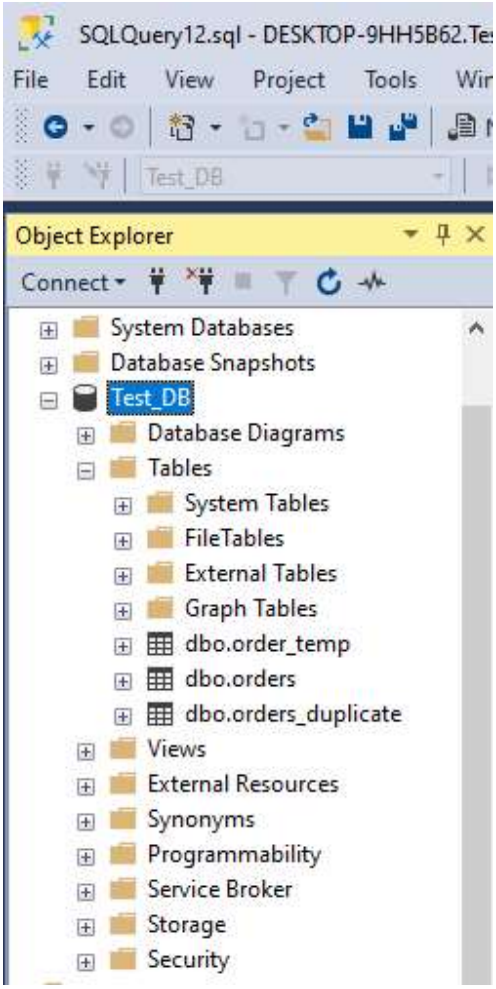
Name: Nitin

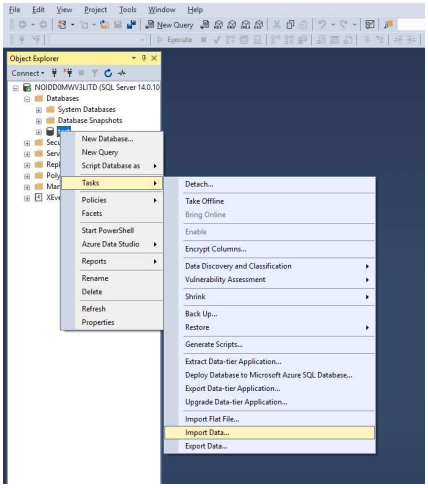
Email Id: nknitinyadav5@gmail.com

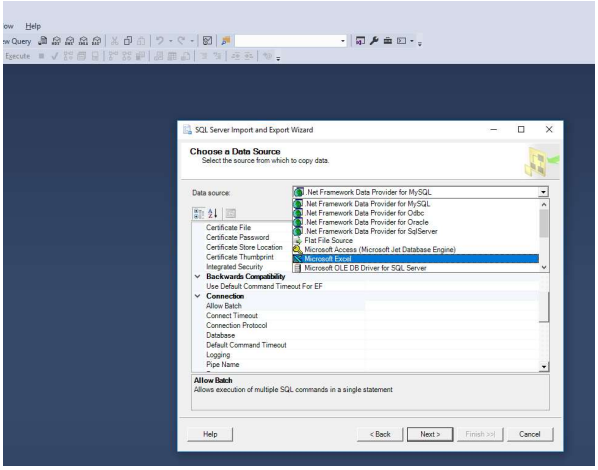
Assessment 1

Create New Database “Test_DB”



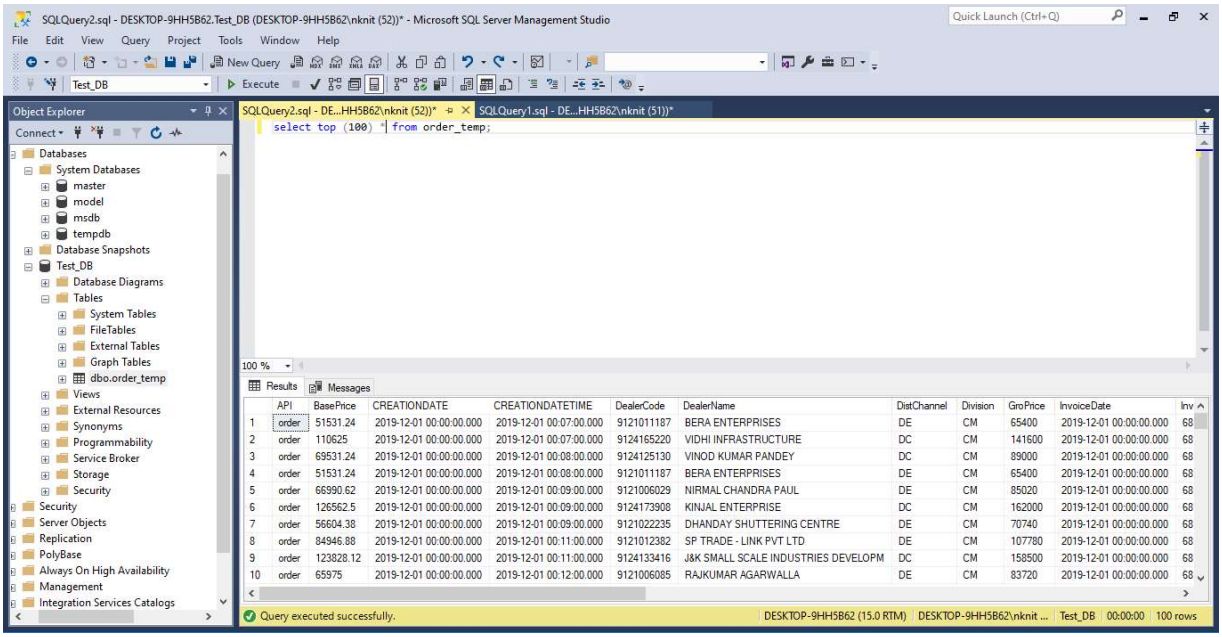
Importing Data From excel sheet in a temporary table with name “order_temp” :





To check that table and data in that is importind properly I checked it with the following query:

Select top 100 from order_temp;



	API	BasePrice	CREATIONDATE	CREATIONDATETIME	DealerCode	DealerName	DistChannel	Division	GroPrice	InvoiceDate	Inv
1	order	51531.24	2019-12-01 00:00:00.000	2019-12-01 00:07:00.000	9121011187	BERA ENTERPRISES	DE	CM	65400	2019-12-01 00:00:00.000	68
2	order	110625	2019-12-01 00:00:00.000	2019-12-01 00:07:00.000	9124165220	VIDHI INFRASTRUCTURE	DC	CM	141600	2019-12-01 00:00:00.000	68
3	order	69531.24	2019-12-01 00:00:00.000	2019-12-01 00:08:00.000	9124125130	VINOD KUMAR PANDEY	DC	CM	89000	2019-12-01 00:00:00.000	68
4	order	51531.24	2019-12-01 00:00:00.000	2019-12-01 00:08:00.000	9121011187	BERA ENTERPRISES	DE	CM	65400	2019-12-01 00:00:00.000	68
5	order	66990.62	2019-12-01 00:00:00.000	2019-12-01 00:09:00.000	9121006029	NIRMAL CHANDRA PAUL	DE	CM	85020	2019-12-01 00:00:00.000	68
6	order	126562.5	2019-12-01 00:00:00.000	2019-12-01 00:09:00.000	9124173908	KINJAL ENTERPRISE	DC	CM	162000	2019-12-01 00:00:00.000	68
7	order	56604.38	2019-12-01 00:00:00.000	2019-12-01 00:09:00.000	9121022235	DHANDAY SHUTTERING CENTRE	DE	CM	70740	2019-12-01 00:00:00.000	68
8	order	84946.88	2019-12-01 00:00:00.000	2019-12-01 00:11:00.000	9121012382	SP TRADE - LINK PVT LTD	DE	CM	107780	2019-12-01 00:00:00.000	68
9	order	123828.12	2019-12-01 00:00:00.000	2019-12-01 00:11:00.000	9124133416	J&K SMALL SCALE INDUSTRIES DEVELOPM	DC	CM	158500	2019-12-01 00:00:00.000	68
10	order	65975	2019-12-01 00:00:00.000	2019-12-01 00:12:00.000	9121006085	RAJKUMAR AGARWALLA	DE	CM	83720	2019-12-01 00:00:00.000	68

