SQL

WEEK 6 WITH TDI

Presented By Kareem Shaaban

1

```sql
/* 1. Create a query to categorize orders based on the `Profit` column:
    - 'Loss' for profits less than 0
    - 'Low Profit' for profits between 0 and 100
    - 'High Profit' for profits greater than 100. */

SELECT Order_ID, SUM(Profit) AS PROFIT,
        CASE WHEN SUM(Profit) < 0 THEN 'Loss'
             WHEN SUM(Profit) BETWEEN 0 AND 100 THEN 'Low Profit'
             ELSE 'High Profit'
        END AS ProfitCategories
FROM SALES
GROUP BY Order_ID;
```

Results | Messages

	Order_ID	PROFIT	ProfitCategories
1	CA-2013-109806	18.0216	Low Profit
2	CA-2013-169103	81.409	Low Profit
3	CA-2012-121797	297.8469	High Profit
4	CA-2011-114517	-3.7236	Loss
5	CA-2012-166604	8.7906	Low Profit
6	CA-2012-104486	25.3814	Low Profit
7	US-2013-117541	14.1406	Low Profit
8	CA-2014-110821	-38.5469	Loss
9	CA-2012-131457	6.7116	Low Profit
10	US-2012-120572	-8.9796	Loss
11	CA-2013-130050	-3.8208	Loss
12	CA-2014-100097	302.4705	High Profit
13	CA-2013-116736	144.6682	High Profit
14	CA-2014-157091	31.587	Low Profit
15	CA-2012-168529	62.737	Low Profit
16	CA-2014-139080	31.0752	Low Profit
17	CA-2012-122946	314.6962	High Profit

Query executed successfully.

2

```sql
/* 2. Write a query to display the `Customer Name` along with
    a flag indicating if their order quantity is
    'High' (Quantity > 10) or 'Low' (Quantity<= 10).*/

SELECT Customer_Name, SUM(CONVERT(INT,Quantity)) AS Quantity,
        CASE WHEN SUM(CONVERT(INT,Quantity)) > 10 THEN 'High'
        ELSE 'Low' END AS QuantityIndicator
FROM SALES
GROUP BY Customer_Name;
```

Results | Messages

	Customer_Name	Quantity	QuantityIndicator
1	Michael Chen	45	High
2	Brian Moss	95	High
3	Tamara Chand	42	High
4	Justin MacKendrick	74	High
5	Doug Bickford	41	High
6	Jesus Ocampo	42	High
7	Sheri Gordon	62	High
8	John Huston	29	High
9	Sean Miller	50	High
10	Liz Carlisle	43	High
11	Benjamin Patterson	14	High
12	Shahid Hopkins	45	High
13	Tracy Collins	28	High
14	Eugene Moren	74	High
15	Alice McCarthy	48	High

Query executed successfully.

3

```sql
/* 3. Create a query to classify orders based on the `Ship Mode`:
    - 'Fast' for 'First Class'
    - 'Standard' for 'Standard Class'*/

SELECT Order_ID, Ship_Mode,
        CASE WHEN Ship_Mode = 'First Class' THEN 'Fast'
        ELSE 'Standard' END AS QuantityIndicator
FROM SALES;
```

Results | Messages

	Order_ID	Ship_Mode	QuantityIndicator
1	CA-2013-152156	Second Class	Standard
2	CA-2013-152156	Second Class	Standard
3	CA-2013-138688	Second Class	Standard
4	US-2012-108966	Standard Class	Standard
5	US-2012-108966	Standard Class	Standard
6	CA-2011-115812	Standard Class	Standard
7	CA-2011-115812	Standard Class	Standard
8	CA-2011-115812	Standard Class	Standard
9	CA-2011-115812	Standard Class	Standard
10	CA-2011-115812	Standard Class	Standard
11	CA-2011-115812	Standard Class	Standard
12	CA-2011-115812	Standard Class	Standard
13	CA-2014-114412	Standard Class	Standard
14	CA-2013-161389	Standard Class	Standard
15	US-2012-118983	Standard Class	Standard

Query executed successfully.

