uOttawa

DTI5126 – FUNDAMENTALS/APPLIED DATA SCIENCE

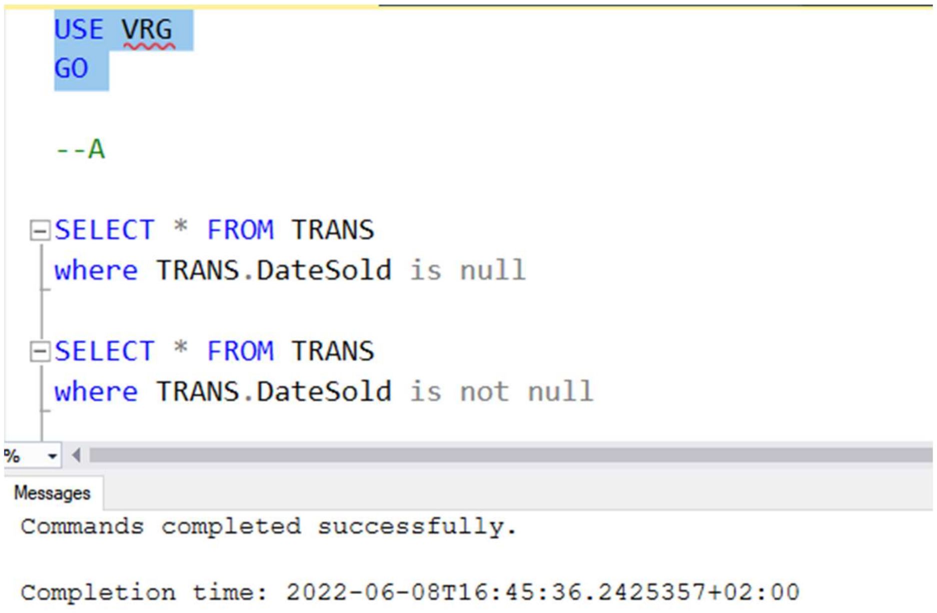
Assignment 1

Name: Ali Amin El-Sayed Mahmoud El-Sherif ID: 300327246

Table of Contents

Part A

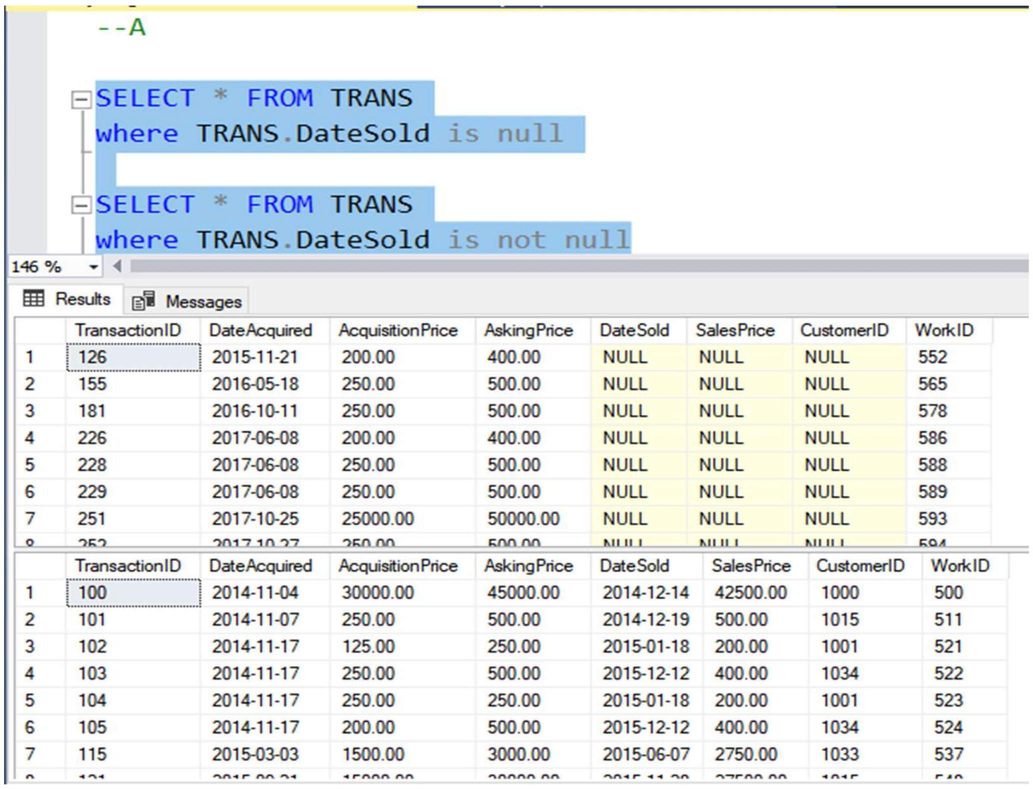
First we called the VGR database that we are going to use



```sql
USE VRG
GO

--A

SELECT * FROM TRANS
where TRANS.DateSold is null

SELECT * FROM TRANS
where TRANS.DateSold is not null
```

```
Commands completed successfully.

Completion time: 2022-06-08T16:45:36.2425357+02:00
```

A) Here we showed the TRANS table with and without (DateSold = null)

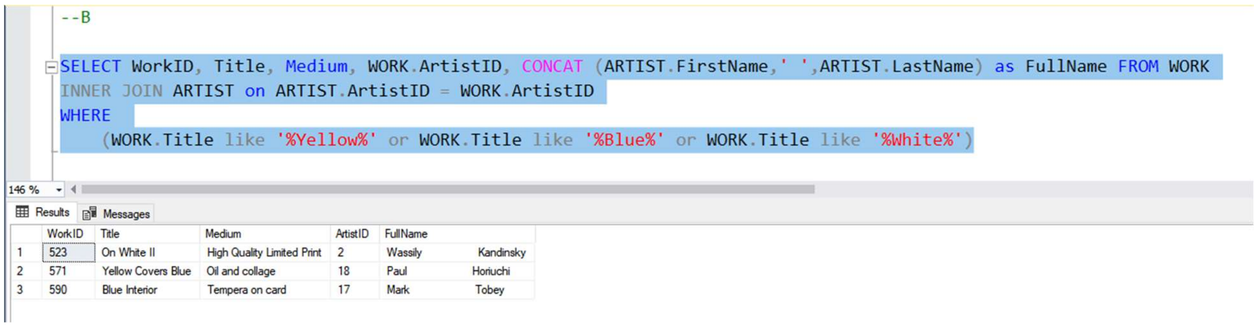


```sql
--A

SELECT * FROM TRANS
where TRANS.DateSold is null

SELECT * FROM TRANS
where TRANS.DateSold is not null
```

	TransactionID	DateAcquired	AcquisitionPrice	AskingPrice	DateSold	SalesPrice	CustomerID	WorkID
1	126	2015-11-21	200.00	400.00	NULL	NULL	NULL	552
2	155	2016-05-18	250.00	500.00	NULL	NULL	NULL	565
3	181	2016-10-11	250.00	500.00	NULL	NULL	NULL	578
4	226	2017-06-08	200.00	400.00	NULL	NULL	NULL	586
5	228	2017-06-08	250.00	500.00	NULL	NULL	NULL	588
6	229	2017-06-08	250.00	500.00	NULL	NULL	NULL	589
7	251	2017-10-25	25000.00	50000.00	NULL	NULL	NULL	593

	TransactionID	DateAcquired	AcquisitionPrice	AskingPrice	DateSold	SalesPrice	CustomerID	WorkID
1	100	2014-11-04	30000.00	45000.00	2014-12-14	42500.00	1000	500
2	101	2014-11-07	250.00	500.00	2014-12-19	500.00	1015	511
3	102	2014-11-17	125.00	250.00	2015-01-18	200.00	1001	521
4	103	2014-11-17	250.00	500.00	2015-12-12	400.00	1034	522
5	104	2014-11-17	250.00	250.00	2015-01-18	200.00	1001	523
6	105	2014-11-17	200.00	500.00	2015-12-12	400.00	1034	524
7	115	2015-03-03	1500.00	3000.00	2015-06-07	2750.00	1033	537

B) We have concatenated the first and last name of the artist as a FullName using "CONCAT" function
Then we have showed the artist's names only if the title of their books include yellow, blue, white word on it using "WHERE" function with medium, title, WorkID, ArtistID and their full name.