Create two tables with name “orders” and “orders_duplicate” table with following query:

Orders

```
CREATE TABLE orders (
   API      nvarchar(255),
BasePrice float,
CREATIONDATE      datetime,
CREATIONDATETIME            datetime,
DealerCode          float,
DealerName          nvarchar(255),
DistChannel         nvarchar(255),
Division   nvarchar(255),
GroPrice  float,
InvoiceDate         datetime,
InvoiceNor          float,
MRP       float,
MaterialDescription            nvarchar(255),
MaterialGrp2        nvarchar(255),
MaterialNumber    float,
Month     nvarchar(255),
NSP       float,
OfficerName         nvarchar(255),
Officercode         nvarchar(255),
OrderCreationdate datetime,
QTY       float,
RSOCode nvarchar(255),
Region    nvarchar(255),
SalesOfficeCode     nvarchar(255),
SuppPLname          nvarchar(255),
SuppPlant           nvarchar(255),
Tax       nvarchar(255),
Year      nvarchar(255),
);
```

orders_duplicate

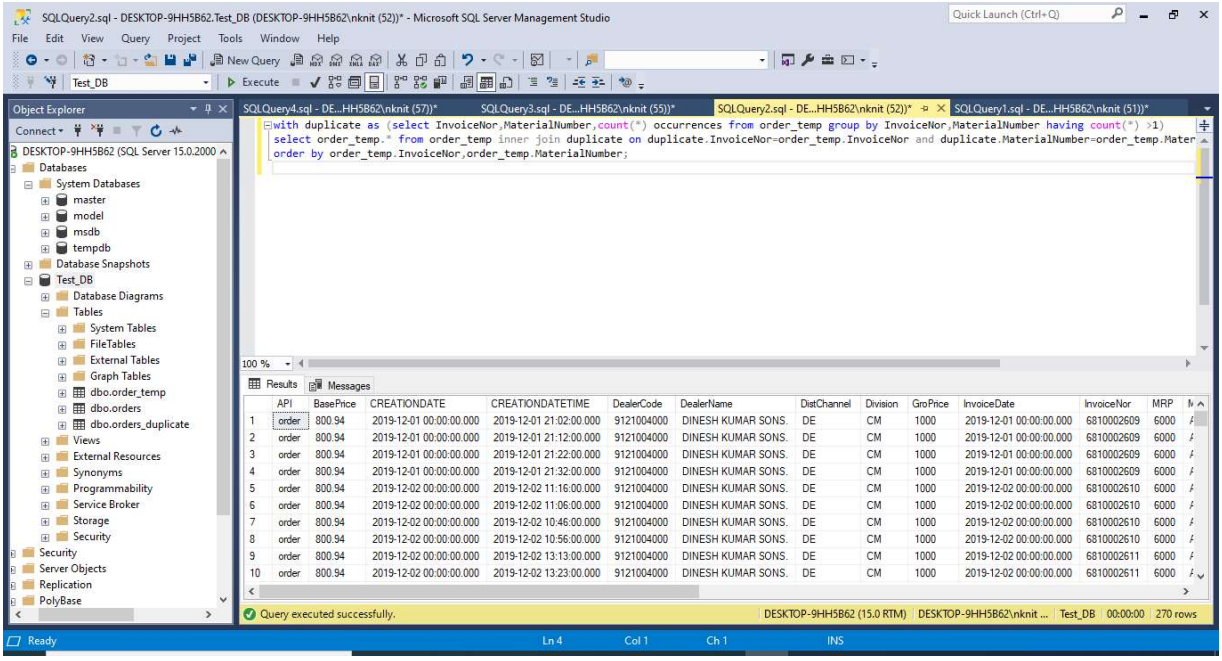
```
CREATE TABLE orders_duplicate (
   API      nvarchar(255),
BasePrice float,
CREATIONDATE      datetime,
CREATIONDATETIME            datetime,
DealerCode          float,
DealerName          nvarchar(255),
DistChannel         nvarchar(255),
Division   nvarchar(255),
GroPrice  float,
InvoiceDate         datetime,
InvoiceNor          float,
MRP       float,
MaterialDescription            nvarchar(255),
MaterialGrp2        nvarchar(255),
MaterialNumber    float,
Month     nvarchar(255),
NSP       float,
```

```
OfficerName          nvarchar(255),
Officercode          nvarchar(255),
OrderCreationdate datetime,
QTY        float,
RSOCode nvarchar(255),
Region     nvarchar(255),
SalesOfficeCode      nvarchar(255),
SuppPLname           nvarchar(255),
SuppPlant            nvarchar(255),
Tax        nvarchar(255),
Year       nvarchar(255),
);
```

We have a table named “orders” which is empty now. In this table I have to insert only the unique data from “order_temp”

1. UPLOAD DATA IN ORDER TABLE WITH UNIQUE KEY ' InvoiceNor, MaterialNumber'.

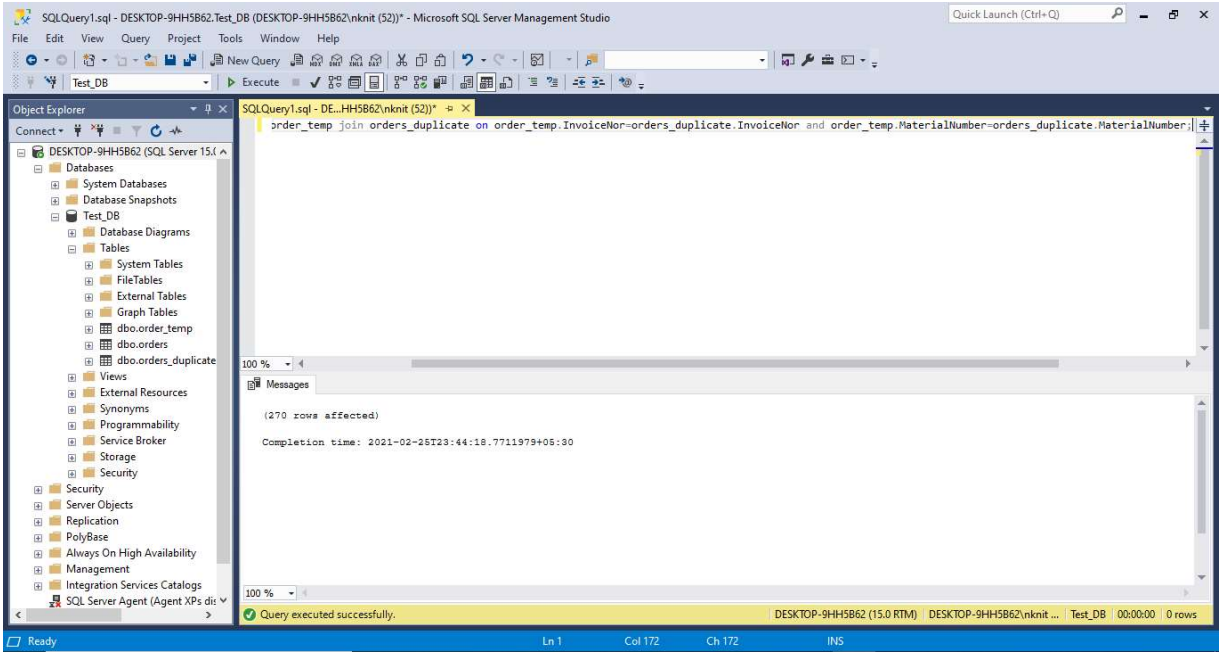
```
Query: with duplicate as (select InvoiceNor,MaterialNumber,count(*) occurrences from
order_temp group by InvoiceNor,MaterialNumber having count(*) >1)
select order_temp.* from order_temp inner join duplicate on
duplicate.InvoiceNor=order_temp.InvoiceNor and
duplicate.MaterialNumber=order_temp.MaterialNumber
order by order_temp.InvoiceNor,order_temp.MaterialNumber;
```



