date.

```
SELECT Manager, Project_Name, Risk_Name, MAX(Date) as Latest_Risk
FROM ProjectsRisks sub2
INNER JOIN Projects ON sub2.Project_ID = Projects.Project_ID
GROUP BY Manager
ORDER BY Date DESC
;
```

OUTPUTS:

Manager	Project_Name	Risk_Name	Latest_Risk
Tally Ismail	Holdlamis	Black and white colobus	2015-08-31

Danit Fausch	Transcof	Squirrel, smith's bush	2014-01-19
Dianna Stanistrete	Namfix	Southern ground hornbill	2015-11-11
Weider McCloughlin	Bitchip	Stanley crane	2016-11-28
Julian Duferie	Fixflex	Crab, sally lightfoot	2000-12-17