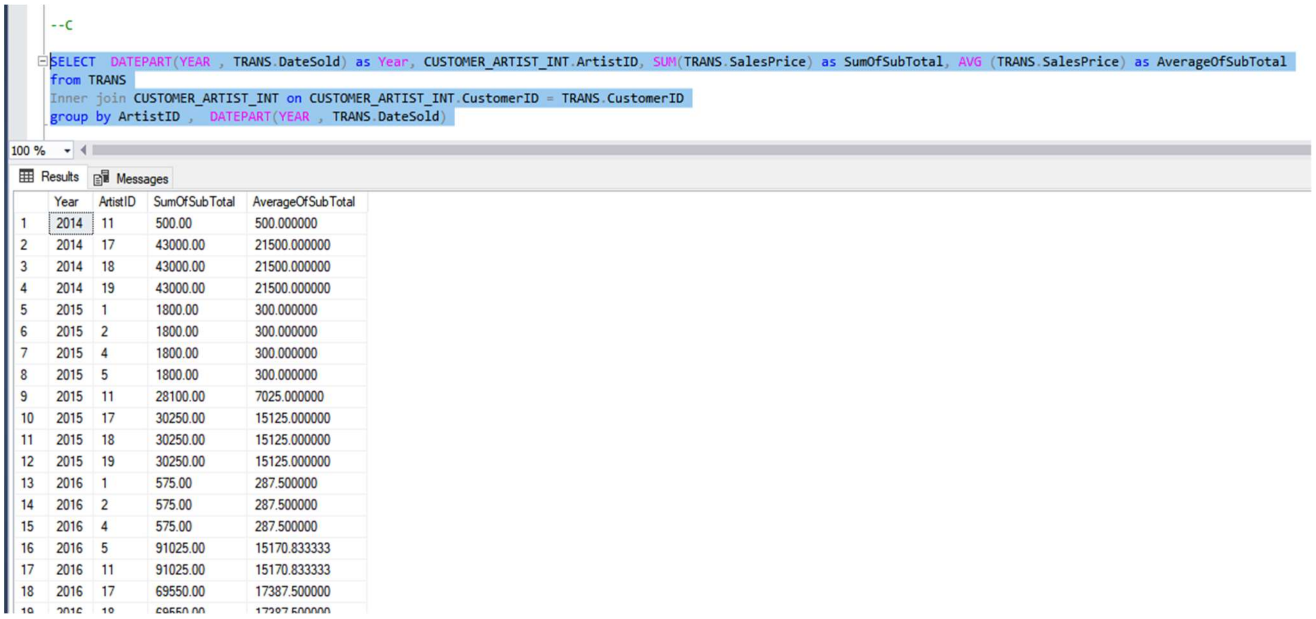
```
--B

SELECT WorkID, Title, Medium, WORK.ArtistID, CONCAT (ARTIST.FirstName,' ',ARTIST.LastName) as FullName FROM WORK
INNER JOIN ARTIST on ARTIST.ArtistID = WORK.ArtistID
WHERE
    (WORK.Title like '%Yellow%' or WORK.Title like '%Blue%' or WORK.Title like '%White%')
```

146 %

Results | Messages

	WorkID	Title	Medium	ArtistID	FullName
1	523	On White II	High Quality Limited Print	2	Wassily Kandinsky
2	571	Yellow Covers Blue	Oil and collage	18	Paul Horiuchi
3	590	Blue Interior	Tempera on card	17	Mark Tobey

C) We have renamed the DateSold as Year, then we have calculated the sum of the sales prices and renamed it as SumOfSubTotal, then calculated the average of the sales price and renamed it as AverageOfSubtotal.



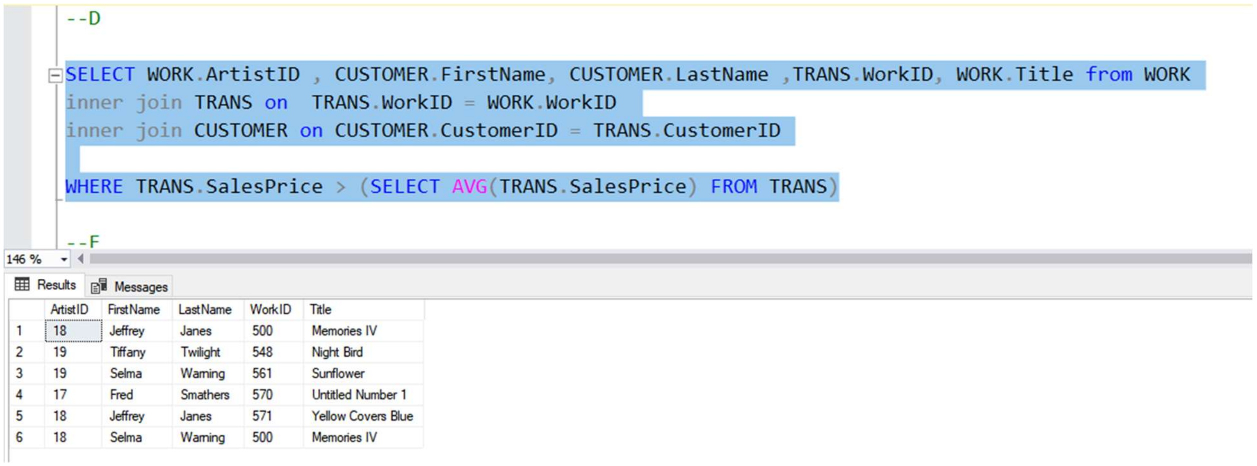
```
--C

SELECT  DATEPART(YEAR , TRANS.DateSold) as Year, CUSTOMER_ARTIST_INT.ArtistID, SUM(TRANS.SalesPrice) as SumOfSubTotal, AVG (TRANS.SalesPrice) as AverageOfSubTotal
from TRANS
Inner join CUSTOMER_ARTIST_INT on CUSTOMER_ARTIST_INT.CustomerID = TRANS.CustomerID
group by ArtistID ,  DATEPART(YEAR , TRANS.DateSold)
```

100 %

Results | Messages

	Year	ArtistID	SumOfSubTotal	AverageOfSubTotal
1	2014	11	500.00	500.000000
2	2014	17	43000.00	21500.000000
3	2014	18	43000.00	21500.000000
4	2014	19	43000.00	21500.000000
5	2015	1	1800.00	300.000000
6	2015	2	1800.00	300.000000
7	2015	4	1800.00	300.000000
8	2015	5	1800.00	300.000000
9	2015	11	28100.00	7025.000000
10	2015	17	30250.00	15125.000000
11	2015	18	30250.00	15125.000000
12	2015	19	30250.00	15125.000000
13	2016	1	575.00	287.500000
14	2016	2	575.00	287.500000
15	2016	4	575.00	287.500000
16	2016	5	91025.00	15170.833333
17	2016	11	91025.00	15170.833333
18	2016	17	69550.00	17387.500000
19	2016	18	69550.00	17387.500000

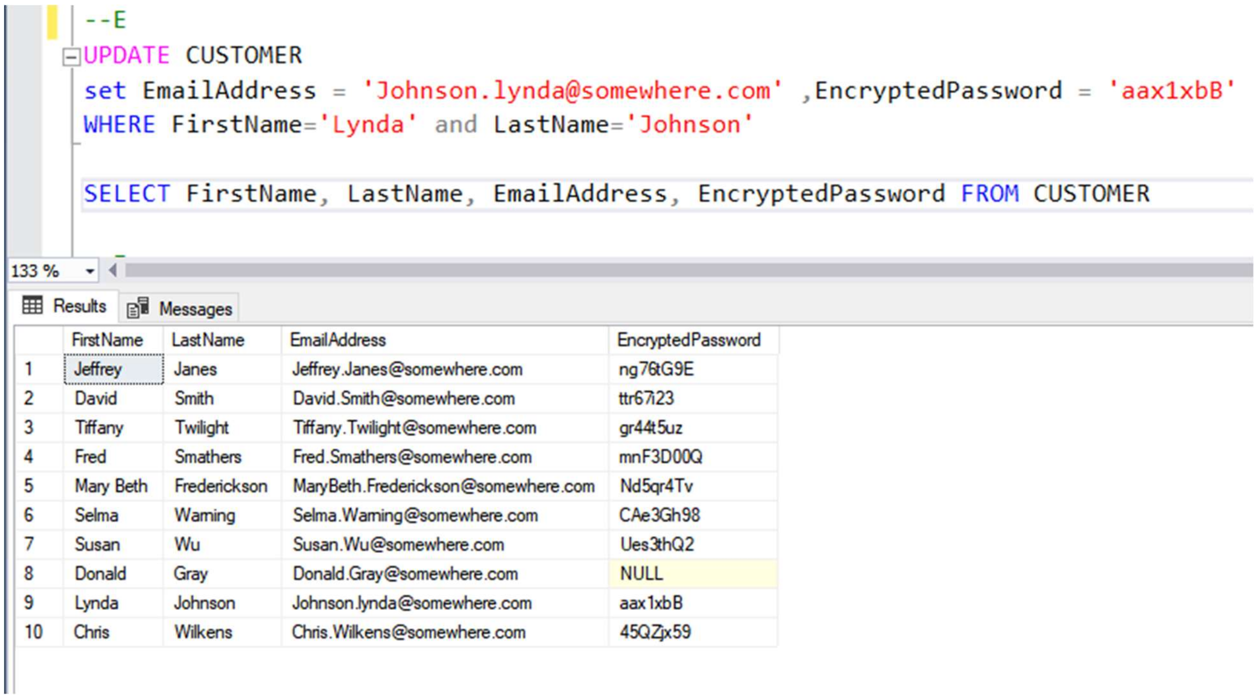
D) We have calculated the artists that have an artwork sold with a SalesPrice above the average SalesPrice by calculating the average of the sales price first then we compared the sales price with the average using (>) comparator then we have displayed the required columns



```
--D

SELECT WORK.ArtistID , CUSTOMER.FirstName, CUSTOMER.LastName ,TRANS.WorkID, WORK.Title from WORK
inner join TRANS on  TRANS.WorkID = WORK.WorkID
inner join CUSTOMER on CUSTOMER.CustomerID = TRANS.CustomerID

WHERE TRANS.SalesPrice > (SELECT AVG(TRANS.SalesPrice) FROM TRANS)

--F
```

