### SQL JOIN OBSERVATIONS ON CHANGING JOINING CONDITION ###
### WE ALL KNOW WHAT HAPPENS IN JOIN WHEN WE USE
### CONDITIONS LIKE T1 JOIN T2 ON T1.ID=T2.ID ###
### DO WE KNOW WHAT WILL HAPPEN IF WE USE T1.ID=1 ###

### LET 'S LEARN ###

```
create table student(s_id int primary key,
                      s_name varchar(20));

insert into student values(1, 'Jack');
insert into student values(2, 'Rithvik');
insert into student values(3, 'Jaspreet');
insert into student values(4, 'Praveen');
insert into student values(5, 'Bisa');
insert into student values(6, 'Suraj');

/*
    s_id    s_name
1   1   Jack
2   2   Rithvik
3   3   Jaspreet
4   4   Praveen
5   5   Bisa
6   6   Suraj
*/


select * from student
```

```
create table marks(school_id int primary key, s_id int,
                       score int, status varchar(20));

insert into marks values(1004, 1, 23, 'fail');
insert into marks values(1008, 6, 95, 'pass');
insert into marks values(1012, 2, 97, 'pass');
insert into marks values(1016, 7, 67, 'pass');
insert into marks values(1020, 3, 100, 'pass');
insert into marks values(1025, 8, 73, 'pass');
insert into marks values(1030, 4, 88, 'pass');
insert into marks values(1035, 9,  13, 'fail');
insert into marks values(1040, 5,  16, 'fail');
insert into marks values(1050, 10, 53, 'pass');
select * from marks

/*
    school_id   s_id    score   status
1   1004    1   23  fail
2   1008    6   95  pass
3   1012    2   97  pass
4   1016    7   67  pass
5   1020    3   100 pass
6   1025    8   73  pass
7   1030    4   88  pass
8   1035    9   13  fail
9   1040    5   16  fail
10  1050    10  53  pass
*/

select s.s_id, m.score from student s inner join marks m on s.s_id=m.s_id
```

```
/*
    s_id    score
1   1   23
2   6   95
3   2   97
4   3   100
5   4   88
6   5   16
*/
/*
what is happening here when we put some value in condition while joining?
Let's remember 6 rows available in students table and
10 rows available in marks table.
*/
select s.s_id, m.score from student s inner join marks m on s.s_id=1

/*
    s_id    score
1   1   23
2   1   95
3   1   97
4   1   67
5   1   100
6   1   73
7   1   88
8   1   13
9   1   16
10  1   53

Here we have taken s.s_id=1 that means s_id column
from student table in condition section.
we can see the the output:same no of rows as marks table.
It actually repeats same value which is mentioned
in the condition.

What if value which was taken does not available
in student columns.(Assume s_id=100)?
*/

select s.s_id, m.score from student s inner join marks m on s.s_id=100

/*
Shows No Result
It does not return any row.
*/

select s.s_id, m.score from student s inner join marks m on m.s_id=1

/*
    s_id    score
1   1   23
2   2   23
3   3   23
4   4   23
5   5   23
6   6   23

Here we have taken m.s_id=1 that means s.s_id column from marks table.
Observation:
Same no of output(6 rows) as student table(6 rows).
```

```
Repeats 23.As we see 23 is score for s_id=1

*/

select s.s_id, m.score from student s inner join marks m on m.s_id=100
/*
No Output
As condition value does not avilable in marks table.
*/
```

***
### Let's assume we have a table Process ###
### where there are two columns###

```
Process_Name,Start_Time.Now You are told the End_Time for a partucular
Process can be derived by next Process's Start_Time.It means Process_Name:P1
End_Time will be Process_Name:P2 Start_Time.
Process_Name are not ordered/sorted as well
***

1.Create another table from the given statement and table which includes
End_Time column.
2.Find out the Process_Name which takes maximum time for execution.

***

create table Process(Process_Name varchar(5) primary key,
                      Start_Time int);

insert into Process values('P1', 10);
insert into Process values('P2', 12);
insert into Process values('P3', 13);
insert into Process values('P4', 18);
insert into Process values('P7', 24);
insert into Process values('P6', 20);
insert into Process values('P5', 19);
insert into Process values('P8', 26);
/* Question 1*/

select Process_Name,Start_Time, lead(Start_Time,1)
over (order by Process_Name) End_Time from process;

/* Question 2*/

Select * from
(select Process_Name,(End_Time-Start_Time) as Diff from
(select Process_Name,Start_Time, lead(Start_Time,1)
 over (order by Process_Name) End_Time  from process) n
) m
where Diff = ( select max(End_Time-Start_Time) from
(select Process_Name,Start_Time, lead(Start_Time,1)
 over (order by Process_Name) End_Time  from process) n )
```

```
/*
***
```