	API	BasePrice	CREATIONDATE	CREATIONDATETIME	DealerCode	DealerName	DistChannel	Division	GroPrice	InvoiceDate	InvoiceNor	MRP
1	order	800.94	2019-12-01 00:00:00.000	2019-12-01 21:02:00.000	9121004000	DINESH KUMAR SONS.	DE	CM	1000	2019-12-01 00:00:00.000	6810002609	6000
2	order	800.94	2019-12-01 00:00:00.000	2019-12-01 21:12:00.000	9121004000	DINESH KUMAR SONS.	DE	CM	1000	2019-12-01 00:00:00.000	6810002609	6000
3	order	800.94	2019-12-01 00:00:00.000	2019-12-01 21:22:00.000	9121004000	DINESH KUMAR SONS.	DE	CM	1000	2019-12-01 00:00:00.000	6810002609	6000
4	order	800.94	2019-12-01 00:00:00.000	2019-12-01 21:32:00.000	9121004000	DINESH KUMAR SONS.	DE	CM	1000	2019-12-01 00:00:00.000	6810002609	6000
5	order	800.94	2019-12-02 00:00:00.000	2019-12-02 11:16:00.000	9121004000	DINESH KUMAR SONS.	DE	CM	1000	2019-12-02 00:00:00.000	6810002610	6000
6	order	800.94	2019-12-02 00:00:00.000	2019-12-02 11:06:00.000	9121004000	DINESH KUMAR SONS.	DE	CM	1000	2019-12-02 00:00:00.000	6810002610	6000
7	order	800.94	2019-12-02 00:00:00.000	2019-12-02 10:46:00.000	9121004000	DINESH KUMAR SONS.	DE	CM	1000	2019-12-02 00:00:00.000	6810002610	6000
8	order	800.94	2019-12-02 00:00:00.000	2019-12-02 10:56:00.000	9121004000	DINESH KUMAR SONS.	DE	CM	1000	2019-12-02 00:00:00.000	6810002610	6000
9	order	800.94	2019-12-02 00:00:00.000	2019-12-02 13:13:00.000	9121004000	DINESH KUMAR SONS.	DE	CM	1000	2019-12-02 00:00:00.000	6810002611	6000
10	order	800.94	2019-12-02 00:00:00.000	2019-12-02 13:23:00.000	9121004000	DINESH KUMAR SONS.	DE	CM	1000	2019-12-02 00:00:00.000	6810002611	6000

1. Selecting all these entries, copied them and pasted in **"orders_duplicate"** table now this table has 270 rows, now all these entries have duplicate rows which are not required to be inserted in **"orders"** table. So after keeping them safe in a temporary table I have deleted these entry from **"order_temp"** using following query.

```
delete order_temp from order_temp JOIN orders_duplicate on
order_temp.InvoiceNor = orders_duplicate.InvoiceNor AND
order_temp.MaterialNumber = orders_duplicate.MaterialNumber
```



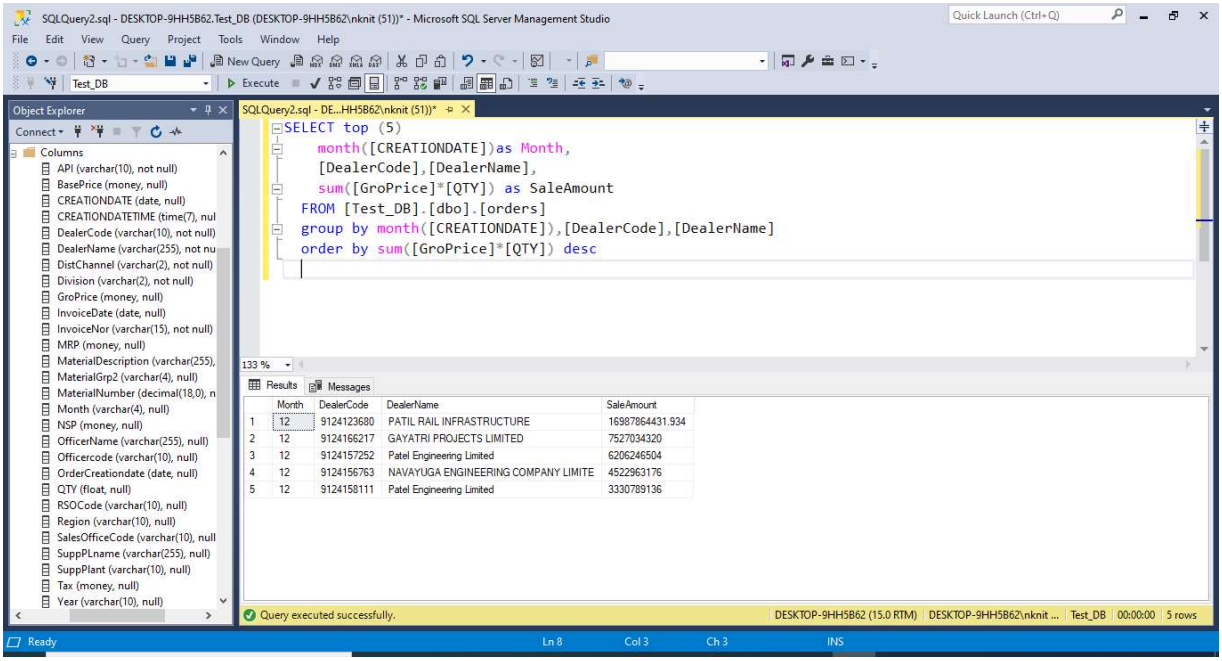
2. Now 270 rows are deleted and we have distinct rows and now inserting these rows to our “orders” table, using following query:

```
Insert into orders
select * from order_temp
```

3. 78715 rows are inserted.