4

```sql
/* 4. Write a query to show the `Sales` along with a message indicating if
    the sales are 'Above Average' or 'Below Average', considering the
     average sales of all orders*/

SELECT Order_ID, SUM(Sales) AS TotalSales,
        CASE WHEN SUM(Sales) < (SELECT AVG(Sales)FROM SALES) THEN 'Below Average'
             ELSE 'Above Average'
        END AS SalesIndicator
FROM SALES
GROUP BY Order_ID;
```

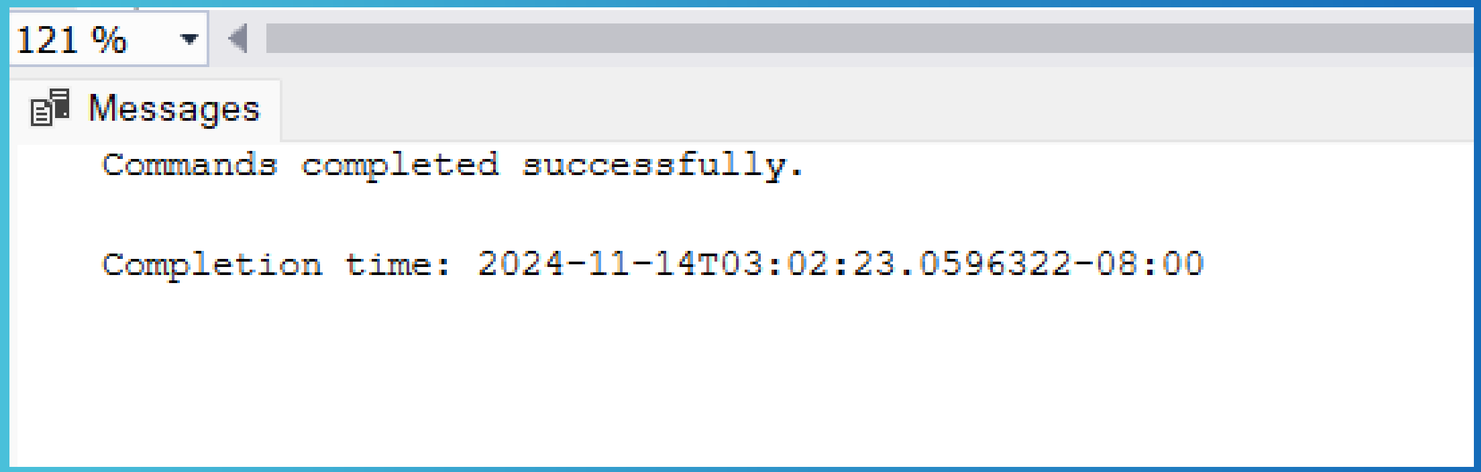
Results | Messages

	Order_ID	TotalSales	SalesIndicator
1	CA-2013-109806	100.164	Below Average
2	CA-2013-169103	1466.32	Above Average
3	CA-2012-121797	2147.336	Above Average
4	CA-2011-114517	109.836	Below Average
5	CA-2012-166604	17.94	Below Average
6	CA-2012-104486	84.556	Below Average
7	US-2013-117541	36.37	Below Average
8	CA-2014-110821	237.608	Above Average
9	CA-2012-131457	14.28	Below Average
10	US-2012-120572	12.828	Below Average
11	CA-2013-130050	9.552	Below Average
12	CA-2014-100097	1115.25	Above Average
13	CA-2013-116736	724.55	Above Average
14	CA-2014-157091	526.45	Above Average
15	CA-2012-168529	145.90	Below Average

Query executed successfully.

