SQL Case Study:

1.Given below table Employee as input which has two columns 'Group' and 'Sequence', Write a SQL query to find the maximum and minimum value of continuous 'sequence' in each group.
Input Table:

Group1	Sequence
A	1
A	2
A	3
A	5
A	6
A	8
A	9
B	11
C	1
C	2
C	3
A	1
A	2
A	3
A	5
A	6
A	8
A	9
B	11
C	1

C	2
C	3

Output Table:

Group1	min_seq	max_seq
A	1	3
A	5	6
A	8	9
B	11	11
C	1	3

Data set :

```
CREATE TABLE Emp(
Group1  varchar(20),
Sequence  int );
INSERT INTO Emp VALUES('A',1);
INSERT INTO Emp VALUES('A',2);
INSERT INTO Emp VALUES('A',3);
INSERT INTO Emp VALUES('A',5);
INSERT INTO Emp VALUES('A',6);
INSERT INTO Emp VALUES('A',8);
INSERT INTO Emp VALUES('A',9);
INSERT INTO Emp VALUES('B',11);
INSERT INTO Emp VALUES('C',1);
INSERT INTO Emp VALUES('C',2);
INSERT INTO Emp VALUES('C',3);
select * from emp;
```

Answer :

```
select distinct b.group1,min(sequence) as min_seq,max(sequence) as max_seq from
```

```
(select a.*,sequence-row_num as grp_split from (select *,row_number() over(partition by
group1 order by group1) as row_num from emp) as a) as b
group by group1,grp_split
order by group1;
```

2.Student Table has three columns Student_name, Total_marks and Year. User has to write a SQL query to display Student_Name, Total_Marks, Year, Prev_yr_Marks for those whose Total_Marks are greater than or equal to the previous year.
Input Table:

Student_Name	Total_Marks	Year
Rahul	90	2010
Sanjay	80	2010
Mohan	70	2010
Rahul	90	2011
Sanjay	85	2011
Mohan	65	2011
Rahul	80	2012
Sanjay	80	2012
Mohan	90	2012

Output Table:

Student_Name	Year	Total_Marks	pre_year
Mohan	2012	90	65
Rahul	2011	90	90
Sanjay	2011	85	80

Dataset:

```
CREATE TABLE Student(
Student_Name  varchar(30),
Total_Marks  int ,
Year  int);
INSERT INTO Student VALUES('Rahul',90,2010);
INSERT INTO Student VALUES('Sanjay',80,2010);
INSERT INTO Student VALUES('Mohan',70,2010);
INSERT INTO Student VALUES('Rahul',90,2011);
INSERT INTO Student VALUES('Sanjay',85,2011);
INSERT INTO Student VALUES('Mohan',65,2011);
INSERT INTO Student VALUES('Rahul',80,2012);
INSERT INTO Student VALUES('Sanjay',80,2012);
INSERT INTO Student VALUES('Mohan',90,2012);
```

Result:

```
select a.* from (select Student_Name,Year,Total_Marks,lag(Total_Marks) over(partition by
Student_Name order by year) as pre_year from student)
as a where a.Total_Marks >= a.pre_year;
```

3.Transaction_tbl Table has four columns CustID,TransID,TransAmt, and TransDate. User has to display all these fields along with maximum TranAmt for each CustID and ratio of TranAmt and maximum TranAmt for each transaction.

Input Table:

CustID	TranID	TranAmt	TranDate
1001	20001	10000	2020-04-25
1001	20002	15000	2020-04-25
1001	20003	80000	2020-04-25
1001	20004	20000	2020-04-25
1002	30001	7000	2020-04-25
1002	30002	15000	2020-04-25
1002	30003	22000	2020-04-25

Output Table:

CustID	CustID	TranAmt	maxamount	ratio	TranDate
1001	1001	10000	80000	0.12	2020-04-25
1001	1001	15000	80000	0.19	2020-04-25
1001	1001	80000	80000	1	2020-04-25
1001	1001	20000	80000	0.25	2020-04-25
1002	1002	7000	22000	0.32	2020-04-25
1002	1002	15000	22000	0.68	2020-04-25
1002	1002	22000	22000	1	2020-04-25