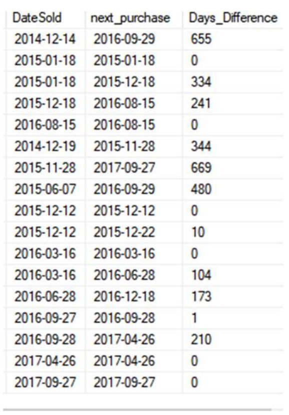
	ArtistID	FirstName	LastName	WorkID	Title
1	18	Jeffrey	Janes	500	Memories IV
2	19	Tiffany	Twilight	548	Night Bird
3	19	Selma	Warning	561	Sunflower
4	17	Fred	Smathers	570	Untitled Number 1
5	18	Jeffrey	Janes	571	Yellow Covers Blue
6	18	Selma	Warning	500	Memories IV

E) Here we have changed the values of the email address and the encrypted password for customer called Lynda Johnson



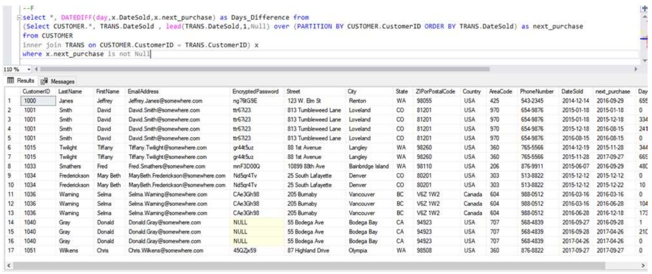
```
--E
UPDATE CUSTOMER
set EmailAddress = 'Johnson.lynda@somewhere.com' ,EncryptedPassword = 'aax1xbB'
WHERE FirstName='Lynda' and LastName='Johnson'

SELECT FirstName, LastName, EmailAddress, EncryptedPassword FROM CUSTOMER
```

	FirstName	LastName	EmailAddress	EncryptedPassword
1	Jeffrey	Janes	Jeffrey.Janes@somewhere.com	ng76tG9E
2	David	Smith	David.Smith@somewhere.com	ttr67i23
3	Tiffany	Twilight	Tiffany.Twilight@somewhere.com	gr44t5uz
4	Fred	Smathers	Fred.Smathers@somewhere.com	mnF3D00Q
5	Mary Beth	Frederickson	MaryBeth.Frederickson@somewhere.com	Nd5qr4Tv
6	Selma	Warning	Selma.Warning@somewhere.com	CAe3Gh98
7	Susan	Wu	Susan.Wu@somewhere.com	Ues3thQ2
8	Donald	Gray	Donald.Gray@somewhere.com	NULL
9	Lynda	Johnson	Johnson.lynda@somewhere.com	aax1xbB
10	Chris	Wilkens	Chris.Wilkens@somewhere.com	45QZjx59



DateSold	next_purchase	Days_Difference
2014-12-14	2016-09-29	655
2015-01-18	2015-01-18	0
2015-01-18	2015-12-18	334
2015-12-18	2016-08-15	241
2016-08-15	2016-08-15	0
2014-12-19	2015-11-28	344
2015-11-28	2017-09-27	669
2015-06-07	2016-09-29	480
2015-12-12	2015-12-12	0
2015-12-12	2015-12-22	10
2016-03-16	2016-03-16	0
2016-03-16	2016-06-28	104
2016-06-28	2016-12-18	173
2016-09-27	2016-09-28	1
2016-09-28	2017-04-26	210
2017-04-26	2017-04-26	0
2017-09-27	2017-09-27	0

F) We have used DATEDIFF function to know the difference between the data sold and the next purchase and we have renamed it as Days_Differance, then we have used Lead function and gave the lead to DateSold over the customer Id ordered by the date sold then we renamed it as the next_purchase and we have added another condition which is "next purchase is not null"



```
--F
select *, DATEDIFF(day,x.DateSold,x.next_purchase) as Days_Difference from
(Select CUSTOMER.*, TRANS.DateSold , lead(TRANS.DateSold,1,Null) over (PARTITION BY CUSTOMER.CustomerID ORDER BY TRANS.DateSold) as next_purchase
from CUSTOMER
inner join TRANS on CUSTOMER.CustomerID = TRANS.CustomerID) x
where x.next_purchase is not Null
```