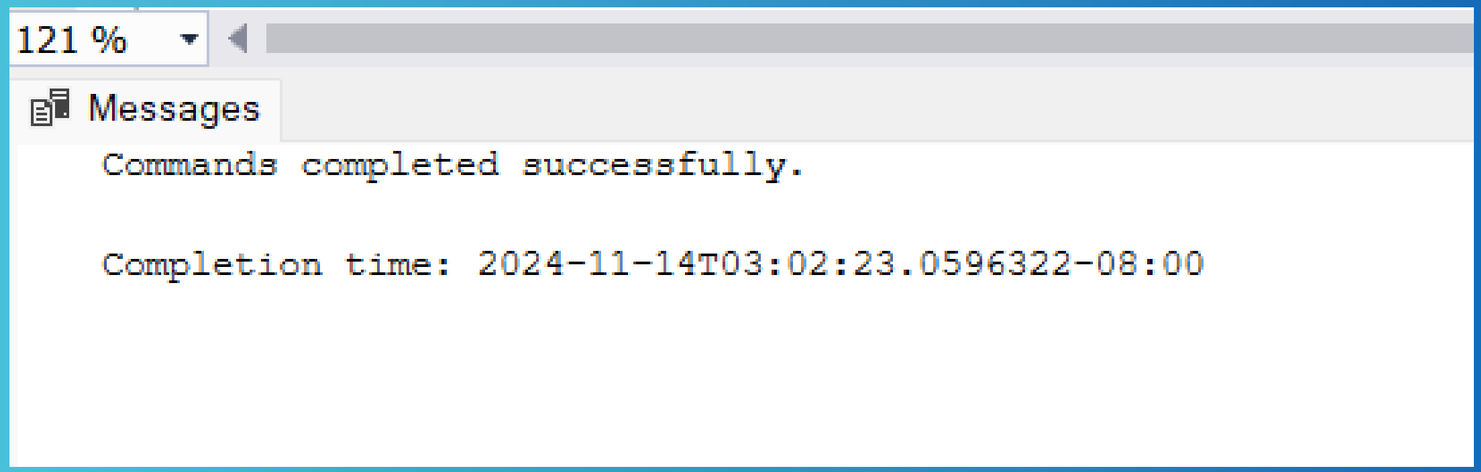
5

```sql
-- Views:

/* 1. Create a view named 'sales_summary' that shows the total sales,
    quantity, and profit for each region.*/

CREATE VIEW Sales_Summary AS
    SELECT
        Region,
        SUM(CONVERT(INT,Quantity)) AS TotalQuantity,
        SUM(CONVERT(INT,Sales)) AS TotalSales,
        SUM(CONVERT(INT,Profit)) AS TotaLProfit
    FROM SALES
    GROUP BY Region;
```



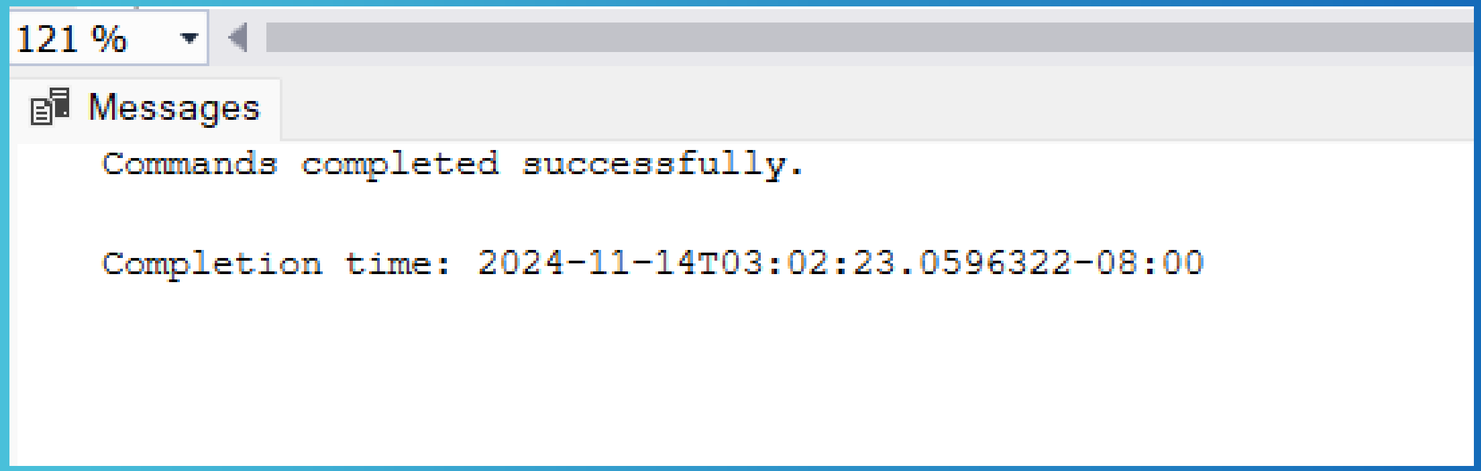
6

```sql
/* 2. Modify the 'sales_summary' view to include the category and
    subcategory columns.*/

ALTER VIEW Sales_Summary AS
    SELECT
        Region, Category, Sub_Category,
        SUM(CONVERT(INT,Quantity)) AS TotalQuantity,
        SUM(CONVERT(INT,Sales)) AS TotalSales,
        SUM(CONVERT(INT,Profit)) AS TotaLProfit
    FROM SALES
    GROUP BY Region, Category, Sub_Category;
```