Dataset:

```
CREATE TABLE TB1(
 CustID int ,
 TranID int ,
 TranAmt float ,
 TranDate date
) ;

INSERT INTO TB1 VALUES (1001, 20001, 10000, CAST('2020-04-25' AS Date));
INSERT INTO TB1  VALUES (1001, 20002, 15000, CAST('2020-04-25' AS Date));
INSERT INTO TB1  VALUES (1001, 20003, 80000, CAST('2020-04-25' AS Date));
INSERT INTO TB1  VALUES (1001, 20004, 20000, CAST('2020-04-25' AS Date));
INSERT INTO TB1  VALUES (1002, 30001, 7000, CAST('2020-04-25' AS Date));
INSERT INTO TB1  VALUES (1002, 30002, 15000, CAST('2020-04-25' AS Date));
INSERT INTO TB1  VALUES (1002, 30003, 22000, CAST('2020-04-25' AS Date));
SELECT * FROM TB1;
```

Result:

```
SELECT a.CustID,a.CustID,a.TranAmt,maxamount,round(a.TranAmt/maxamount,2) as ratio,a.TranDate from TB1 as a
inner join (select CustID,max(TranAmt) as maxamount from TB1  group by CustID) as b on
a.CustID = b.CustID;
```

4.Write a SQL to find all Employees who earn more than the average salary in their corresponding department.
Input Table:

EmpID	EmpName	Salary	DeptID
1001	Mark	60000	2
1002	Antony	40000	2
1003	Andrew	15000	1
1004	Peter	35000	1
1005	John	55000	1
1006	Albert	25000	3
1007	Donald	35000	3

Output Table:

EmpID	EmpName	Salary	DeptID	average
1001	Mark	60000	2	50000
1005	John	55000	1	35000
1007	Donald	35000	3	30000

Dataset:

```
CREATE TABLE Employee(
 EmpID int ,
 EmpName varchar(30) ,
 Salary Bigint ,
 DeptID int
```

);
INSERT into Employee VALUES (1001, 'Mark', 60000, 2);
INSERT into Employee VALUES (1002, 'Antony', 40000, 2);
INSERT into Employee VALUES (1003, 'Andrew', 15000,1);
INSERT into Employee VALUES (1004, 'Peter', 35000, 1);
INSERT into Employee VALUES (1005,'John', 55000, 1);
INSERT into Employee VALUES (1006, 'Albert', 25000, 3);
INSERT into Employee VALUES (1007,'Donald', 35000, 3);

Result:

```
select a.*,b.average from Employee as a
inner join
(select Distinct DeptId,avg(salary) as average from Employee group by DeptID order by DeptID) as b
on
a.DeptID = B.DeptID
where salary > average;
```

5.Write SQL to display total number of matches played, matches won, matches tied and matches lost for each team.

Input Table:

id	Team1	Team2	Result
1	India	Australia	India
2	India	England	England
3	SouthAfrica	India	India
4	Australia	England	**null**
5	England	SouthAfrica	SouthAfrica
6	Australia	India	Australia

Output table:

team	matche_played	match_won	match_tie	match_lost
India	4	2	0	2

SouthAfrica	2	1	0	1
Australia	3	1	1	1
England	3	1	1	1

Dataset:

```
create table Match_result
(
id int,
Team1 varchar(100),
Team2 Varchar(100),
Result Varchar(100)
);

insert into Match_result Values (1,'India','Australia','India');
insert into Match_result Values (2,'India','England','England');
insert into Match_result Values (3,'SouthAfrica','India','India');
insert into Match_result Values (4,'Australia','England',NULL);
insert into Match_result Values (5,'England','SouthAfrica','SouthAfrica');
insert into Match_result Values (6,'Australia','India','Australia');
```