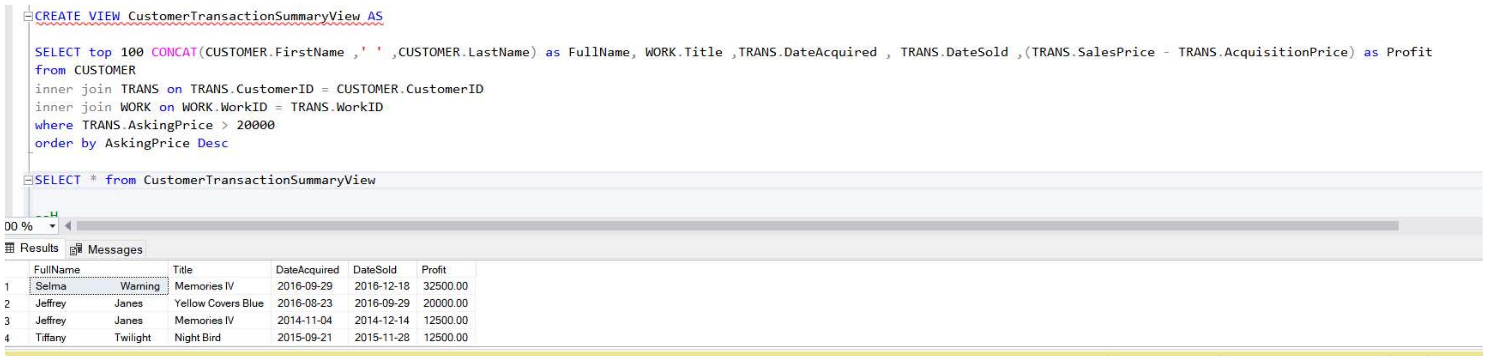
110 %

Results | Messages

	CustomerID	LastName	FirstName	EmailAddress	EncryptedPassword	Street	City	State	ZIPorPostalCode	Country	AreaCode	PhoneNumber	DateSold	next_purchase	Day
1	1000	Janes	Jeffrey	Jeffrey.Janes@somewhere.com	ng76tG9E	123 W. Elm St	Renton	WA	98055	USA	425	543-2345	2014-12-14	2016-09-29	655
2	1001	Smith	David	David.Smith@somewhere.com	ttr67i23	813 Tumbleweed Lane	Loveland	CO	81201	USA	970	654-9876	2015-01-18	2015-01-18	0
3	1001	Smith	David	David.Smith@somewhere.com	ttr67i23	813 Tumbleweed Lane	Loveland	CO	81201	USA	970	654-9876	2015-01-18	2015-12-18	334
4	1001	Smith	David	David.Smith@somewhere.com	ttr67i23	813 Tumbleweed Lane	Loveland	CO	81201	USA	970	654-9876	2015-12-18	2016-08-15	241
5	1001	Smith	David	David.Smith@somewhere.com	ttr67i23	813 Tumbleweed Lane	Loveland	CO	81201	USA	970	654-9876	2016-08-15	2016-08-15	0
6	1015	Twilight	Tiffany	Tiffany.Twilight@somewhere.com	gr44t5uz	88 1st Avenue	Langley	WA	98260	USA	360	765-5566	2014-12-19	2015-11-28	344
7	1015	Twilight	Tiffany	Tiffany.Twilight@somewhere.com	gr44t5uz	88 1st Avenue	Langley	WA	98260	USA	360	765-5566	2015-11-28	2017-09-27	66
8	1033	Smathers	Fred	Fred.Smathers@somewhere.com	mnF3D00Q	10899 88th Ave	Bainbridge Island	WA	98110	USA	206	876-9911	2015-06-07	2016-09-29	48
9	1034	Frederickson	Mary Beth	MaryBeth.Frederickson@somewhere.com	Nd5qr4Tv	25 South Lafayette	Denver	CO	80201	USA	303	513-8822	2015-12-12	2015-12-12	0
10	1034	Frederickson	Mary Beth	MaryBeth.Frederickson@somewhere.com	Nd5qr4Tv	25 South Lafayette	Denver	CO	80201	USA	303	513-8822	2015-12-12	2015-12-22	10
11	1036	Warning	Selma	Selma.Warning@somewhere.com	CAe3Gh98	205 Burnaby	Vancouver	BC	V6Z 1W2	Canada	604	988-0512	2016-03-16	2016-03-16	0
12	1036	Warning	Selma	Selma.Warning@somewhere.com	CAe3Gh98	205 Burnaby	Vancouver	BC	V6Z 1W2	Canada	604	988-0512	2016-03-16	2016-06-28	104
13	1036	Warning	Selma	Selma.Warning@somewhere.com	CAe3Gh98	205 Burnaby	Vancouver	BC	V6Z 1W2	Canada	604	988-0512	2016-06-28	2016-12-18	17
14	1040	Gray	Donald	Donald.Gray@somewhere.com	NULL	55 Bodega Ave	Bodega Bay	CA	94923	USA	707	568-4839	2016-09-27	2016-09-28	1
15	1040	Gray	Donald	Donald.Gray@somewhere.com	NULL	55 Bodega Ave	Bodega Bay	CA	94923	USA	707	568-4839	2016-09-28	2017-04-26	21
16	1040	Gray	Donald	Donald.Gray@somewhere.com	NULL	55 Bodega Ave	Bodega Bay	CA	94923	USA	707	568-4839	2017-04-26	2017-04-26	0
17	1051	Wilkens	Chris	Chris.Wilkens@somewhere.com	45QZjx59	87 Highland Drive	Olympia	WA	98508	USA	360	876-8822	2017-09-27	2017-09-27	0

G)

we have created view "virtual table" called CustomerTransactionSummaryView, which view the customer's full name using CONCAT function, we have also calculated the profit by finding the difference between AcquisitionPrice and SalesPrice, we have joined table TRANS and CUSTOMER by the common key and table WORK and TRANS by the common key to put a condition which is the asking price in TRANS table is >20000 and ordered descending, and finally we have displayed the required attributes from table customer.



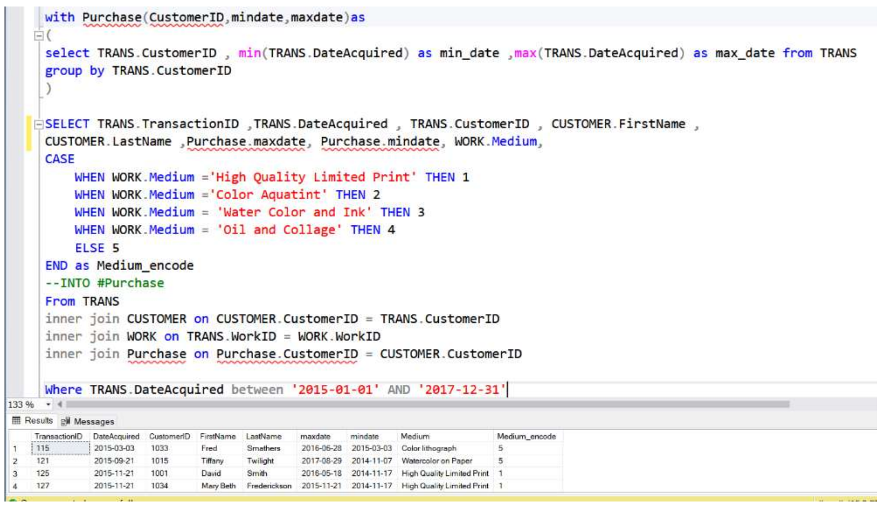
```
CREATE VIEW CustomerTransactionSummaryView AS

SELECT top 100 CONCAT(CUSTOMER.FirstName ,' ' ,CUSTOMER.LastName) as FullName, WORK.Title ,TRANS.DateAcquired , TRANS.DateSold ,(TRANS.SalesPrice - TRANS.AcquisitionPrice) as Profit
from CUSTOMER
inner join TRANS on TRANS.CustomerID = CUSTOMER.CustomerID
inner join WORK on WORK.WorkID = TRANS.WorkID
where TRANS.AskingPrice > 20000
order by AskingPrice Desc

SELECT * from CustomerTransactionSummaryView
```

00 %

Results | Messages

	FullName	Title	DateAcquired	DateSold	Profit
1	Selma Warning	Memories IV	2016-09-29	2016-12-18	32500.00
2	Jeffrey Janes	Yellow Covers Blue	2016-08-23	2016-09-29	20000.00
3	Jeffrey Janes	Memories IV	2014-11-04	2014-12-14	12500.00
4	Tiffany Twilight	Night Bird	2015-09-21	2015-11-28	12500.00

H)



```
with Purchase(CustomerID,mindate,maxdate)as
(
select TRANS.CustomerID , min(TRANS.DateAcquired) as min_date ,max(TRANS.DateAcquired) as max_date from TRANS
group by TRANS.CustomerID
)

SELECT TRANS.TransactionID ,TRANS.DateAcquired , TRANS.CustomerID , CUSTOMER.FirstName ,
CUSTOMER.LastName ,Purchase.maxdate, Purchase.mindate, WORK.Medium,
CASE
    WHEN WORK.Medium ='High Quality Limited Print' THEN 1
    WHEN WORK.Medium ='Color Aquatint' THEN 2
    WHEN WORK.Medium = 'Water Color and Ink' THEN 3
    WHEN WORK.Medium = 'Oil and Collage' THEN 4
    ELSE 5
END as Medium_encode
--INTO #Purchase
From TRANS
inner join CUSTOMER on CUSTOMER.CustomerID = TRANS.CustomerID
inner join WORK on TRANS.WorkID = WORK.WorkID
inner join Purchase on Purchase.CustomerID = CUSTOMER.CustomerID

Where TRANS.DateAcquired between '2015-01-01' AND '2017-12-31'
```

133 %

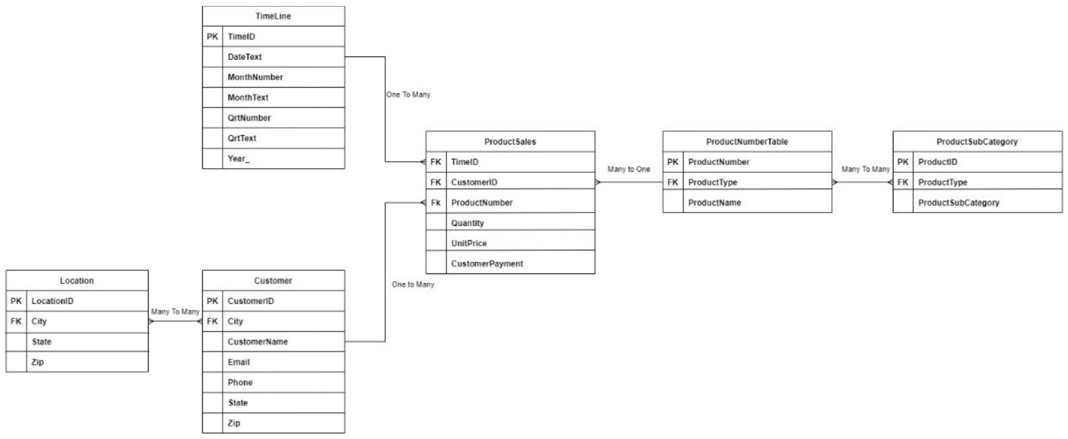
Results | Messages

	TransactionID	DateAcquired	CustomerID	FirstName	LastName	maxdate	mindate	Medium	Medium_encode
1	115	2015-03-03	1033	Fred	Smathers	2016-06-28	2015-03-03	Color lithograph	5
2	121	2015-09-21	1015	Tiffany	Twilight	2017-08-29	2014-11-07	Watercolor on Paper	5
3	125	2015-11-21	1001	David	Smith	2016-05-18	2014-11-17	High Quality Limited Print	1
4	127	2015-11-21	1034	Mary Beth	Frederickson	2015-11-21	2014-11-17	High Quality Limited Print	1

Part B

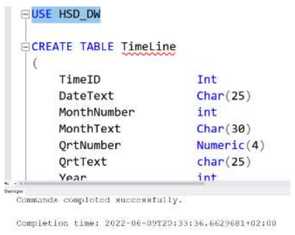
1:

Snowflake schema for the data warehouse: our main table is product sales and we have connected it with the other tables by their primary keys as a foreign key in product sales table, we have also taken table product sub category's primary key as a foreign key in table product number table & location's primary key as a foreign key in table customer.