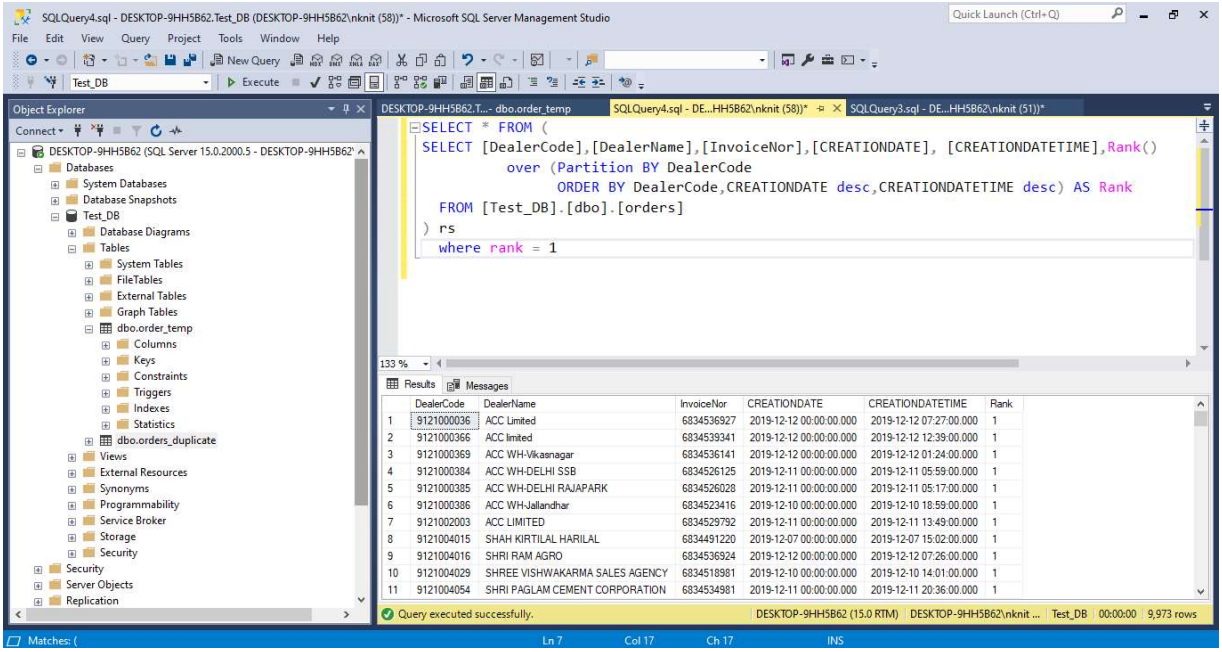
Query 2 : FIND TOP 5 DEALER IN MONTH ON BASIS OF AMOUNT

```
SELECT top (5)
    month([CREATIONDATE])as Month,
       [DealerCode],[DealerName],
    sum([GroPrice]*[QTY]) as SaleAmount
  FROM [Test_DB].[dbo].[orders]
  group by month([CREATIONDATE]),[DealerCode],[DealerName]
         order by sum([GroPrice]*[QTY]) desc
```



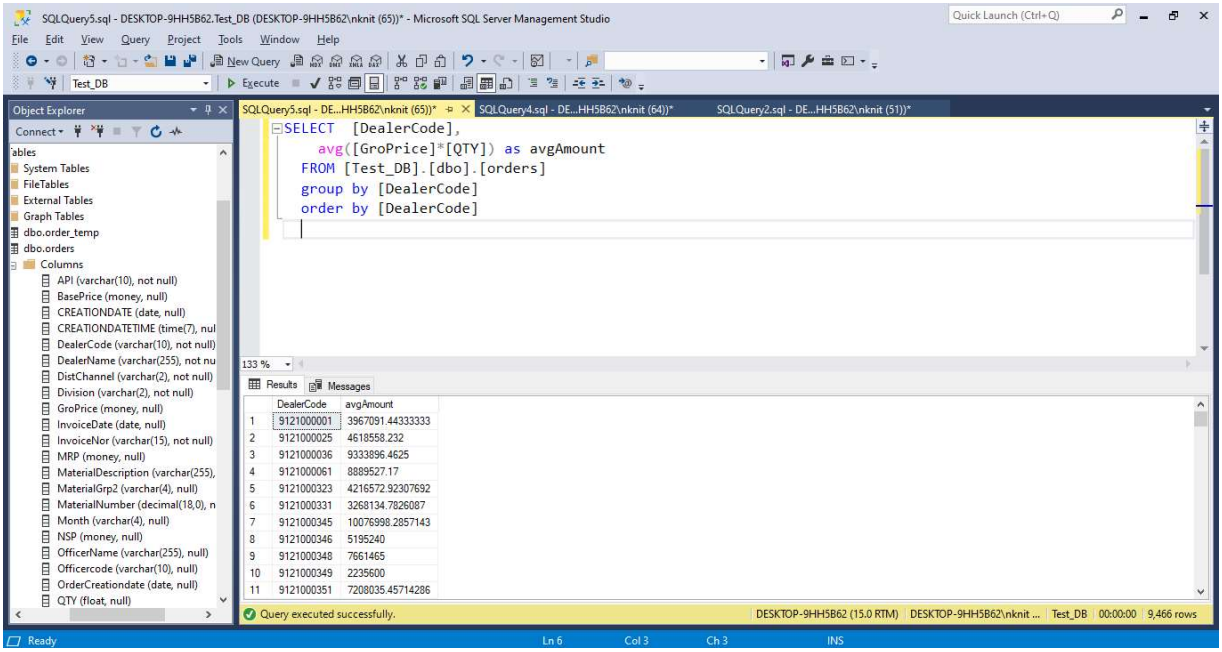
Query 3 : FIND LAST ORDER OF EACH DELEAR

```
SELECT * FROM (
SELECT [DealerCode],[DealerName],[InvoiceNor],[CREATIONDATE], [CREATIONDATETIME],Rank()
          over (Partition BY DealerCode
                ORDER BY DealerCode,CREATIONDATE desc,CREATIONDATETIME desc) AS Rank
  FROM [Test_DB].[dbo].[orders]
) rs  where rank = 1
```



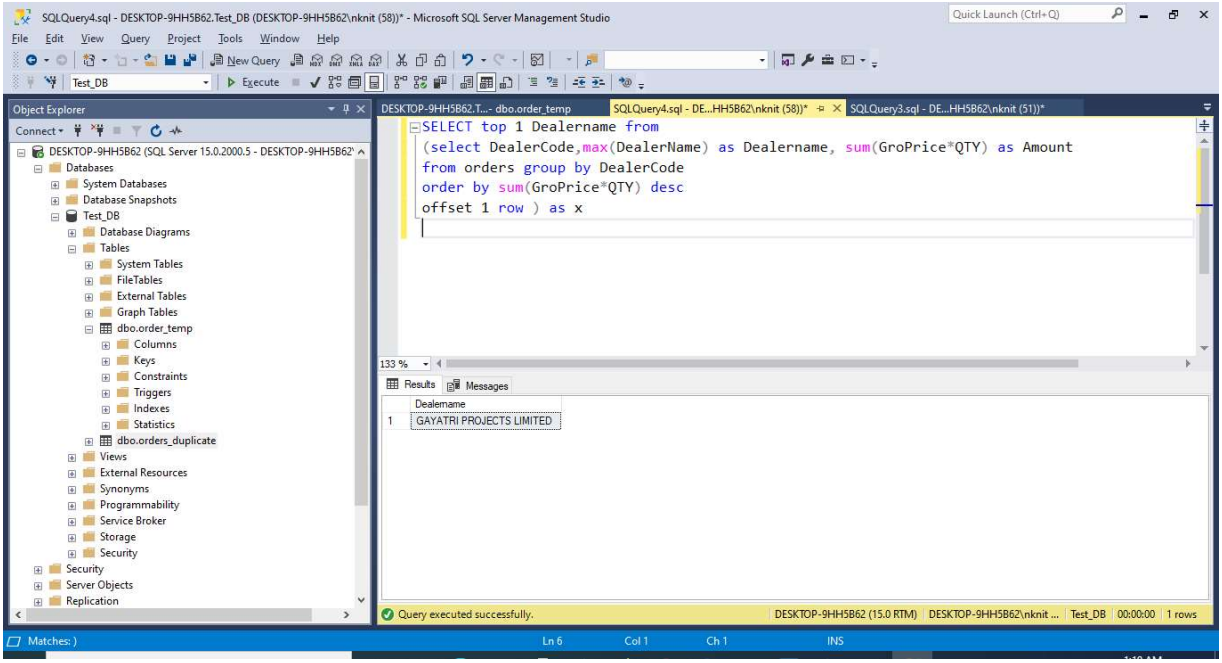
Query 4. FIND AVARAGE AMOUNT OF EACH DELEAR

```
SELECT [DealerCode],
    avg([GroPrice]*[QTY]) as avgAmount
  FROM [Test_DB].[dbo].[orders]
  group by [DealerCode]
        order by [DealerCode]
```