### Find out Latest transcation from Each acct_id? ###
### I have a table "transactions" that has columns "acct_id" "trans_date" and "trans_type"
### and I want to filter this table so that I have just the last transaction for each account.
### Clearly I could do something like

```
SELECT acct_id, max(trans_date) as trans_date
FROM transactions GROUP BY acct_id;
```

### but then I lose my trans_type

```
Let's try
***
*/

create table transactions(acct_id varchar(5) ,
                      trans_date timestamp,trans_type varchar(10));

insert into transactions values('A1', DEFAULT,'ATM');
insert into transactions values('A1',DEFAULT,'Check');
insert into transactions values('A3',DEFAULT,'Deposit'); /*Latest*/
insert into transactions values('A4',DEFAULT,'Deposit');
insert into transactions values('A2',DEFAULT,'Check');
insert into transactions values('A3',DEFAULT,'Check');

Select * from transactions
/*
    acct_id trans_date  trans_type
1   A1  000005518443    ATM
2   A1  000005518444    Check
3   A3  000005518445    Deposit
4   A4  000005518446    Deposit
5   A2  000005518447    Check
6   A3  000005518448    Check
*/

/* Method 1 */

SELECT t1.*
FROM transactions t1
LEFT OUTER JOIN transactions t2
  ON (t1.acct_id = t2.acct_id AND t1.trans_date < t2.trans_date)
WHERE t2.acct_id IS NULL;

/*
    acct_id trans_date  trans_type
1   A1  000005518444    Check
2   A4  000005518446    Deposit
3   A2  000005518447    Check
4   A3  000005518448    Check
*/

/* Method 2 */

SELECT acct_id, trans_date, trans_type
FROM transactions a
WHERE trans_date = (
   SELECT MAX( trans_date )
   FROM transactions b
   WHERE a.acct_id = b.acct_id
)
```

```
/*
    acct_id trans_date  trans_type
1   A4  000005518446    Deposit
2   A3  000005518448    Check
3   A2  000005518447    Check
4   A1  000005518444    Check
*/
```

```
Let's discuss about the below table

    S_id Name  Math English Science History Geography
1   1    Modi   70  80       76      60      78
2   2    Amit   80  70       86      68      58
3   3    Rahul  75  85       71      69      68
4   4    Sonia  78  82       66      69      75
5   5    Sambit 60  84       72      64      71
6   6    Arnab  78  85       66      69      65
```

### Find out Maximum, Minimum, Average Marks obtained by each student and ### put them into three separate columns

```
*/

create table Students(S_id int Primary Key,Name varchar(20),
               Math int, English int, Science int,History int,Geography int);

insert into Students values(1, 'Modi',70,80,76,60,78);
insert into Students values(2, 'Amit',80,70,86,68,58);
insert into Students values(3, 'Rahul',75,85,71,69,68);
insert into Students values(4, 'Sonia',78,82,66,69,75);
insert into Students values(5, 'Sambit',60,84,72,64,71);
insert into Students values(6, 'Arnab',78,85,66,69,65);


Select * from Students

/*
    S_id    Name    Math    English Science History Geography
1   1   Modi    70  80  76  60  78
2   2   Amit    80  70  86  68  58
3   3   Rahul   75  85  71  69  68
4   4   Sonia   78  82  66  69  75
5   5   Sambit  60  84  72  64  71
6   6   Arnab   78  85  66  69  65
*/

SELECT S_id,Name,
CASE
    WHEN Math >= English and Math >= Science and Math >= History
    and Math >= Geography THEN Math
    WHEN English >= Math and English >= Science and English >= History
    and English >= Geography THEN English
    WHEN Science >= English and Science >= Math and Science >= History
    and Science >= Geography THEN Science
    WHEN History >= English and History >= Science and History >= Math
    and History >= Geography THEN History
    WHEN Geography >= English and Geography >= Science and
    Geography >= History and Geography >= Math THEN Geography
    ELSE 0
END AS Max_Marks,
(Math+English+Science+History+Geography)/5 as Avg_Marks
FROM Students;

/*
    S_id    Name    Max_Marks   Avg_Marks
1   1   Modi    80  72
2   2   Amit    86  72
3   3   Rahul   85  73
4   4   Sonia   82  74
5   5   Sambit  84  70
6   6   Arnab   85  72
*/
```

### Getting count of total present and absent from attendence ###
### in a single sql query ###

```
###attendance_table details Emp_ID(varchar),date(datetime),attendance(char(2))
attendence column consists A or P A for Absent and P for Present

select Emp_ID
      ,count(case when status ='A'  then 1 end) as absent_count
      ,count(case when status ='P' then 1 end) as present_count
      ,count(distinct date) as Tot_count
  from attendence_table where date between '2019-08-01' and '2014-08-31'
 group
    by Emp_ID ;
```