2:

We have used HSD_DW data base, then we have created table called timeline

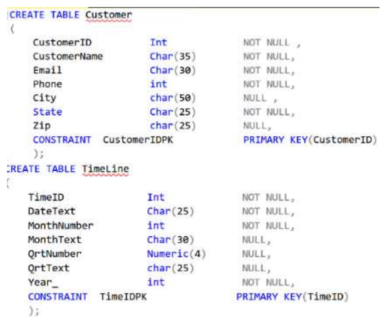


```
USE HSD_DW

CREATE TABLE TimeLine
(
    TimeID              Int
    DateText            Char(25)
    MonthNumber         int
    MonthText           Char(30)
    QrtNumber           Numeric(4)
    QrtText             char(25)
    Year                int
```

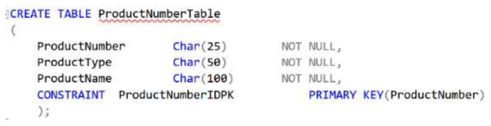
```
Commands completed successfully.

Completion time: 2022-06-09T20:33:36.6629681+02:00
```

And we have also created four tables which are (Customer, Timeline, ProductNumberTable, and ProductSales).

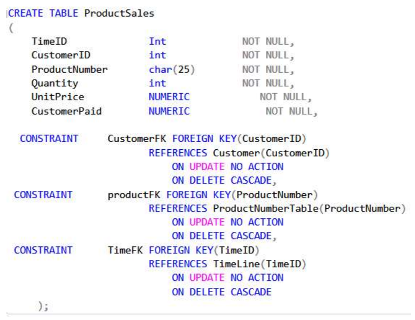
We have used constraint key word to identify the primary key for each table, and in product sales table we have collected the primary keys of all tables as foreign keys in our table



```
CREATE TABLE Customer
(
    CustomerID          Int             NOT NULL ,
    CustomerName        Char(35)        NOT NULL,
    Email               Char(30)        NOT NULL,
    Phone               int             NOT NULL,
    City                char(50)        NULL ,
    State               Char(25)        NOT NULL,
    Zip                 char(25)        NULL,
    CONSTRAINT  CustomerIDPK            PRIMARY KEY(CustomerID)
    );
CREATE TABLE TimeLine
(
    TimeID              Int             NOT NULL,
    DateText            Char(25)        NOT NULL,
    MonthNumber         int             NOT NULL,
    MonthText           Char(30)        NULL,
    QrtNumber           Numeric(4)      NULL,
    QrtText             char(25)        NULL,
    Year_               int             NOT NULL,
    CONSTRAINT  TimeIDPK               PRIMARY KEY(TimeID)
    );
```

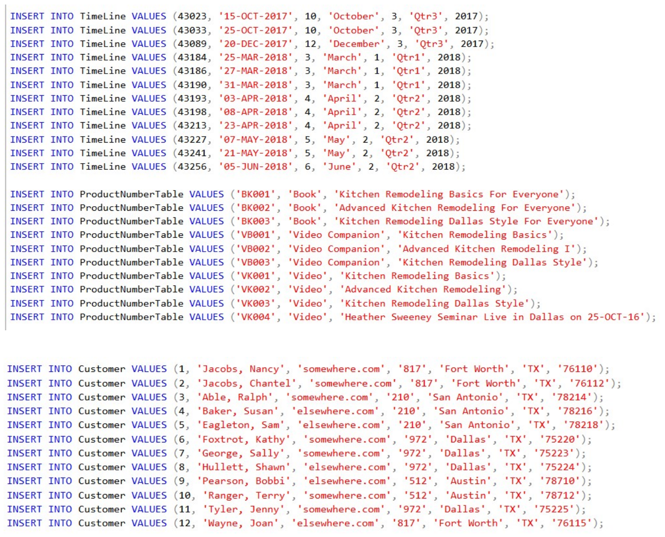


```
CREATE TABLE ProductNumberTable
(
    ProductNumber       Char(25)        NOT NULL,
    ProductType         Char(50)        NOT NULL,
    ProductName         Char(100)       NOT NULL,
    CONSTRAINT  ProductNumberIDPK         PRIMARY KEY(ProductNumber)
    );
```

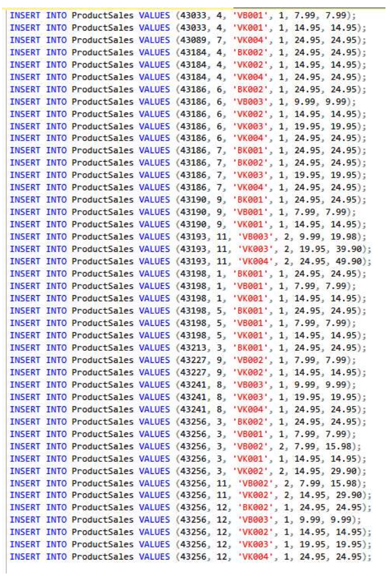


```
CREATE TABLE ProductSales
(
    TimeID              Int             NOT NULL,
    CustomerID          int             NOT NULL,
    ProductNumber       char(25)        NOT NULL,
    Quantity            int             NOT NULL,
    UnitPrice           NUMERIC            NOT NULL,
    CustomerPaid        NUMERIC             NOT NULL,

 CONSTRAINT     CustomerFK FOREIGN KEY(CustomerID)
                    REFERENCES Customer(CustomerID)
                        ON UPDATE NO ACTION
                        ON DELETE CASCADE,
 CONSTRAINT     productFK FOREIGN KEY(ProductNumber)
                    REFERENCES ProductNumberTable(ProductNumber)
                        ON UPDATE NO ACTION
                        ON DELETE CASCADE,
 CONSTRAINT     TimeFK FOREIGN KEY(TimeID)
                    REFERENCES TimeLine(TimeID)
                        ON UPDATE NO ACTION
                        ON DELETE CASCADE
    );
```

Inserting data from given files:

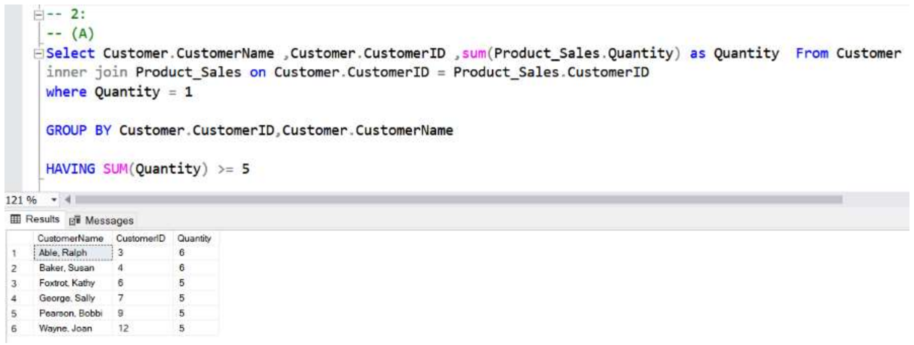


```
INSERT INTO TimeLine VALUES (43023, '15-OCT-2017', 10, 'October', 3, 'Qtr3', 2017);
INSERT INTO TimeLine VALUES (43033, '25-OCT-2017', 10, 'October', 3, 'Qtr3', 2017);
INSERT INTO TimeLine VALUES (43089, '20-DEC-2017', 12, 'December', 3, 'Qtr3', 2017);
INSERT INTO TimeLine VALUES (43184, '25-MAR-2018', 3, 'March', 1, 'Qtr1', 2018);
INSERT INTO TimeLine VALUES (43186, '27-MAR-2018', 3, 'March', 1, 'Qtr1', 2018);
INSERT INTO TimeLine VALUES (43190, '31-MAR-2018', 3, 'March', 1, 'Qtr1', 2018);
INSERT INTO TimeLine VALUES (43193, '03-APR-2018', 4, 'April', 2, 'Qtr2', 2018);
INSERT INTO TimeLine VALUES (43198, '08-APR-2018', 4, 'April', 2, 'Qtr2', 2018);
INSERT INTO TimeLine VALUES (43213, '23-APR-2018', 4, 'April', 2, 'Qtr2', 2018);
INSERT INTO TimeLine VALUES (43227, '07-MAY-2018', 5, 'May', 2, 'Qtr2', 2018);
INSERT INTO TimeLine VALUES (43241, '21-MAY-2018', 5, 'May', 2, 'Qtr2', 2018);
INSERT INTO TimeLine VALUES (43256, '05-JUN-2018', 6, 'June', 2, 'Qtr2', 2018);

INSERT INTO ProductNumberTable VALUES ('BK001', 'Book', 'Kitchen Remodeling Basics For Everyone');
INSERT INTO ProductNumberTable VALUES ('BK002', 'Book', 'Advanced Kitchen Remodeling For Everyone');
INSERT INTO ProductNumberTable VALUES ('BK003', 'Book', 'Kitchen Remodeling Dallas Style For Everyone');
INSERT INTO ProductNumberTable VALUES ('VB001', 'Video Companion', 'Kitchen Remodeling Basics');
INSERT INTO ProductNumberTable VALUES ('VB002', 'Video Companion', 'Advanced Kitchen Remodeling I');
INSERT INTO ProductNumberTable VALUES ('VB003', 'Video Companion', 'Kitchen Remodeling Dallas Style');
INSERT INTO ProductNumberTable VALUES ('VK001', 'Video', 'Kitchen Remodeling Basics');
INSERT INTO ProductNumberTable VALUES ('VK002', 'Video', 'Advanced Kitchen Remodeling');
INSERT INTO ProductNumberTable VALUES ('VK003', 'Video', 'Kitchen Remodeling Dallas Style');
INSERT INTO ProductNumberTable VALUES ('VK004', 'Video', 'Heather Sweeney Seminar Live in Dallas on 25-OCT-16');


INSERT INTO Customer VALUES (1, 'Jacobs, Nancy', 'somewhere.com', '817', 'Fort Worth', 'TX', '76110');
INSERT INTO Customer VALUES (2, 'Jacobs, Chantel', 'somewhere.com', '817', 'Fort Worth', 'TX', '76112');
INSERT INTO Customer VALUES (3, 'Able, Ralph', 'somewhere.com', '210', 'San Antonio', 'TX', '78214');
INSERT INTO Customer VALUES (4, 'Baker, Susan', 'elsewhere.com', '210', 'San Antonio', 'TX', '78216');
INSERT INTO Customer VALUES (5, 'Eagleton, Sam', 'elsewhere.com', '210', 'San Antonio', 'TX', '78218');
INSERT INTO Customer VALUES (6, 'Foxtrot, Kathy', 'somewhere.com', '972', 'Dallas', 'TX', '75220');
INSERT INTO Customer VALUES (7, 'George, Sally', 'somewhere.com', '972', 'Dallas', 'TX', '75223');
INSERT INTO Customer VALUES (8, 'Hullett, Shawn', 'elsewhere.com', '972', 'Dallas', 'TX', '75224');
INSERT INTO Customer VALUES (9, 'Pearson, Bobbi', 'elsewhere.com', '512', 'Austin', 'TX', '78710');
INSERT INTO Customer VALUES (10, 'Ranger, Terry', 'somewhere.com', '512', 'Austin', 'TX', '78712');
INSERT INTO Customer VALUES (11, 'Tyler, Jenny', 'somewhere.com', '972', 'Dallas', 'TX', '75225');
INSERT INTO Customer VALUES (12, 'Wayne, Joan', 'elsewhere.com', '817', 'Fort Worth', 'TX', '76115');
```



```
INSERT INTO ProductSales VALUES (43033, 4, 'VB001', 1, 7.99, 7.99);
INSERT INTO ProductSales VALUES (43033, 4, 'VK001', 1, 14.95, 14.95);
INSERT INTO ProductSales VALUES (43089, 7, 'VK004', 1, 24.95, 24.95);
INSERT INTO ProductSales VALUES (43184, 4, 'BK002', 1, 24.95, 24.95);
INSERT INTO ProductSales VALUES (43184, 4, 'VK002', 1, 14.95, 14.95);
INSERT INTO ProductSales VALUES (43184, 4, 'VK004', 1, 24.95, 24.95);
INSERT INTO ProductSales VALUES (43186, 6, 'BK002', 1, 24.95, 24.95);
INSERT INTO ProductSales VALUES (43186, 6, 'VB003', 1, 9.99, 9.99);
INSERT INTO ProductSales VALUES (43186, 6, 'VK002', 1, 14.95, 14.95);
INSERT INTO ProductSales VALUES (43186, 6, 'VK003', 1, 19.95, 19.95);
INSERT INTO ProductSales VALUES (43186, 6, 'VK004', 1, 24.95, 24.95);
INSERT INTO ProductSales VALUES (43186, 7, 'BK001', 1, 24.95, 24.95);
INSERT INTO ProductSales VALUES (43186, 7, 'BK002', 1, 24.95, 24.95);
INSERT INTO ProductSales VALUES (43186, 7, 'VK003', 1, 19.95, 19.95);
INSERT INTO ProductSales VALUES (43186, 7, 'VK004', 1, 24.95, 24.95);
INSERT INTO ProductSales VALUES (43190, 9, 'BK001', 1, 24.95, 24.95);
INSERT INTO ProductSales VALUES (43190, 9, 'VB001', 1, 7.99, 7.99);
INSERT INTO ProductSales VALUES (43190, 9, 'VK001', 1, 14.95, 14.95);
INSERT INTO ProductSales VALUES (43193, 11, 'VB003', 2, 9.99, 19.98);
INSERT INTO ProductSales VALUES (43193, 11, 'VK003', 2, 19.95, 39.90);
INSERT INTO ProductSales VALUES (43193, 11, 'VK004', 2, 24.95, 49.90);
INSERT INTO ProductSales VALUES (43198, 1, 'BK001', 1, 24.95, 24.95);
INSERT INTO ProductSales VALUES (43198, 1, 'VB001', 1, 7.99, 7.99);
INSERT INTO ProductSales VALUES (43198, 1, 'VK001', 1, 14.95, 14.95);
INSERT INTO ProductSales VALUES (43198, 5, 'BK001', 1, 24.95, 24.95);
INSERT INTO ProductSales VALUES (43198, 5, 'VB001', 1, 7.99, 7.99);
INSERT INTO ProductSales VALUES (43198, 5, 'VK001', 1, 14.95, 14.95);
INSERT INTO ProductSales VALUES (43213, 3, 'BK001', 1, 24.95, 24.95);
INSERT INTO ProductSales VALUES (43227, 9, 'VB002', 1, 7.99, 7.99);
INSERT INTO ProductSales VALUES (43227, 9, 'VK002', 1, 14.95, 14.95);
INSERT INTO ProductSales VALUES (43241, 8, 'VB003', 1, 9.99, 9.99);
INSERT INTO ProductSales VALUES (43241, 8, 'VK003', 1, 19.95, 19.95);
INSERT INTO ProductSales VALUES (43241, 8, 'VK004', 1, 24.95, 24.95);
INSERT INTO ProductSales VALUES (43256, 3, 'BK002', 1, 24.95, 24.95);
INSERT INTO ProductSales VALUES (43256, 3, 'VB001', 1, 7.99, 7.99);
INSERT INTO ProductSales VALUES (43256, 3, 'VB002', 2, 7.99, 15.98);
INSERT INTO ProductSales VALUES (43256, 3, 'VK001', 1, 14.95, 14.95);
INSERT INTO ProductSales VALUES (43256, 3, 'VK002', 2, 14.95, 29.90);
INSERT INTO ProductSales VALUES (43256, 11, 'VB002', 2, 7.99, 15.98);
INSERT INTO ProductSales VALUES (43256, 11, 'VK002', 2, 14.95, 29.90);
INSERT INTO ProductSales VALUES (43256, 12, 'BK002', 1, 24.95, 24.95);
INSERT INTO ProductSales VALUES (43256, 12, 'VB003', 1, 9.99, 9.99);
INSERT INTO ProductSales VALUES (43256, 12, 'VK002', 1, 14.95, 14.95);
INSERT INTO ProductSales VALUES (43256, 12, 'VK003', 1, 19.95, 19.95);
INSERT INTO ProductSales VALUES (43256, 12, 'VK004', 1, 24.95, 24.95);
```