Query 5. FIND 2ND HIGHEST BILLED AMOUNT DEALEAR NAME

```
SELECT top 1 Dealername from
(select DealerCode,max(DealerName) as Dealername, sum(GroPrice*QTY) as Amount
from orders group by DealerCode
order by sum(GroPrice*QTY) desc
       offset 1 row ) as x;
```



Query 6. FIND HOURLY AMOUNT BILLED

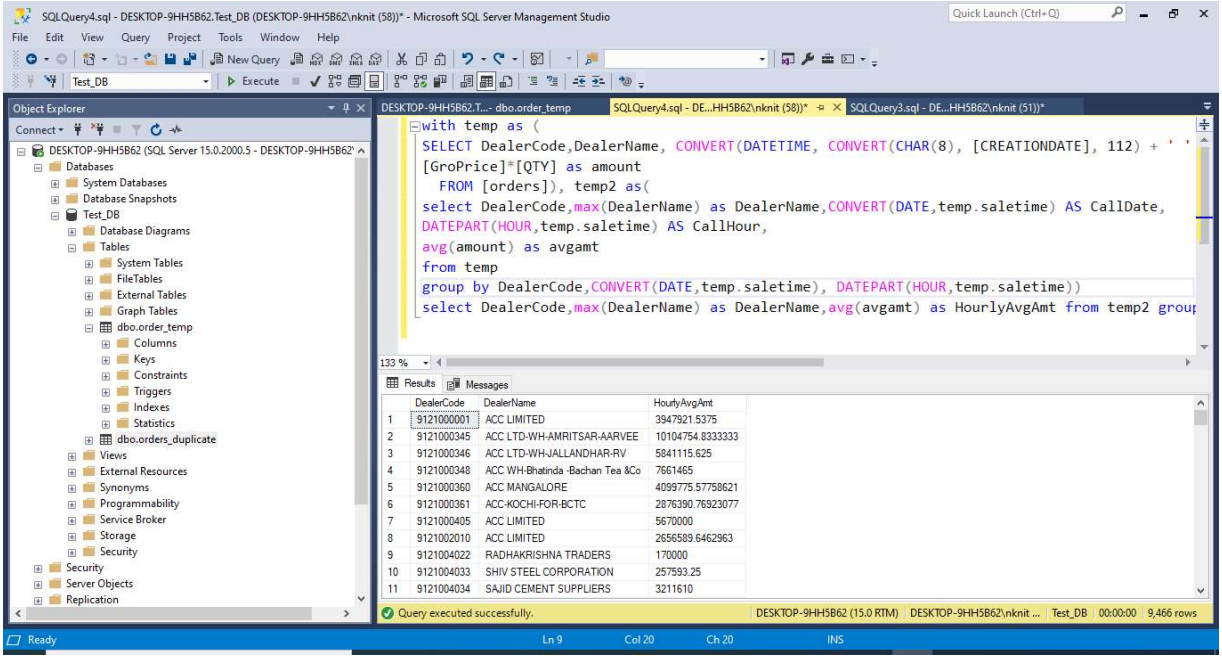
```
with temp as (
SELECT DealerCode,DealerName, CONVERT(DATETIME, CONVERT(CHAR(8), [CREATIONDATE], 112) + '
' + CONVERT(CHAR(8), [CREATIONDATETIME], 108)) as saletime,
[GroPrice]*[QTY] as amount
  FROM [orders]), temp2 as(
select DealerCode,max(DealerName) as DealerName,CONVERT(DATE,temp.saletime) AS CallDate,
DATEPART(HOUR,temp.saletime) AS CallHour,
avg(amount) as avgamt
from temp
group by DealerCode,CONVERT(DATE,temp.saletime), DATEPART(HOUR,temp.saletime))
    select DealerCode,max(DealerName) as DealerName,avg(avgamt) as HourlyAvgAmt from temp2
    group by DealerCode
```

Comments:

1) --CONVERT(CHAR(8), [CREATIONDATETIME], 108) : this function will convert the given datetime to hh:mm:ss as string.

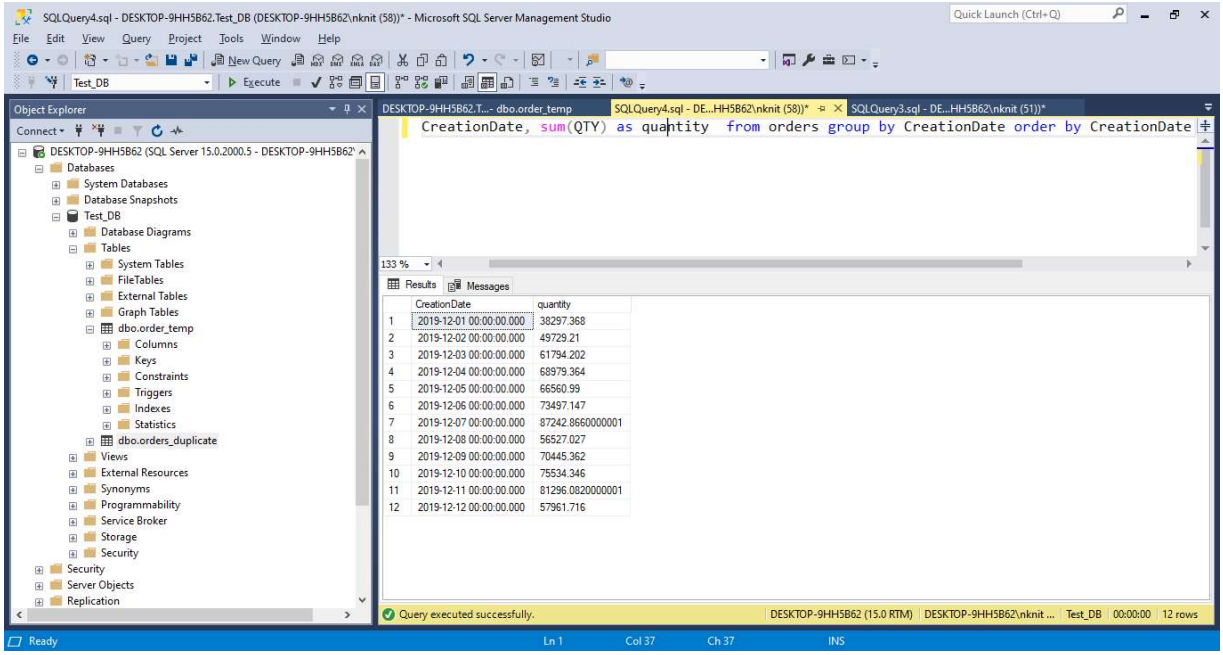
2) -- CONVERT(CHAR(8), [CREATIONDATE], 112) : this function will convert the given datetime to yyyy-mm-dd as string.