### Split Name column into three columns in SQL###
### First Name, Middle Name, Last Name###

```
create table Students(EmpName varchar(20));

insert into Students values('Tapas Kumar Pal');
insert into Students values('Tapas Ranjan Sinha');
insert into Students values('Mayuresh Kumar');
insert into Students values('Neetha B');
insert into Students values('PJ');


Select * from Students;

/*
    Name
1   Tapas Kumar Pal
2   Tapas Ranjan Sinha
3   Mayuresh Kumar
4   Neetha B
5   PJ
*/
Select Ltrim(SubString(EmpName,1,Isnull(Nullif(CHARINDEX(' ',EmpName),0),1000)))
As FirstName,
Ltrim(SUBSTRING(EmpName,CharIndex(' ',EmpName),
Case When (CHARINDEX(' ',EmpName,
                     CHARINDEX(' ',EmpName)+1)-CHARINDEX(' ',EmpName))<=0 then 0
else CHARINDEX(' ',EmpName,CHARINDEX(' ',EmpName)+1)-
```

```
                CHARINDEX(' ',EmpName) end )) as MiddleName,
Ltrim(SUBSTRING(EmpName,
                Isnull(Nullif(CHARINDEX(' ',EmpName,
                    Charindex(' ',EmpName)+1),0),CHARINDEX(' ',EmpName)),
Case when Charindex(' ',EmpName)=0 then 0 else LEN(EmpName) end)) as LastName
From Students

/*
    FirstName   MiddleName  LastName
1   Tapas       Kumar       Pal
2   Tapas       Ranjan      Sinha
3   Mayuresh    NULL        Kumar
4   Neetha      NULL        B
5   PJ
*/
```

### Binary Tree Node###

```
You are given a table, BST, containing two columns: N and P,
where N represents the value of a node in Binary Tree,
and P is the parent of N.


Write a query to find the node type of Binary Tree
ordered by the value of the node.
Output one of the following for each node:

Root: If node is root node.
Leaf: If node is leaf node.
Inner: If node is neither root nor leaf node.
Sample Input

Sample Input:
N  P
1  2
3  2
6  8
9  8
2  5
8  5
5 Null


Sample Output:

1 Leaf
2 Inner
3 Leaf
5 Root
6 Leaf
8 Inner
9 Leaf

<<<< Method 1


SELECT N, IF(P IS NULL,'Root',
IF((SELECT COUNT(*) FROM BST WHERE P=B.N)>0,'Inner','Leaf'))
FROM BST AS B ORDER BY N
```

```
<<<<Method 2

SELECT N,
CASE WHEN P IS NULL THEN 'Root'
WHEN N NOT IN
(SELECT DISTINCT P FROM BST WHERE P IS NOT NULL )
AND N IS NOT NULL THEN 'Leaf'
ELSE 'Inner'
END AS Type
FROM BST ORDER BY N;
```

### Delete/ Fetach Duplicate rows from Table in SQL

```
<<<<<<<<To Fetch Duplicate Rows>>>>>>>>

SELECT
name, email, COUNT(*)
FROM
users
GROUP BY
name, email
HAVING COUNT(*) > 1

<<<<<<To Delete the Duplicate Rows>>>>>>

DELETE users
WHERE rowid NOT IN
(SELECT MIN(rowid)
FROM users
GROUP BY name, email);
```

### Diffrence between EXISTS and IN in SQL
### Which is better

The reason is that the EXISTS operator works based on the “at least found” principle.
It returns true and stops scanning table once at least one matching row found.

On the other hands, when the IN operator is combined with a subquery,
MySQL must process the subquery first, and
then uses the result of the subquery to process the whole query.

****The general rule of thumb is that if the subquery contains a large volume of data,
the EXISTS operator provides a better performance.

However, the query that uses the IN operator will perform faster
if the result set returned from the subquery is very small.****

### Order of EXECUTION in SQL

```
FROM
ON
JOIN
WHERE
GROUP BY
WITH CUBE or WITH ROLLUP
HAVING
SELECT
DISTINCT
```

```
ORDER BY
TOP
```

### Select * OR Select ALL COLUMNS FROM TABLE ###
### WHICH ONE IS BETTER? ###

Select * is asking to pull everything from the table i.e.
high I/O whereas in Select with column name is just asking to pull what you need.
This will have low I/O compared to Select * if you are not pulling all the columns.
If you are pulling all the columns then
select * is not different than select with column names.

### Inner join returning more records than exists in table?###

Yes, this is possible.Duplicate value on joining key will give
more records than exitsing table.

#### Diffrence between RANK(), DENSE RANK() and ROW_NUMBER()?###

Row_Number() will generate a unique number for every row,
even if one or more rows has the same value.

RANK() will assign the same number for the row
which contains the same value and skips the next number.

DENSE_RANK () will assign the same number for the row
which contains the same value without skipping the next number.

### Which Index is faster? ###
### Clustered index or Non clustered index? ###