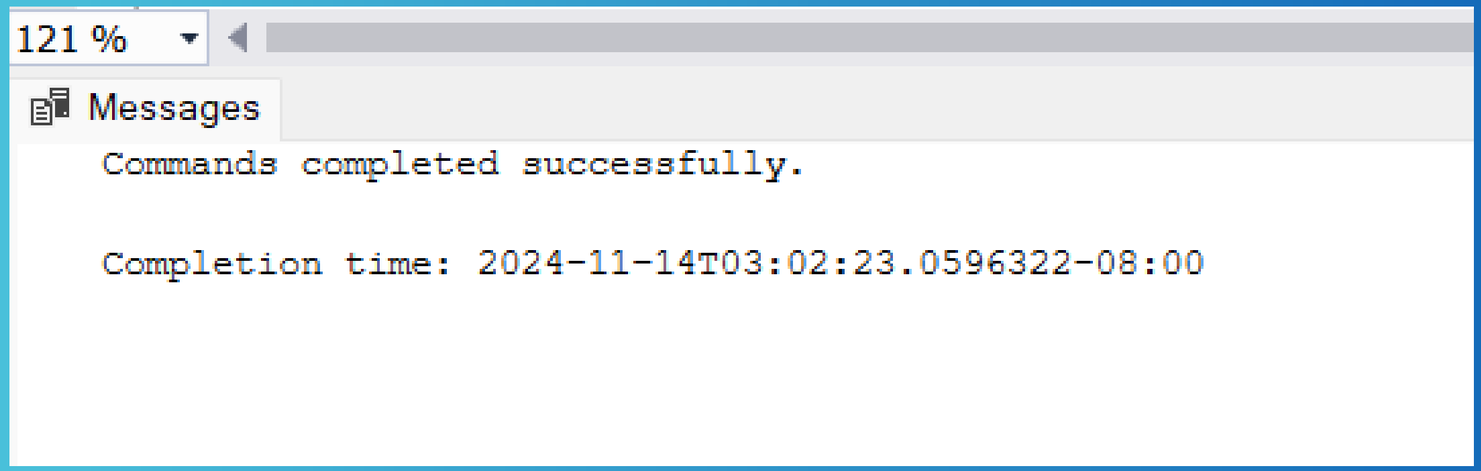
7

```sql
/* 3. Drop the ‘sales_summary’ view.*/

DROP VIEW Sales_Summary;
```



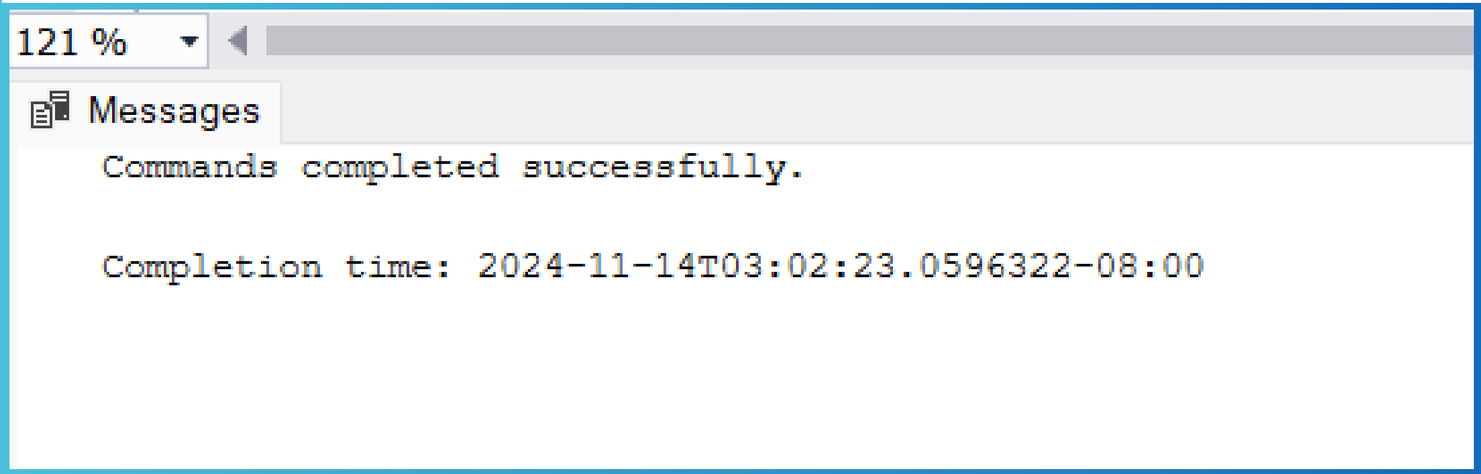
8

```sql
/* 4. Create a view named ‘customer sales’ that shows the total sales
   and profit for each customer*/

CREATE VIEW "Customer Sales" AS
   SELECT
        Customer_Name,
        SUM(CONVERT(INT,Sales)) AS TotalSales,
        SUM(CONVERT(INT,Profit)) AS TotaLProfit
   FROM SALES
   GROUP BY Customer_Name;
```