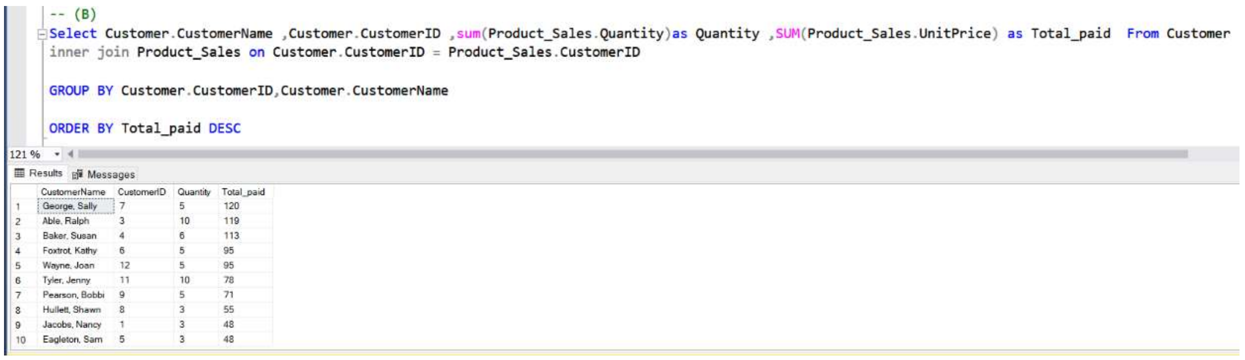
A) An order containing at least five products with different product numbers



```
-- 2:
-- (A)
Select Customer.CustomerName ,Customer.CustomerID ,sum(Product_Sales.Quantity) as Quantity  From Customer
inner join Product_Sales on Customer.CustomerID = Product_Sales.CustomerID
where Quantity = 1

GROUP BY Customer.CustomerID,Customer.CustomerName

HAVING SUM(Quantity) >= 5
```

	CustomerName	CustomerID	Quantity
1	Able, Ralph	3	6
2	Baker, Susan	4	6
3	Foxtrot, Kathy	6	5
4	George, Sally	7	5
5	Pearson, Bobbi	9	5
6	Wayne, Joan	12	5

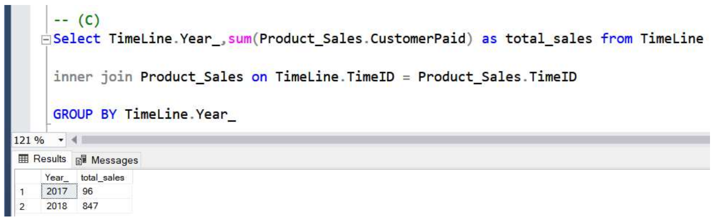
B) The customers that made the largest order "the largest bill"



```
-- (B)
Select Customer.CustomerName ,Customer.CustomerID ,sum(Product_Sales.Quantity)as Quantity ,SUM(Product_Sales.UnitPrice) as Total_paid  From Customer
inner join Product_Sales on Customer.CustomerID = Product_Sales.CustomerID

GROUP BY Customer.CustomerID,Customer.CustomerName

ORDER BY Total_paid DESC
```

	CustomerName	CustomerID	Quantity	Total_paid
1	George, Sally	7	5	120
2	Able, Ralph	3	10	119
3	Baker, Susan	4	6	113
4	Foxtrot, Kathy	6	5	95
5	Wayne, Joan	12	5	95
6	Tyler, Jenny	11	10	78
7	Pearson, Bobbi	9	5	71
8	Hullett, Shawn	8	3	55
9	Jacobs, Nancy	1	3	48
10	Eagleton, Sam	5	3	48

C) Calculating sales per year

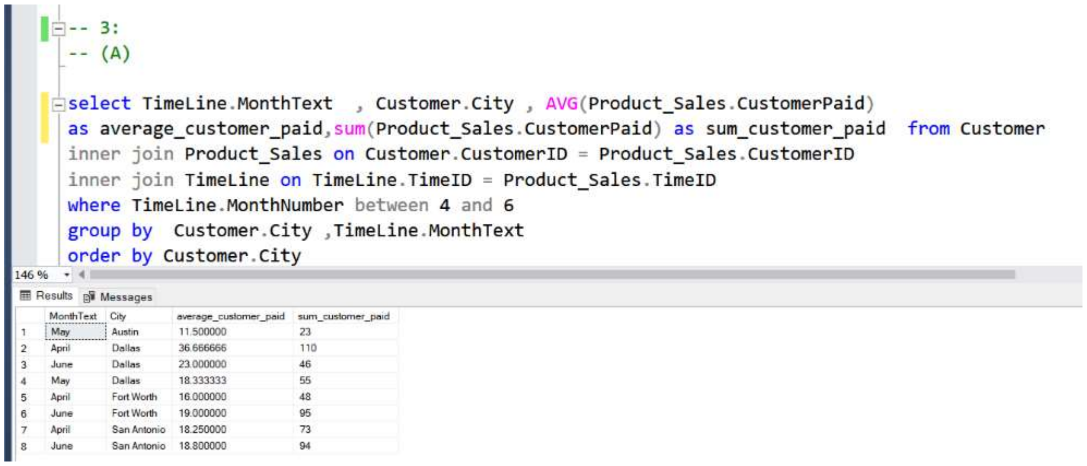


```
-- (C)
Select TimeLine.Year_,sum(Product_Sales.CustomerPaid) as total_sales from TimeLine

inner join Product_Sales on TimeLine.TimeID = Product_Sales.TimeID

GROUP BY TimeLine.Year_
```

	Year_	total_sales
1	2017	96
2	2018	847

3:

A) We are figuring out where is the drop, and we have found that there is a decrease in Dallas city in sum of customers' payments through the specified 3 months.



```sql
-- 3:
-- (A)

select TimeLine.MonthText  , Customer.City , AVG(Product_Sales.CustomerPaid)
as average_customer_paid,sum(Product_Sales.CustomerPaid) as sum_customer_paid  from Customer
inner join Product_Sales on Customer.CustomerID = Product_Sales.CustomerID
inner join TimeLine on TimeLine.TimeID = Product_Sales.TimeID
where TimeLine.MonthNumber between 4 and 6
group by  Customer.City ,TimeLine.MonthText
order by Customer.City
```

	MonthText	City	average_customer_paid	sum_customer_paid
1	May	Austin	11.500000	23
2	April	Dallas	36.666666	110
3	June	Dallas	23.000000	46
4	May	Dallas	18.333333	55
5	April	Fort Worth	16.000000	48
6	June	Fort Worth	19.000000	95
7	April	San Antonio	18.250000	73
8	June	San Antonio	18.800000	94

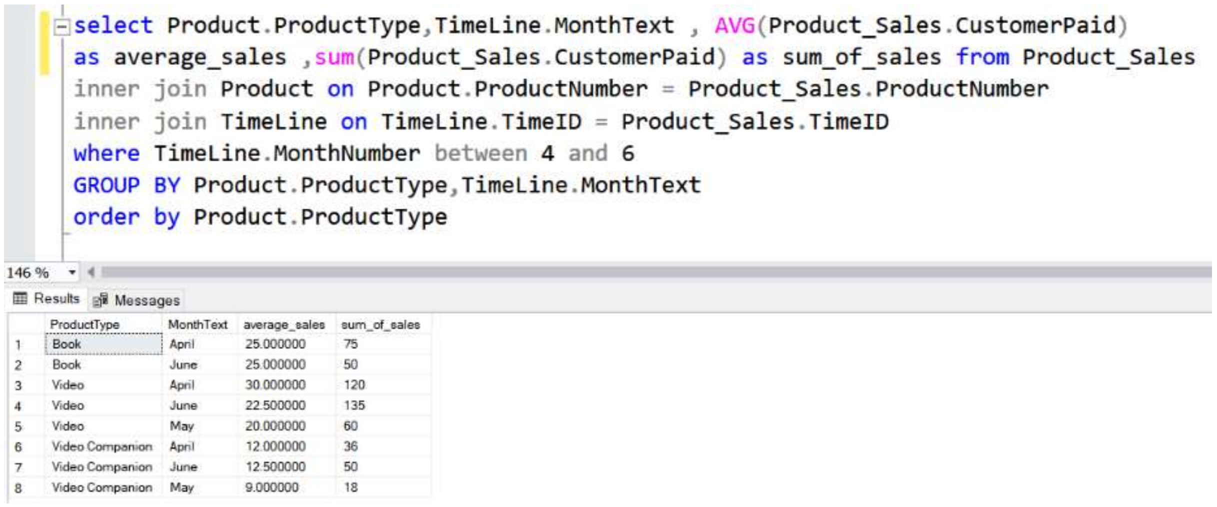
We have calculated the sum and averae of sales in the required months.