3) -- CONVERT(DATETIME, CONVERT(CHAR(8), [CREATIONDATE], 112) + ' ' + CONVERT(CHAR(8), [CREATIONDATETIME], 108)) : this function will convert the resulted string of format ‘yyyy-mm-dd hh:mm:ss’ (2019-12-01 01:28:00.000) to DATETIME



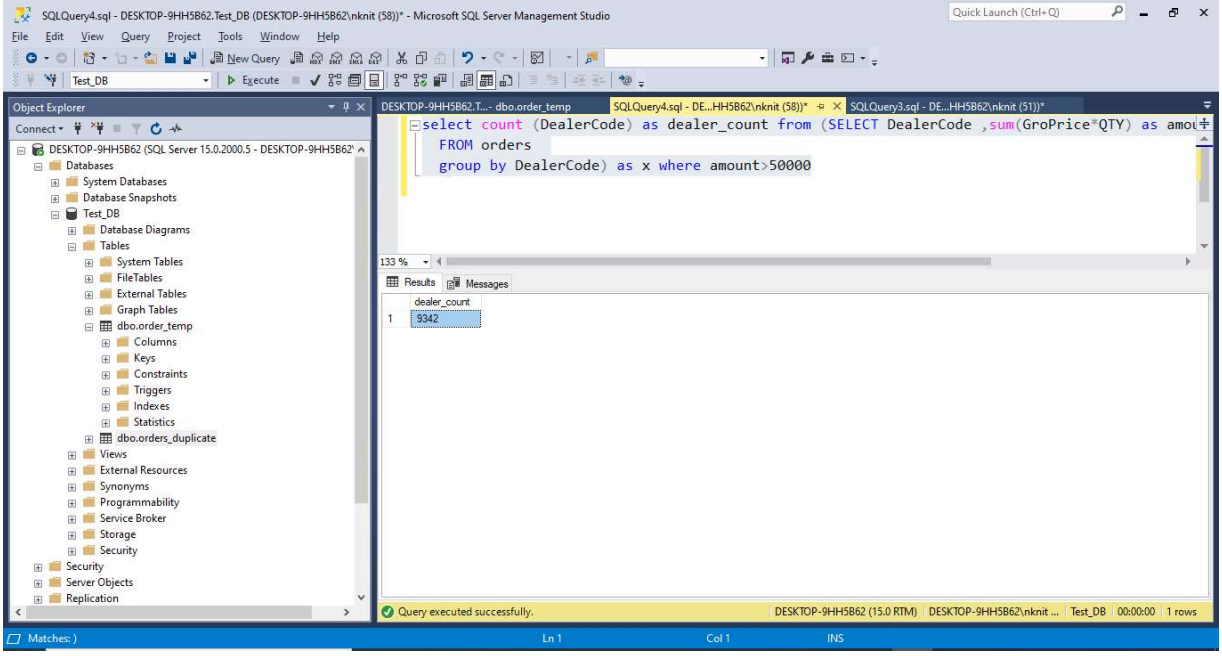
Query 7 . FIND DATEWISE TOTAL QUATITY

```
select CreationDate, sum(QTY) as quantity  from orders group by CreationDate order
by CreationDate
```



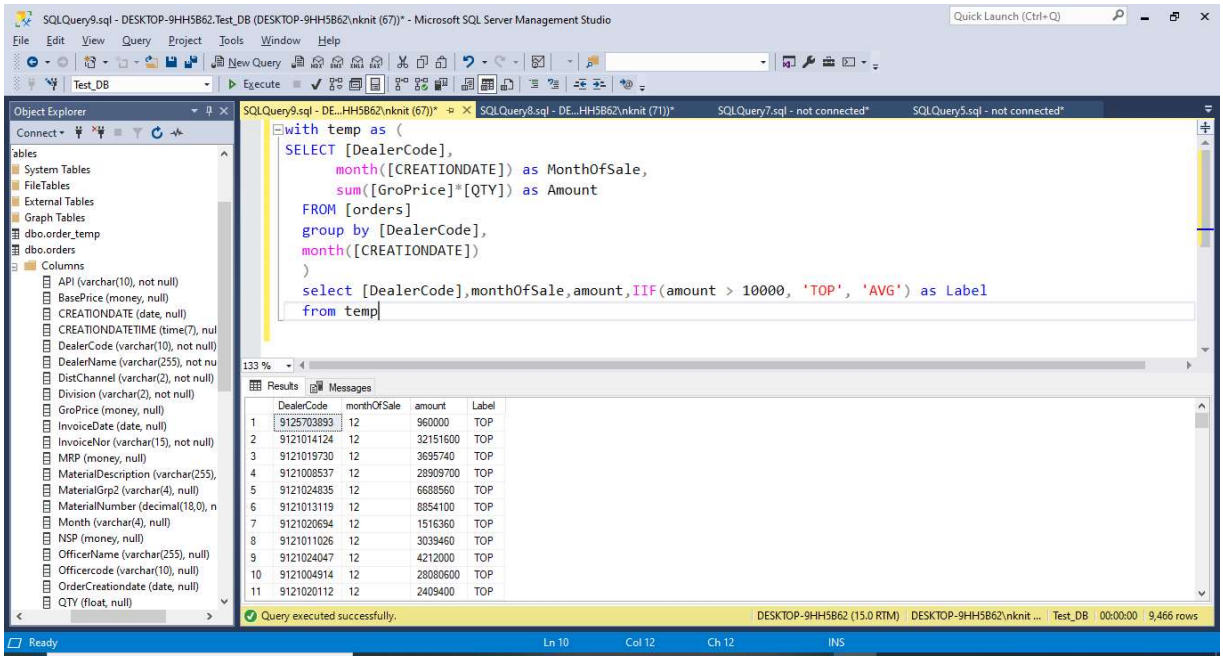
Query 8. FIND TOTAL NO OF DELEAR WHO HAVE AMOUNT GREATER THAN 50000

```
select count (DealerCode) as dealer_count from (SELECT DealerCode ,sum(GroPrice*QTY) as
amount
  FROM orders group by DealerCode) as x where amount>50000;
```



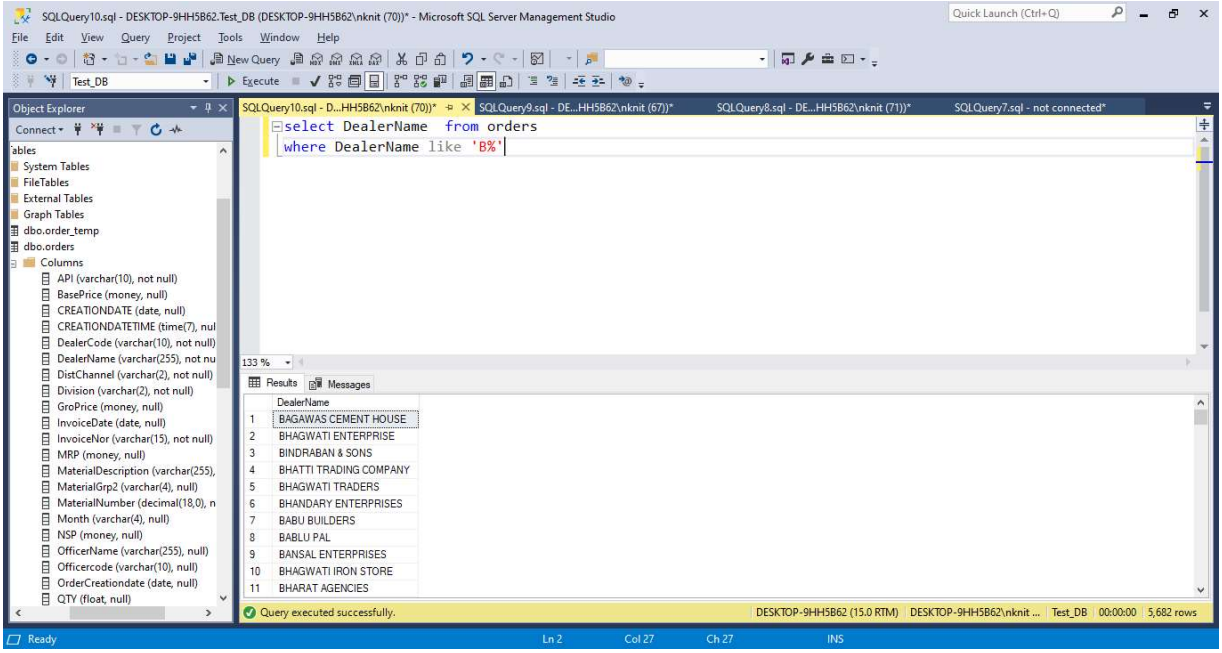
QUERY 9. PUT A LABLE **TOP** IF DEALER AMOUNT IN MONTH IS GEATER THAN 10000 ELSE LABLE **AVG** AND SHOW LIST

```
with temp as (
SELECT [DealerCode],
      month([CREATIONDATE]) as MonthOfSale,
      sum([GroPrice]*[QTY]) as Amount
  FROM [orders]
  group by [DealerCode],
  month([CREATIONDATE])
  )
  select [DealerCode],monthOfSale,amount,IIF(amount > 10000, 'TOP', 'AVG') as Label
        from temp
```