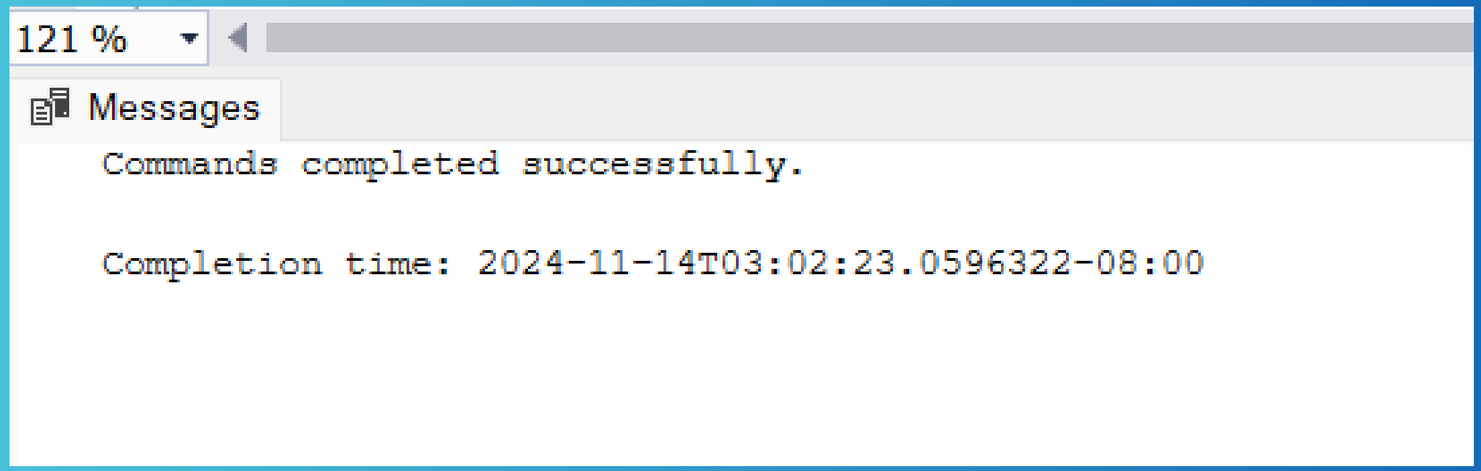
9

```sql
/* 5. Modify the ‘customer sales’ view to include
   the city and state columns.*/

ALTER VIEW "Customer Sales" AS
   SELECT
        Customer_Name, State, City,
        SUM(CONVERT(INT,Sales)) AS TotalSales,
        SUM(CONVERT(INT,Profit)) AS TotaLProfit
   FROM SALES
   GROUP BY Customer_Name, State, City;
```



10

```sql
/* 6. Create a view named ‘product_sales’ that shows the total sales
     and profit for each product category and subcategory.*/

CREATE VIEW Product_Sales AS
    SELECT
        Category, Sub_Category,
        SUM(CONVERT(INT,Sales)) AS TotalSales,
        SUM(CONVERT(INT,Profit)) AS TotaLProfit
    FROM SALES
    GROUP BY Category, Sub_Category;
```