Result:

```
select ax.team,ax.tot_match,bx.match_won,ifnull(cx.match_tie,0) as match_tie,ax.tot_match -
bx.match_won as match_lost
  from (select a.team,sum(tot) as tot_match
from (select team1 as team, count(*) as tot from match_result group by team1
union all
select team2 as team, count(*) as tot from match_result group by team2) as a
group by a.team) as ax
left join
(select result,count(*) as match_won from match_result  where result is not null group by result) as
bx
on
ax.team = bx.result
left join
(select team1 as team,count(*) as match_tie from match_result where Result is NULL
union all
```

```
select team2 as team,count(*) as match_tie from match_result where result is NULL) as cx
on ax.team = cx.team;
```

1.Compare each Employee salary to the average salary in the company?
2.Find the percentage of salary that each employee is receiving in their respective department?
3.Find the highest paid employee in each department ?
4.Find the second lowest paid employee in each department?

Input Table :

Employee_2:

EmpID	EmpName	DeptID
101	Mark	12
102	John	13
103	Smith	15
104	Donald	12
105	James	15
106	Maria	15
107	David	12
108	Martin	13
109	Robert	16
110	Michael	16
111	Rajesh	12
112	Ravi	12

dept_details

DeptID	DeptName
11	**HR**
12	**Developer**

13	Tester
14	Analyst
15	Sales specialist
16	Admin manager

salary_details:

DeptName	Salary
HR	25000
Developer	40000
Tester	30000
Analyst	45000
Sales specialist	50000
Admin manager	27000

Dataset :

```
create table employee_11(
EmpID integer primary key,
EmpName varchar(20),
DeptID integer );

create table dept_details(
DeptID integer primary key,
DeptName Varchar(20)
);
create table salary_details(
DeptName varchar(20),
salary integer
) ;
insert into employee_11 values(101,'Mark',12);
insert into employee_11 values(102,'John',13);
insert into employee_11 values(103,'Smith',15);
```

```
insert into employee_11 values(104,'Donald',12);
insert into employee_11 values(105,'James',15);
insert into employee_11 values(106,'Maria',15);
insert into employee_11 values(107,'David',12);
insert into employee_11 values(108,'Martin',13);
insert into employee_11 values(109,'Robert',16);
insert into employee_11 values(110,'Michael',16);
insert into employee_11 values(111,'Rajesh',12);
insert into employee_11 values(112,'Ravi',12);

insert into salary_details values('HR',25000);
insert into salary_details values('Developer',40000);
insert into salary_details values('Tester',30000);
insert into salary_details values('Analyst',45000);
insert into salary_details values('Sales specialist',50000);
insert into salary_details values('Admin manager',27000);

insert into dept_details values(11,'HR');
insert into dept_details values(12,'Developer');
insert into dept_details values(13,'Tester');
insert into dept_details values(14,'Analyst');
insert into dept_details values(15,'Sales specialist');
insert into dept_details values(16,'Admin manager');
```

1.**Input** : Employee Table has four columns namely EmpName, DeptName, DeptNo and Salary
Problem Statement : Write a SQL query to get the output as shown in the Output tables ?

Input Table :

EmpName	DeptName	DeptNo	Salary

Mark	HR	101	30000
John	Accountant	101	20000
Smith	Analyst	101	25000
Donald	HR	201	40000
James	Analyst	201	22000
Maria	Analyst	201	38000
David	Manager	201	33000
Martin	Analyst	301	22000
Robert	Analyst	301	56000
Michael	Manager	301	34000
Robert	Accountant	301	37000
Michael	Analyst	301	28000

Output Table:

EmpName	DeptName	DeptNo	Salary	rank1
Mark	HR	101	30000	1
Donald	HR	201	40000	1
Robert	Analyst	301	56000	1
John	Accountant	101	20000	1
James	Analyst	201	22000	1
Martin	Analyst	301	22000	1