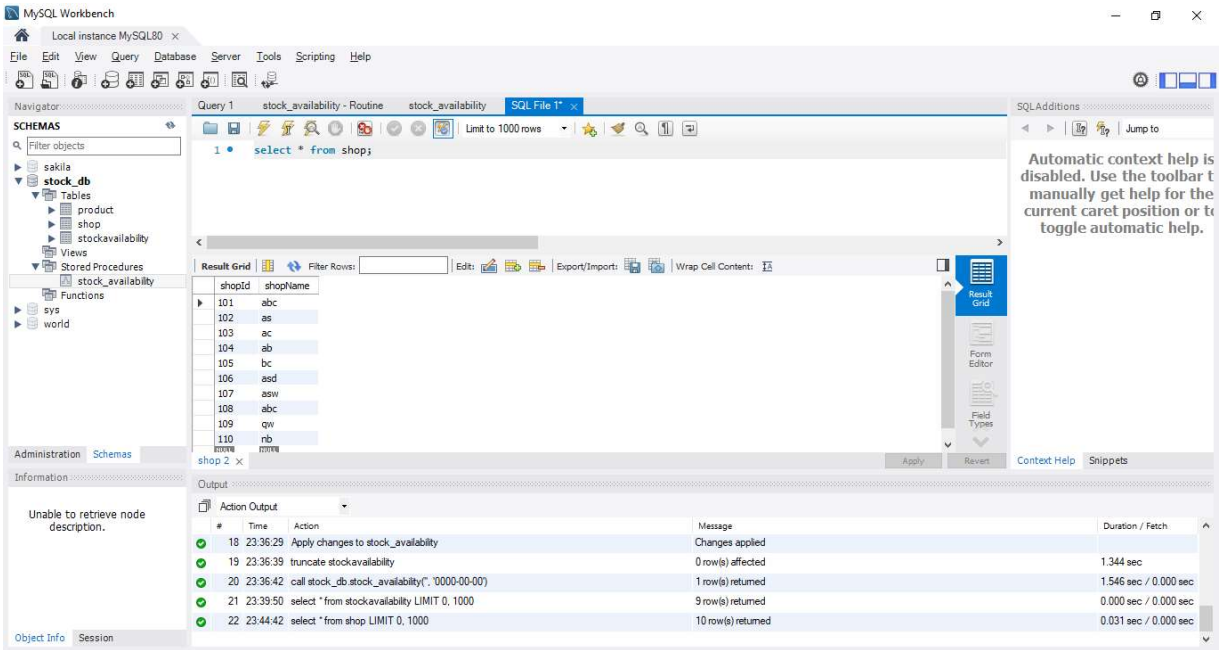
QUERY 10. LIST DELEAR NAME WHOSE NAME STARTS WITH **B**

```
select DealerName  from orders
       where DealerName like 'B%'
```

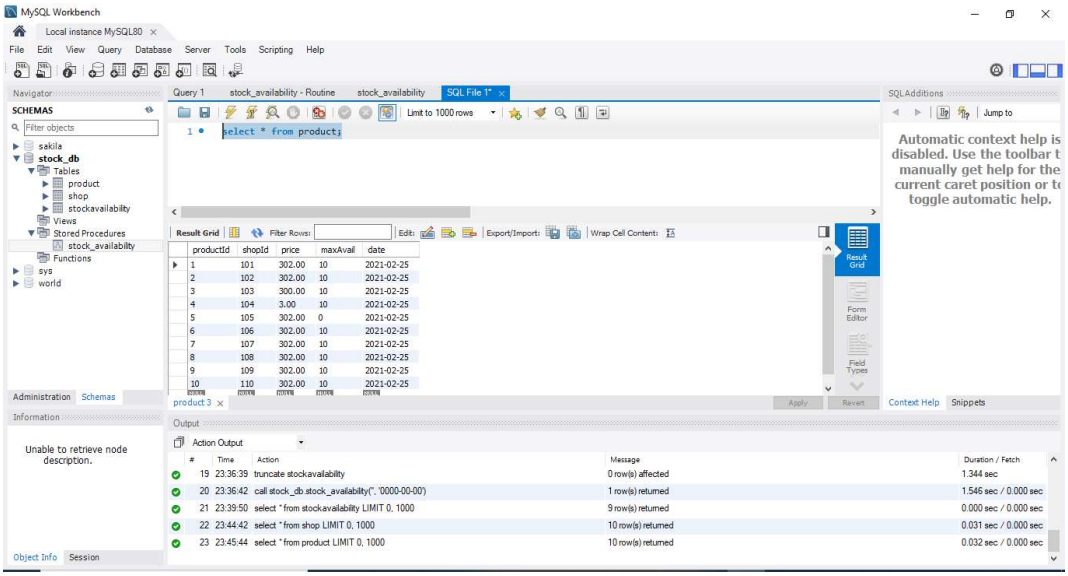


Assessment 2

```
CREATE TABLE `shop` (
  `shopId` int NOT NULL,
  `shopName` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`shopId`),
  KEY `shop_index1` (`shopId`)
)

INSERT INTO shop (shopId,shopName)
values
(101,'abc'),
(102,'as'),
(103,'ac'),
(104,'ab'),
(105,'bc'),
(106,'asd'),
(107,'asw'),
(108,'abc'),
(109,'qw'),
(110,'nb');

Select * from shop;
```



```
CREATE TABLE `product` (
  `productId` int NOT NULL,
  `shopId` int NOT NULL,
  `price` decimal(10,2) DEFAULT NULL,
  `maxAvail` int DEFAULT NULL,
  `date` date DEFAULT NULL,
  PRIMARY KEY (`productId`),
  KEY `shopId` (`shopId`),
  KEY `product_index1` (`productId`,`shopId`),
  CONSTRAINT `product_ibfk_1` FOREIGN KEY (`shopId`) REFERENCES `shop` (`shopId`)
)

INSERT INTO product(productId,shopId,price,maxAvail,date)
  Values
  (1,101,302,10,'25022021')
  (2,102,302,10,'2021-02-25'),
  (3,103,300,10,'2021-02-25'),
  (4,104,3,10,'2021-02-25'),
  (5,105,302,0,'2021-02-25'),
  (6,106,302,10,'2021-02-25'),
  (7,107,302,10,'2021-02-25'),
  (8,108,302,10,'2021-02-25'),
  (9,109,302,10,'2021-02-25'),
  (10,110,302,10,'2021-02-25');

  select * from product;
```



```
CREATE TABLE `stockavailability` (
  `stkId` int NOT NULL,
  `productId` int NOT NULL,
  `stkAvail` int DEFAULT NULL,
  `price` decimal(10,2) DEFAULT NULL,
  `date` date DEFAULT NULL,
  PRIMARY KEY (`stkId`),
  KEY `StockAvailability_index1` (`stkId`,`productId`)
)
```

Initially stockavailability table is blank and as required the data will be inserted from product table using stored procedure.