11

```sql
-- AdvancedSQLqueries

/* 1. Write a query to find the top 3 customers with the highest total sales.*/

SELECT
    TOP(3) Customer_Name, SUM(Sales) AS TotalSales
FROM SALES
GROUP BY Customer_Name
ORDER BY TotalSales DESC;
```

Results Messages

	Customer_Name	TotalSales
1	Sean Miller	25043.05
2	Tamara Chand	19052.218
3	Raymond Buch	15117.339

12

```sql
/* 2. Write a query to find the total sales and profit for each region,
    including only orders with a specific ship mode*/

SELECT Region, SUM(Sales) AS TotalSales, SUM(Profit) AS TotalProfit FROM SALES
WHERE Ship_Mode = 'Standard Class'
GROUP BY Region;
```

Results | Messages

	Region	TotalSales	TotalProfit
1	East	405321.831	57023.2081
2	south	227613.5535	26952.233
3	West	406752.7985	54760.9657
4	central	318527.56	25352.3807

13

```sql
/* 3. Write a query to find the average sales
     and profit for each category and subcategory*/

SELECT
        Category, Sub_Category,
        AVG(CONVERT(INT,Sales)) AS TotalSales,
        AVG(CONVERT(INT,Profit)) AS TotaLProfit
FROM SALES
GROUP BY Category, Sub_Category
ORDER BY TotaLProfit DESC, TotalSales DESC;
```

Results | Messages

	Category	Sub_Category	TotalSales	TotaLProfit
1	Technology	Copiers	2198	817
2	Technology	Accessories	216	54
3	Technology	Phones	371	50
4	Furniture	Chairs	532	43
5	Office Supplies	Appliances	230	38
6	Technology	Machines	1645	29
7	Office Supplies	Envelopes	64	27
8	Office Supplies	Storage	264	25
9	Office Supplies	Paper	57	24
10	Office Supplies	Binders	133	19
11	Office Supplies	Labels	34	15
12	Furniture	Furnishings	95	13
13	Office Supplies	Art	34	8
14	Office Supplies	Fasteners	13	4
15	Office Supplies	Supplies	245	-6

Query executed successfully.

14

```sql
/* 4. Write a query to find the top 3 products
     with the highest total sales and profit.*/

SELECT
        TOP(3) Product_ID,
        SUM(CONVERT(INT,Sales)) AS TotalSales,
        SUM(CONVERT(INT,Profit)) AS TotaLProfit
FROM SALES
GROUP BY Product_ID
ORDER BY TotaLProfit DESC, TotalSales DESC;
```

Results | Messages

	Product_ID	TotalSales	TotaLProfit
1	TEC-CO-10004722	61600	25200
2	OFF-BI-10003527	27454	7751
3	TEC-CO-10001449	18840	6984

15

```sql
-- Windowfunctions

/* 1. Write a query to find the running total of sales for each order*/

SELECT
    Order_ID,
    TotalSales,
    SUM(TotalSales)
        OVER(ORDER BY TotalSales ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS RunningTotal
FROM
    (SELECT
        Order_ID,
        SUM(CONVERT(INT, Sales)) AS TotalSales
    FROM SALES
    GROUP BY Order_ID) AS SalesSummary
ORDER BY TotalSales, RunningTotal;
```

Results | Messages

	Order_ID	TotalSales	RunningTotal
1	CA-2014-124114	1	1
2	CA-2014-106691	1	2
3	US-2014-100209	1	3
4	US-2014-162068	1	4
5	CA-2011-112718	1	5
6	CA-2012-146829	1	6
7	CA-2013-168361	1	7
8	US-2011-152723	1	8
9	CA-2011-112403	1	9
10	CA-2014-165099	1	10
11	CA-2013-108644	2	12
12	CA-2014-101728	2	14
13	US-2014-155299	2	16
14	CA-2014-145093	2	18
15	CA-2014-160045	2	20
16	CA-2013-108364	2	22

Query executed successfully.