Dataset :

```
Create Table Employee_2
(EmpName Varchar(30),
DeptName Varchar(25),
DeptNo Bigint,
```

```
Salary Bigint);


Insert into Employee_2 Values('Mark','HR',101,30000);
Insert into Employee_2 Values('John','Accountant',101,20000);
Insert into Employee_2 Values('Smith','Analyst',101,25000);
Insert into Employee_2 Values('Donald','HR',201,40000);
Insert into Employee_2 Values('James','Analyst',201,22000);
Insert into Employee_2 Values('Maria','Analyst',201,38000);
Insert into Employee_2 Values('David','Manager',201,33000);
Insert into Employee_2 Values('Martin','Analyst',301,22000);
Insert into Employee_2 Values('Robert','Analyst',301,56000);
Insert into Employee_2 Values('Michael','Manager',301,34000);
Insert into Employee_2 Values('Robert','Accountant',301,37000);
Insert into Employee_2 Values('Michael','Analyst',301,28000);
```

Result :

```
create view emp1 as  select *,dense_rank() over(Partition by DeptNo order by
Salary desc) as rank1 from emp;
select * from emp1 where rank1 = 1;
create view emp2 as  select *,dense_rank() over(Partition by DeptNo order by
Salary) as rank1 from emp;
select * from emp2 where rank1 = 1;
select * from emp1 where rank1 = 1
union
select * from emp2 where rank1 = 1;
```

2.**Problem Statement** :- Write a SQL query to print the desired Output as shown below using two tables i.e:- Emp_Table and Month_Table

Input table:

Emp_Table :

SerialNo	Name	Month_ID	Amount

1	JOHN	1	1000
1	JOHN	2	3000
8	DAVID	3	4000
8	DAVID	5	2000

Month_Table :

Month_ID	Month
1	JAN
2	FEB
3	MAR
4	APR
5	MAY
6	JUN
7	JUL
8	AUG
9	SEP
10	OCT
11	NOV
12	DEC

Output Table:

name	month	amount
DAVID	JAN	**NULL**
DAVID	FEB	**NULL**
DAVID	MAR	4000
DAVID	APR	**NULL**

DAVID	MAY	2000
DAVID	JUN	**NULL**
DAVID	JUL	**NULL**
DAVID	AUG	**NULL**
DAVID	SEP	**NULL**
DAVID	OCT	**NULL**
DAVID	NOV	**NULL**
DAVID	DEC	**NULL**
JOHN	JAN	1000
JOHN	FEB	3000
JOHN	MAR	**NULL**
JOHN	APR	**NULL**
JOHN	MAY	**NULL**
JOHN	JUN	**NULL**
JOHN	JUL	**NULL**
JOHN	AUG	**NULL**
JOHN	SEP	**NULL**
JOHN	OCT	**NULL**
JOHN	NOV	**NULL**
JOHN	DEC	**NULL**

Dataset :

```
Create Table Emp_Table (
SerialNo int,
Name Varchar(30),
Month_ID int,
Amount Bigint );
```

```
Insert into Emp_Table Values  (1,'JOHN',1,1000);
Insert into Emp_Table Values  (1,'JOHN',2,3000);
Insert into Emp_Table Values  (8,'DAVID',3,4000);
Insert into Emp_Table Values  (8,'DAVID',5,2000);

Create Table Month_Table(
Month_ID int,
Month Varchar(30));

Insert into Month_Table Values (1, 'JAN');
Insert into Month_Table Values (2, 'FEB');
Insert into Month_Table Values (3, 'MAR');
Insert into Month_Table Values (4, 'APR');
Insert into Month_Table Values (5, 'MAY');
Insert into Month_Table Values (6, 'JUN');
Insert into Month_Table Values (7, 'JUL');
Insert into Month_Table Values (8, 'AUG');
Insert into Month_Table Values (9, 'SEP');
Insert into Month_Table Values (10, 'OCT');
Insert into Month_Table Values (11, 'NOV');
Insert into Month_Table Values (12, 'DEC');
```