For example, in "Books" there's a decrease in sum of sales between april and june.

In "Videos" there's a huge drop in sum of sales between April and May, then a huge increase reaching the peak in June.

In "Video Companions" there's also a huge drop in sum of sales between April and May, then a great increase in June.

As previously mentioned, customers' payments was checked in cities, then, each product's sum of sales through the months.

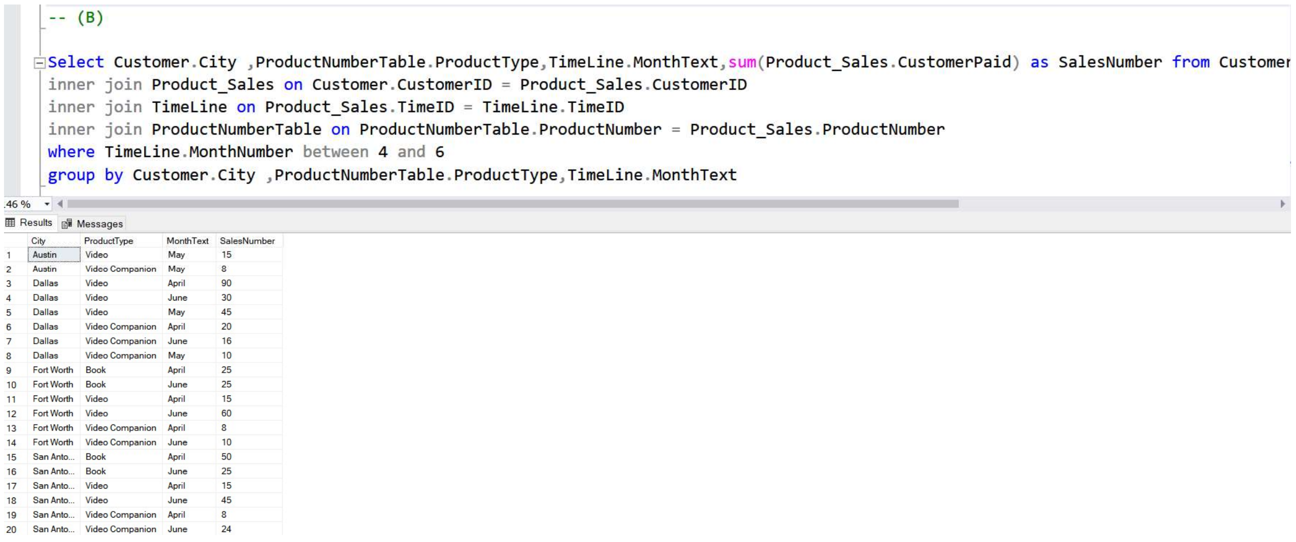


```sql
select Product.ProductType,TimeLine.MonthText , AVG(Product_Sales.CustomerPaid)
as average_sales ,sum(Product_Sales.CustomerPaid) as sum_of_sales from Product_Sales
inner join Product on Product.ProductNumber = Product_Sales.ProductNumber
inner join TimeLine on TimeLine.TimeID = Product_Sales.TimeID
where TimeLine.MonthNumber between 4 and 6
GROUP BY Product.ProductType,TimeLine.MonthText
order by Product.ProductType
```

	ProductType	MonthText	average_sales	sum_of_sales
1	Book	April	25.000000	75
2	Book	June	25.000000	50
3	Video	April	30.000000	120
4	Video	June	22.500000	135
5	Video	May	20.000000	60
6	Video Companion	April	12.000000	36
7	Video Companion	June	12.500000	50
8	Video Companion	May	9.000000	18

B) A Report for sales from each city in each month.

The highest sales are in Dallas city by selling videos in April.

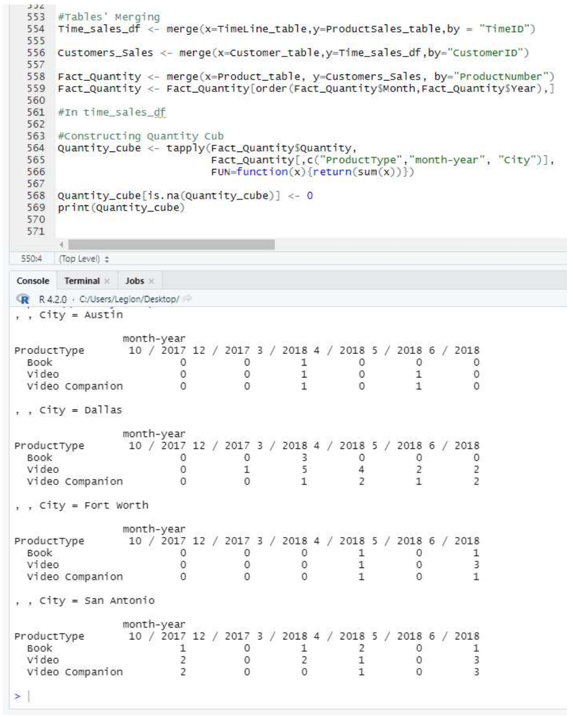


```sql
-- (B)

Select Customer.City ,ProductNumberTable.ProductType,TimeLine.MonthText,sum(Product_Sales.CustomerPaid) as SalesNumber from Customer
inner join Product_Sales on Customer.CustomerID = Product_Sales.CustomerID
inner join TimeLine on Product_Sales.TimeID = TimeLine.TimeID
inner join ProductNumberTable on ProductNumberTable.ProductNumber = Product_Sales.ProductNumber
where TimeLine.MonthNumber between 4 and 6
group by Customer.City ,ProductNumberTable.ProductType,TimeLine.MonthText
```

	City	ProductType	MonthText	SalesNumber
1	Austin	Video	May	15
2	Austin	Video Companion	May	8
3	Dallas	Video	April	90
4	Dallas	Video	June	30
5	Dallas	Video	May	45
6	Dallas	Video Companion	April	20
7	Dallas	Video Companion	June	16
8	Dallas	Video Companion	May	10
9	Fort Worth	Book	April	25
10	Fort Worth	Book	June	25
11	Fort Worth	Video	April	15
12	Fort Worth	Video	June	60
13	Fort Worth	Video Companion	April	8
14	Fort Worth	Video Companion	June	10
15	San Anto...	Book	April	50
16	San Anto...	Book	June	25
17	San Anto...	Video	April	15
18	San Anto...	Video	June	45
19	San Anto...	Video Companion	April	8
20	San Anto...	Video Companion	June	24

4:

After reading the dimensions files and product sales fact table, OLAP cube has been constructed.



```r
#Tables' Merging
Time_sales_df <- merge(x=TimeLine_table,y=ProductSales_table,by = "TimeID")

Customers_Sales <- merge(x=Customer_table,y=Time_sales_df,by="CustomerID")

Fact_Quantity <- merge(x=Product_table, y=Customers_Sales, by="ProductNumber")
Fact_Quantity <- Fact_Quantity[order(Fact_Quantity$Month,Fact_Quantity$Year),]

#In time_sales_df

#Constructing Quantity Cub
Quantity_cube <- tapply(Fact_Quantity$Quantity,
                        Fact_Quantity[,c("ProductType","month-year", "City")],
                        FUN=function(x){return(sum(x))})

Quantity_cube[is.na(Quantity_cube)] <- 0
print(Quantity_cube)
```

```
, , City = Austin

                 month-year
ProductType       10 / 2017 12 / 2017 3 / 2018 4 / 2018 5 / 2018 6 / 2018
  Book                    0         0        1        0        0        0
  Video                   0         0        1        0        1        0
  Video Companion         0         0        1        0        1        0

, , City = Dallas

                 month-year
ProductType       10 / 2017 12 / 2017 3 / 2018 4 / 2018 5 / 2018 6 / 2018
  Book                    0         0        3        0        0        0
  Video                   0         1        5        4        2        2
  Video Companion         0         0        1        2        1        2

, , City = Fort Worth

                 month-year
ProductType       10 / 2017 12 / 2017 3 / 2018 4 / 2018 5 / 2018 6 / 2018
  Book                    0         0        0        1        0        1
  Video                   0         0        0        1        0        3
  Video Companion         0         0        0        1        0        1

, , City = San Antonio

                 month-year
ProductType       10 / 2017 12 / 2017 3 / 2018 4 / 2018 5 / 2018 6 / 2018
  Book                    1         0        1        2        0        1
  Video                   2         0        2        1        0        3
  Video Companion         2         0        0        1        0        3

>
```