16

```sql
/* 2. Write a query to find the average sales and profit for each region
    including a running total.*/

SELECT
    Region,
    TotalSales,
    TotaLProfit,
    SUM(TotalSales)
        OVER(ORDER BY Region ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS SalesRT,
    SUM(TotaLProfit)
        OVER(ORDER BY Region ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS ProfitRT
FROM
    (SELECT Region,
        SUM(CONVERT(INT, Sales)) AS TotalSales,
        SUM(CONVERT(INT,Profit)) AS TotaLProfit
    FROM SALES
    GROUP BY Region) AS Summary;
```

Results | Messages

	Region	TotalSales	TotaLProfit	SalesRT	ProfitRT
1	central	501256	39719	501256	39719
2	East	678834	91521	1180090	131240
3	South	391750	46721	1571840	177961
4	West	725514	108386	2297354	286347

17

```sql
/* 3. Write a query to find the total sales and profit for each product
    category and sub category including a running total*/

SELECT
    Category,Sub_Category,
    TotalSales,
    TotaLProfit,
    SUM(TotalSales)
        OVER(ORDER BY Category,Sub_Category ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS SalesRT,
    SUM(TotaLProfit)
        OVER(ORDER BY Category,Sub_Category ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS ProfitRT
FROM
    (SELECT Category,Sub_Category,
        SUM(CONVERT(INT, Sales)) AS TotalSales,
        SUM(CONVERT(INT,Profit)) AS TotaLProfit
    FROM SALES
    GROUP BY Category,Sub_Category) AS Summary;
```

Results | Messages

	Category	Sub_Category	TotalSales	TotaLProfit	SalesRT	ProfitRT
1	Furniture	Bookcases	114879	-3479	114879	-3479
2	Furniture	Chairs	328454	26586	443333	23107
3	Furniture	Furnishings	91705	13070	535038	36177
4	Furniture	Tables	206968	-17733	742006	18444
5	Office S...	Appliances	107538	18132	849544	36576
6	Office S...	Art	27137	6530	876681	43106
7	Office S...	Binders	203428	30200	10801...	73306
8	Office S...	Envelopes	16477	6956	10965...	80262
9	Office S...	Fasteners	3024	952	10996...	81214
10	Office S...	Labels	12507	5558	11121...	86772
11	Office S...	Paper	78475	34053	11905...	120825
12	Office S...	Storage	223862	21280	14144...	142105
13	Office S...	Supplies	46679	-1187	14611...	140918
14	Techno...	Accessories	167401	41932	16285...	182850
15	Techno...	Copiers	149530	55618	17780...	238468
16	Techno...	Machines	189243	3387	19673...	241855

Query executed successfully.

18

```sql
--CommonTableExpressions(CTEs)

/* 1. Write a query to find the total sales and profit for each region using
    a CTEto calculate the running total*/

WITH SUMMARY AS
    (SELECT Region,
        SUM(CONVERT(INT, Sales)) AS TotalSales,
        SUM(CONVERT(INT,Profit)) AS TotaLProfit
    FROM SALES
    GROUP BY Region)

SELECT Region,TotalSales,TotaLProfit,
        SUM(TotalSales)
            OVER(ORDER BY Region ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS SalesRT,
        SUM(TotaLProfit)
            OVER(ORDER BY Region ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS ProfitRT
FROM SUMMARY;
```

Results | Messages

	Region	TotalSales	TotaLProfit	SalesRT	ProfitRT
1	central	501256	39719	501256	39719
2	East	678834	91521	1180090	131240
3	South	391750	46721	1571840	177961
4	West	725514	108386	2297354	286347

19

```sql
/* 2. Write a query to find the top 3 customers with the highest total
    sales, using a CTE to calculate the ranking*/

WITH SUMMARY AS
    (SELECT Customer_ID,Customer_Name,
            SUM(Sales) AS TOTALSALES
    FROM SALES
    GROUP BY Customer_ID,Customer_Name )

SELECT TOP(3)*, RANK() OVER(ORDER BY TOTALSALES DESC) AS "RANK"
FROM SUMMARY
ORDER BY TOTALSALES DESC;
```

Results | Messages

	Customer_ID	Customer_Name	TOTALSALES	RANK
1	SM-20320	Sean Miller	25043.05	1
2	TC-20980	Tamara Chand	19052.218	2
3	RB-19360	Raymond Buch	15117.339	3