Result :

```
select z.name,z.month,b.amount from
(select distinct e.serialNo,e.Name,m.month_id,m.month from emp_table as e
cross join month_table as m order by name) as z
left join
emp_table as b on
z.month_id = b.month_id and z.name = b.name order by name;
```

3.Input :- Sales Table has three columns namely Id, Product and Sales
Problem Statement :- Write a SQL query to get the output as shown in the Output tables
Input Table :

Id	Product	Sales
1001	Keyboard	20
1002	Keyboard	25
1003	Laptop	30
1004	Laptop	35
1005	Laptop	40
1006	Monitor	45
1007	WebCam	50
1008	WebCam	55

Output Table:

id	product	sales
1001	Keyboard	20
1002	Keyboard	45
1003	Laptop	30
1004	Laptop	65
1005	Laptop	105
1006	Monitor	45
1007	WebCam	50
1008	WebCam	105

4.Input :- StudentInfo Table has three columns namely StudentName, Subjects and Marks

Problem Statement :- Write a SQL query to get the output as shown in the Output table

Input Table:

StudentName	Subjects	Marks
David	English	85
David	Maths	90
David	Science	88
John	English	75
John	Maths	85
John	Science	80
Tom	English	83
Tom	Maths	80
Tom	Science	92

Output Table:

StudentName	English	maths	science
David	85	90	88
John	75	85	80
Tom	83	80	92

Data set:

```
Create Table Student1
(
StudentName Varchar(30),
Subjects Varchar(30),
Marks Bigint
```

```
);

insert into Student1 Values ('David', 'English', 85);
insert into Student1 Values ('David', 'Maths', 90);
insert into Student1 Values ('David', 'Science', 88);
insert into Student1 Values ('John', 'English', 75);
insert into Student1 Values ('John', 'Maths', 85);
insert into Student1 Values ('John', 'Science', 80);
insert into Student1 Values ('Tom', 'English', 83);
insert into Student1 Values ('Tom', 'Maths', 80);
insert into Student1 Values ('Tom', 'Science', 92);
select * from student1;
```

Result :

```
select a.*,b.maths,c.science from (select distinct StudentName, Marks as English
from student1 where subjects = 'English') as a
left join
(select distinct StudentName, Marks as Maths from student1 where subjects =
'Maths') as b
on a.studentname = b.studentname
left join
(select distinct StudentName, Marks as Science from student1 where subjects =
'Science') as c
on
a.studentname = c.studentname;
```

5.Input :- Employee Table has three columns namely EmployeeID, EmployeeName and ManagerID
Problem Statement :- Write a SQL query to get the output as shown in the Output table

Input Table:

EmployeeID	EmployeeName	ManagerID

100	Mark	103
101	John	104
102	Maria	103
103	Tom	**NULL**
104	David	103

Output Table:

Employeename	Managername
Mark	Tom
John	David
Maria	Tom
Tom	Boss
David	Tom

Result:

```
select ax.employeename,ifnull(bx.managername,'Boss') as Managername from
employee_1 as ax left join
(select  b.employeename,a.employeename as managername from employee_1 as
a
inner join
employee_1 as b
on
a.employeeid = b.managerid) as bx
on ax.employeename = bx.employeename;
```

Dataset:

```
Create Table Employee_1(
EmployeeID Varchar(20),
EmployeeName Varchar(20),
ManagerID varchar(20));
```

```
Insert Into Employee_1 Values(100,'Mark',103);
Insert Into Employee_1 Values(101,'John',104);
Insert Into Employee_1 Values(102,'Maria',103);
Insert Into Employee_1 Values(103,'Tom',NULL);
Insert Into Employee_1 Values(104, 'David',103);
```

1.Input :- Balance table has two columns namely Balance and Dates.

Problem Statements :- Write SQL to derive Start_Date and End_Date columns when there is a continuous amount in the Balance column as shown below.