Stored procedure for inserting data in stockavailability :-

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `stock_availability`(prodId varchar(50),Date
datetime)
BEGIN
Declare stock varchar(50);
DECLARE is_complete,quantity_in_stock,qty INTEGER default 0;
Declare stock_cursor CURSOR FOR
select  productId from product;
declare continue handler for not found SET  is_complete = 1;
OPEN stock_cursor;
FETCH stock_cursor into stock;
REPEAT
     SELECT  maxAvail INTO quantity_in_stock
     FROM  product
     WHERE  productId = stock;
IF quantity_in_stock = 0
THEN
Select concat('Error Code:5000 - Item out Of Stock for product Id'," ", stock) as error;
ELSE
set @query=concat("Insert Into stockavailability(stkId, productId, stkAvail, price, date)
select concat(productId,shopId) as stkId,productId,maxAvail,price,date from product where
maxAvail !=0 and productId = ",stock,";");
prepare stmt from @query ;
execute stmt ;
deallocate prepare stmt;
end if;
FETCH stock_cursor into stock;
     UNTIL  is_complete = 1
     END REPEAT;
CLOSE stock_cursor;

END
```

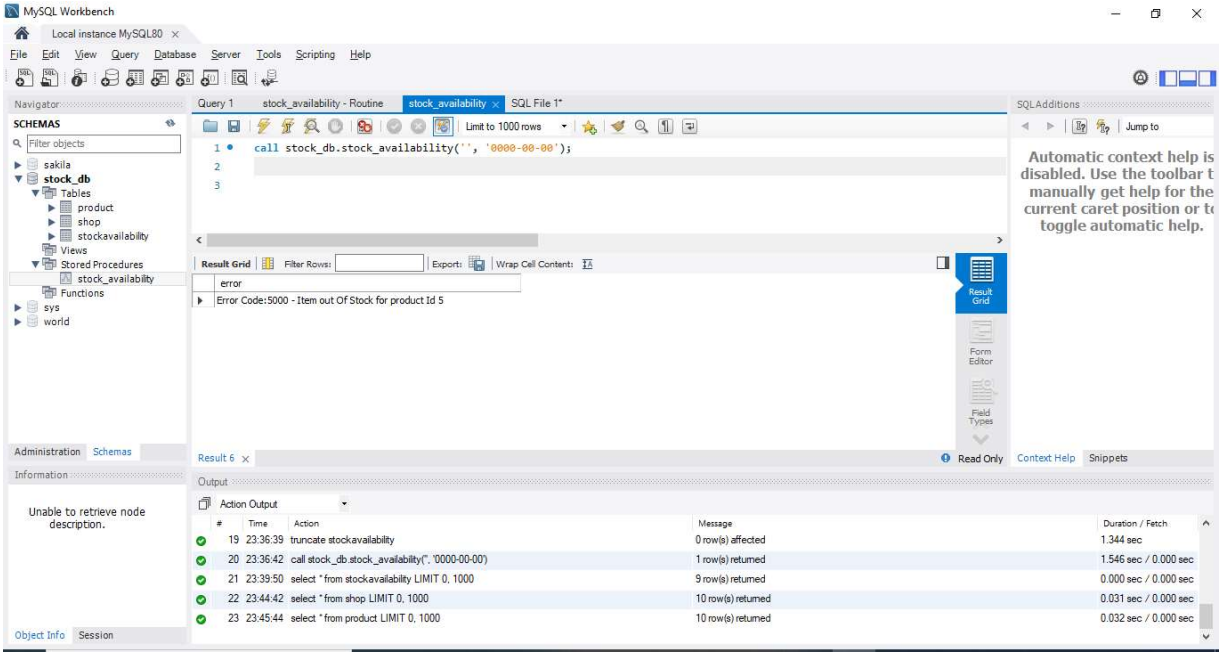
Now Calling the procedure using the statement :

```
call stock_db.stock_availability('', '0000-00-00');
```

Calling the above procedure data has been inserted in the stockavailability table for 9 shops.
The data has not been inserted for productId = ‘5’ as the maxAvail=0.

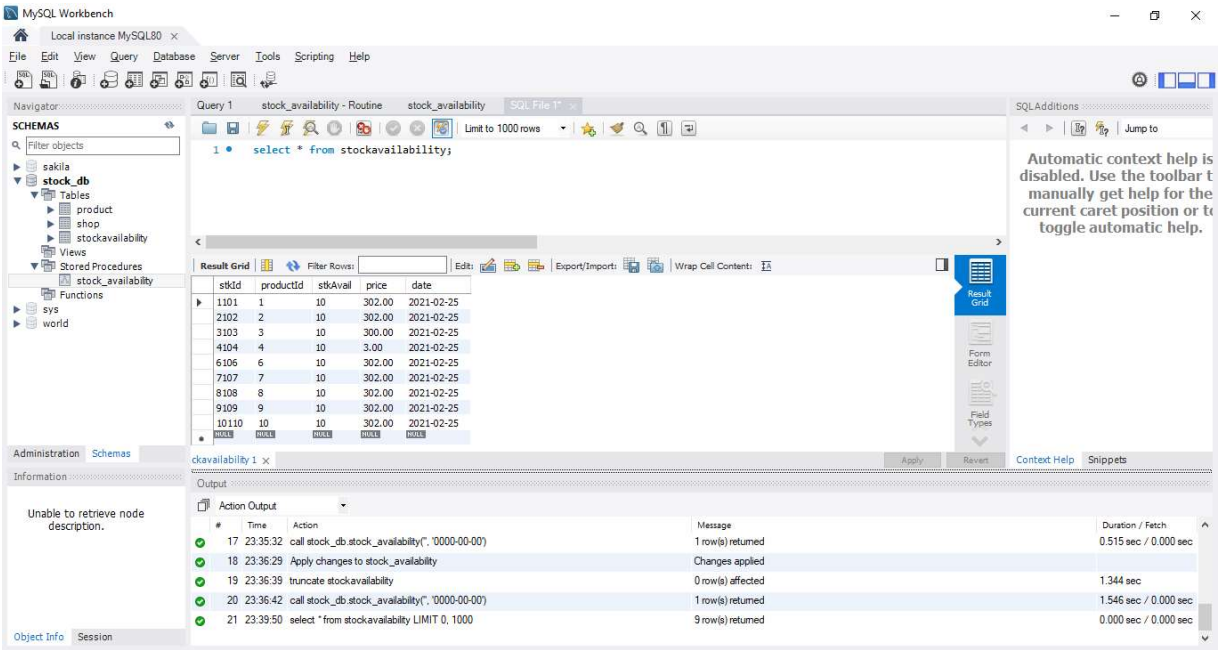
As required the shop with maxAvail = 0 or the shop with no product availability the data will not be inserted in the table-stockavailability and error message will be displayed as shown below:

Output: Error Code:5000 - Item out Of Stock for product Id 5.



Now checking whether the data has been inserted into the table stockavailability.

Select * from stockavailability;



As seen data has been inserted for the shops having products availability that is for 9 shops.