20

```sql
/* 3. Write a query to find the average sales for each category and
    subcategory using a CTE to calculate the running average*/

WITH SUMMARY AS
    (SELECT Category, Sub_Category, AVG(CONVERT(INT,Sales)) AS AVERAGE
     FROM SALES
     GROUP BY Category, Sub_Category)

SELECT Category, Sub_Category, "AVERAGE" AS "AVG Sales",
       AVG("AVERAGE")
           OVER(ORDER BY Category, Sub_Category ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS RunningAVG
FROM SUMMARY;
```

Results | Messages

	Category	Sub_Category	AVG Sales	RunningAVG
1	Furniture	Bookcases	503	503
2	Furniture	Chairs	532	517
3	Furniture	Furnishings	95	376
4	Furniture	Tables	648	444
5	Office Supplies	Appliances	230	401
6	Office Supplies	Art	34	340
7	Office Supplies	Binders	133	310
8	Office Supplies	Envelopes	64	279
9	Office Supplies	Fasteners	13	250
10	Office Supplies	Labels	34	228
11	Office Supplies	Paper	57	213
12	Office Supplies	Storage	264	217
13	Office Supplies	Supplies	245	219
14	Technology	Accessories	216	219
15	Technology	Copiers	2198	351
16	Technology	Machines	1645	431
17	Technology	Phones	371	428

Query executed successfully.

21

```sql
/* 4. Write a query to find the total sales and profit for each customer
    using a CTE to calculate the running total*/

WITH SUMMARY AS
    (SELECT Customer_ID, Customer_Name, SUM(Sales) AS TotalSales, SUM(Profit) AS TotalProfit
     FROM SALES
     GROUP BY Customer_ID, Customer_Name)

SELECT Customer_ID, Customer_Name, TotalSales,TotalProfit,
        SUM(TotalSales)
            OVER(ORDER BY Customer_ID, Customer_Name ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS SalesRT,
        SUM(TotaLProfit)
            OVER(ORDER BY Customer_ID, Customer_Name ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS ProfitRT
FROM SUMMARY;
```

Results | Messages

	Customer_ID	Customer_Name	TotalSales	TotalProfit	SalesRT	ProfitRT
1	AA-10315	Alex Avila	5563.56	-362.8825	5563.56	-362.8825
2	AA-10375	Allen Armold	1056.39	277.3824	6619.95	-85.5001
3	AA-10480	Andrew Allen	1790.512	435.8274	8410.462	350.3273
4	AA-10645	Anna Andreadi	5086.935	857.8033	13497.397	1208.1306
5	AB-10015	Aaron Bergman	886.156	129.3465	14383.553	1337.4771
6	AB-10060	Adam Bellavance	7755.62	2054.58...	22139.173	3392.0656
7	AB-10105	Adrian Barton	14473.571	5444.80...	36612.744	8836.8711
8	AB-10150	Aimee Bixby	966.71	313.6597	37579.454	9150.5308
9	AB-10165	Alan Barnes	1113.838	220.813	38693.292	9371.3438
10	AB-10255	Alejandro Balle...	914.532	264.5675	39607.824	9635.9113
11	AB-10600	Ann Blume	1515.862	-274.9604	41123.686	9360.9509
12	AC-10420	Alyssa Crouse	925.80	-62.1342	42049.486	9298.8167
13	AC-10450	Amy Cox	5527.846	1366.00...	47577.332	10664.8...

Query executed successfully.

22

```sql
/* 5. Write a query to find the total sales and profit for each region using
    a CTEto calculate the running total and including only orders
    within a specific year and month*/

WITH SUMMARY AS
    (SELECT Region,
        SUM(Sales) AS TotalSales,
        SUM(Profit) AS TotalProfit
     FROM SALES
     WHERE CONVERT(DATE,Order_Date) BETWEEN '1/1/2011' AND '1/31/2011'
     GROUP BY Region)

SELECT Region,TotalSales,TotalProfit,
        SUM(TotalSales)
            OVER(ORDER BY Region ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS SalesRT,
        SUM(TotaLProfit)
            OVER(ORDER BY Region ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS ProfitRT
FROM SUMMARY
ORDER BY Region;
```

Results | Messages

	Region	TotalSales	TotalProfit	SalesRT	ProfitRT
1	Central	1539.906	118.4902	1539.906	118.4902
2	East	436.174	-39.3564	1976.08	79.1338
3	South	9322.092	2346.664	11298.172	2425.7978
4	West	2648.057	20.9733	13946.229	2446.7711

DONE