Input table:

Balance	Dates
26000	2020-01-01
26000	2020-01-02
26000	2020-01-03
30000	2020-01-04
30000	2020-01-05
26000	2020-01-06
26000	2020-01-07
32000	2020-01-08
31000	2020-01-09

Output table:

balance	starting_date	ending_date
26000	2020-01-01	2020-01-03
30000	2020-01-04	2020-01-05

26000	2020-01-06	2020-01-07
32000	2020-01-08	2020-01-08
31000	2020-01-09	2020-01-09

Dataset :

```
Create Table BalanceTbl(
Balance int,
Dates Date
)

Insert into BalanceTbl Values(26000,'2020-01-01');
Insert into BalanceTbl Values(26000,'2020-01-02');
Insert into BalanceTbl Values(26000,'2020-01-03');
Insert into BalanceTbl Values(30000,'2020-01-04');
Insert into BalanceTbl Values(30000,'2020-01-05');
Insert into BalanceTbl Values(26000,'2020-01-06');
Insert into BalanceTbl Values(26000,'2020-01-07');
Insert into BalanceTbl Values(32000,'2020-01-08');
Insert into BalanceTbl Values(31000,'2020-01-09');
```

Result :

```
select balance,min(dates) as starting_date,max(dates) as ending_date from (select
a.*,sum(result) over(order by dates) as rnk_result from (select *,lag(balance)
over(order by dates) as lag_value,
case
when lag(balance) over(order by dates) = balance then 0
else 1
end as result
 from tbl) as a) as b
 group by rnk_result;
```

2.Input : Account Table has four columns namely TransDate,TranID,TranType and Amount
Problem Statements : Write SQL to derive the Net_Balance column based on Credit/Debit of the Amount.

Input Table:

TranDate	TranID	TranType	Amount
2020-05-12 5:29:44	A10001	Credit	50000
2020-05-13 10:30:20	B10001	Debit	10000
2020-05-13 11:27:50	B10002	Credit	20000
2020-05-14 8:35:30	C10001	Debit	5000
2020-05-14 9:43:51	C10002	Debit	5000
2020-05-15 5:51:11	D10001	Credit	30000

Output Table:

TranDate	TranID	TranType	Amount	transaction
2020-05-12 5:29:44	A10001	Credit	50000	50000
2020-05-13 10:30:20	B10001	Debit	10000	40000
2020-05-13 11:27:50	B10002	Credit	20000	60000
2020-05-14 8:35:30	C10001	Debit	5000	55000
2020-05-14 9:43:51	C10002	Debit	5000	50000
2020-05-15 5:51:11	D10001	Credit	30000	80000

Dataset:

```
Create Table Account_Table(
TranDate DateTime,
TranID Varchar(20),
TranType Varchar(10),
```

```
Amount Float);

INSERT into Account_Table VALUES ('2020-05-12 05:29:44.120', 'A10001','Credit',
50000);
INSERT into Account_Table VALUES ('2020-05-13 10:30:20.100', 'B10001','Debit',
10000);
INSERT into Account_Table VALUES ('2020-05-13 11:27:50.130', 'B10002','Credit',
20000);
INSERT into Account_Table VALUES ('2020-05-14 08:35:30.123', 'C10001','Debit',
5000);
INSERT into Account_Table VALUES ('2020-05-14T09:43:51.100', 'C10002','Debit',
5000);
INSERT into Account_Table VALUES ('2020-05-15T05:51:11.117', 'D10001','Credit',
30000);
```

Result:

```
with mytable as
(select *,
case
when trantype = "credit" then amount
when trantype = "debit" then amount*-1
end as temp
 from account_table)
 select trandate,tranid,trantype,amount,sum(temp) over(order by trandate) as
transaction from mytable;
```

3.Input :- Transaction_Table has four columns namely AccountNumber, TransactionTime, TransactionID and Balance
Problem Statements :- Write SQL to get the most recent / latest balance, and TransactionID for each AccountNumber

Input Table:

AccountNumber	Transaction_Time	Transaction_ID	Balance

550	2020-05-12 5:29:44	1001	2000
550	2020-05-15 10:29:26	1002	8000
460	2020-03-15 11:29:24	1003	9000
460	2020-04-30 11:29:57	1004	7000
460	2020-04-30 12:32:44	1005	5000
640	2020-02-18 6:29:34	1006	5000
640	2020-02-18 6:29:37	1007	9000

Output Table:

AccountNumber	Transaction_Time	Transaction_ID	Balance
460	2020-04-30 12:32:44	1005	5000
550	2020-05-15 10:29:26	1002	8000
640	2020-02-18 6:29:37	1007	9000

Dataset:

```
create table transaction_table
(
AccountNumber int,
Transaction_Time  datetime,
Transaction_ID int,
Balance Int
);
insert into transaction_table values (550,'2020-05-12 05:29:44.120',1001,2000);
insert into transaction_table values (550,'2020-05-15 10:29:25.630',1002,8000);
```

```
insert into transaction_table values (460,'2020-03-15 11:29:23.620',1003,9000);
insert into transaction_table values (460,'2020-04-30 11:29:57.320',1004,7000);
insert into transaction_table values (460,'2020-04-30 12:32:44.233',1005,5000);
insert into transaction_table values (640,'2020-02-18 06:29:34.420',1006,5000);
insert into transaction_table values (640,'2020-02-18 06:29:37.120',1007,9000);
```

Result:

```
with a as (select *,row_number()
 over(partition by accountnumber order by transaction_time) as timing
 from transaction_table)
 select * from a where (transaction_id,timing) in (select
transaction_id,max(timing)
 over(partition by accountnumber) from
 (select *,row_number()
 over(partition by accountnumber order by transaction_time) as timing
 from transaction_table) as b);
```

4.SQL Scenario based Interview Question and Answer

Input :- SalesTable has four columns namely ID, Product , SalesYear and QuantitySold

Input Table:

ID	Product	SalesYear	QuantitySold
1	Laptop	1998	2500
2	Laptop	1999	3600
3	Laptop	2000	4200
4	Keyboard	1998	2300
5	Keyboard	1999	4800
6	Keyboard	2000	5000
7	Mouse	1998	6000

8	Mouse	1999	3400
9	Mouse	2000	4600

Output Table:

TotalSales	1998	1999	2000
TotalSales	10800	11800	13800

Dataset:

```
Create Table Sales (
ID int,
Product Varchar(25),
SalesYear Int,
QuantitySold Int);

Insert into Sales Values(1,'Laptop',1998,2500),(2,'Laptop',1999,3600)
,(3,'Laptop',2000,4200)
,(4,'Keyboard',1998,2300)
,(5,'Keyboard',1999,4800)
,(6,'Keyboard',2000,5000)
,(7,'Mouse',1998,6000)
,(8,'Mouse',1999,3400)
,(9,'Mouse',2000,4600);
```

Result:

```
select 'TotalSales' as TotalSales,
sum(case when SalesYear=1998 then QuantitySold end) as "1998" ,
sum(case when SalesYear=1999 then QuantitySold end) as "1999",
sum(case when SalesYear=2000 then QuantitySold end) as "2000"
from Sales
```

5.Write a SQL query to print all the numbers from 1 to 100 by using recursive CTE(Common Table Expression) ?

Result:

```
WITH RECURSIVE cte AS (
   SELECT 1 AS n
   UNION ALL
   SELECT n + 1 FROM cte WHERE n < 100
)
SELECT n FROM cte;
```

SQL : Theoretical Questions

1. What is the difference between DROP/DELETE/TRUNCATE?
2. What is the difference between count(*) and count(column_name)?
3. What is the difference between group by and having?
4. What is the difference between lead and lag function?
5. What is the difference between alter and update?
6. What is the difference between rank and dense_rank?
7. What is the difference between CTE and views?
8. What is the difference between limit and offset?
9. What is the difference between primary key and foreign key?
10. What is the difference between inner and natural join?