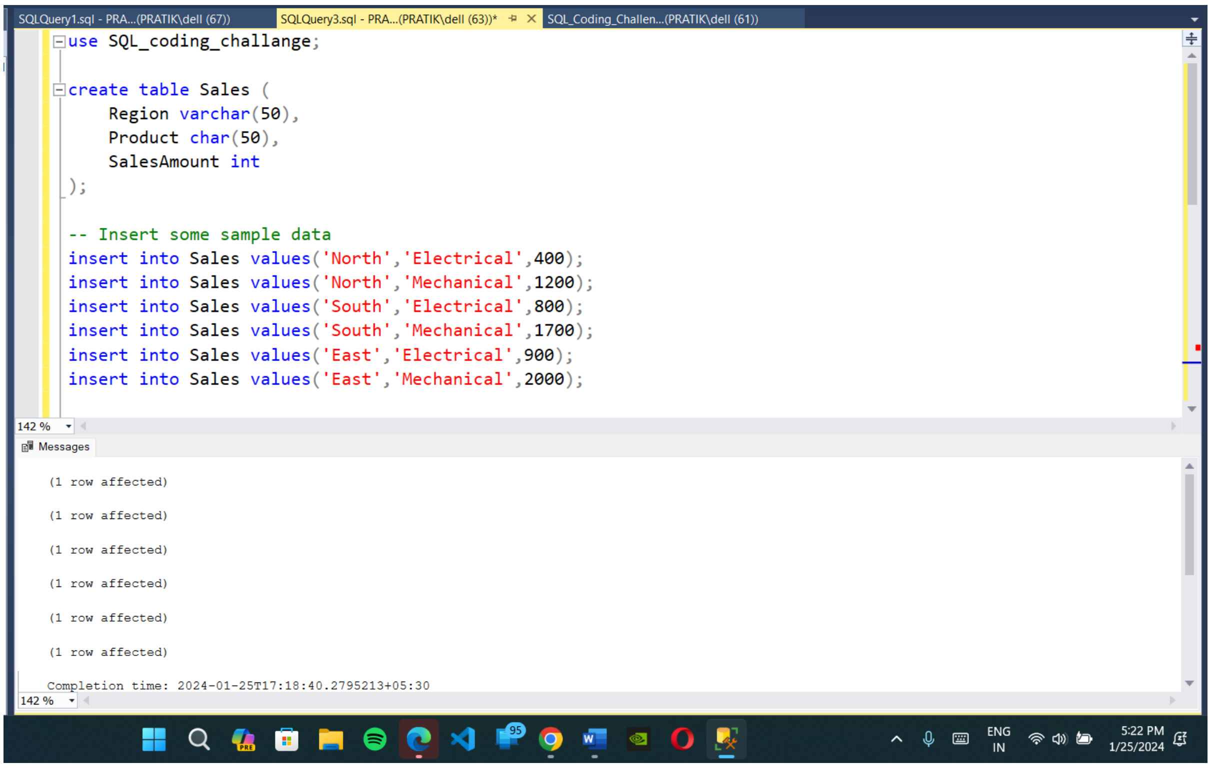
SQL Coding Challenge 1

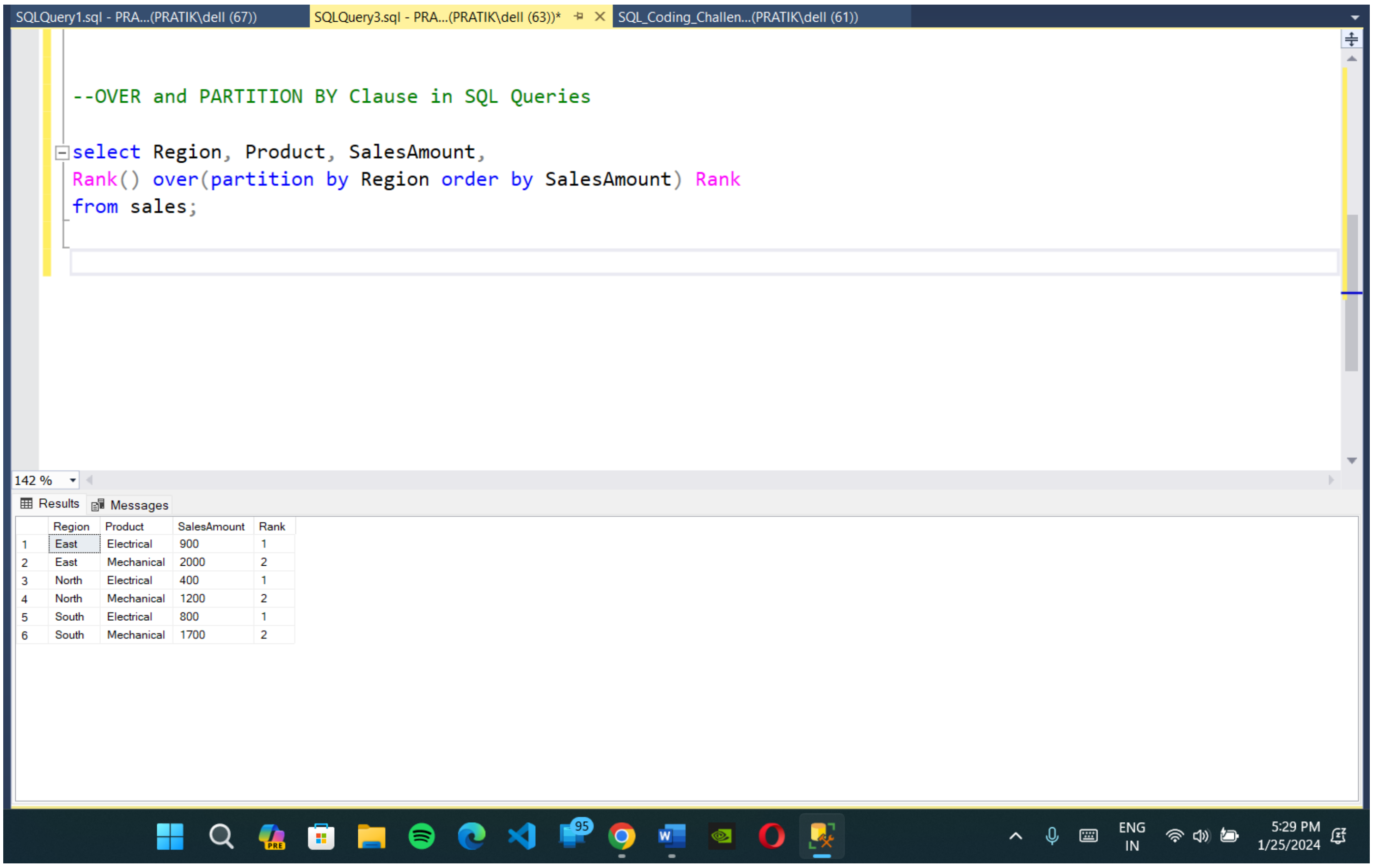
- Database Information:
  - Created Database name as SQL_coding_challange
  - Created one tables: Sales


SQLQuery1.sql - PRA...(PRATIK\dell (67))
SQLQuery3.sql - PRA...(PRATIK\dell (63))*
SQL_Coding_Challen...(PRATIK\dell (61))
142 %
Messages

```sql
use SQL_coding_challange;

create table Sales (
    Region varchar(50),
    Product char(50),
    SalesAmount int
);

-- Insert some sample data
insert into Sales values('North','Electrical',400);
insert into Sales values('North','Mechanical',1200);
insert into Sales values('South','Electrical',800);
insert into Sales values('South','Mechanical',1700);
insert into Sales values('East','Electrical',900);
insert into Sales values('East','Mechanical',2000);
```

```
(1 row affected)

(1 row affected)

(1 row affected)

(1 row affected)

(1 row affected)

(1 row affected)

Completion time: 2024-01-25T17:18:40.2795213+05:30
```

- Over and Partition by:
  - The OVER clause use to specifies the rows over which a window function.
  - It can include both the PARTITION BY clause and the ORDER BY clause.
  - PARTITION BY clause is used to divide the result set into groups based on some column
  - As here we are dividing the result according to there region.
  - Here also use RANK() function to give each row specific rank according to region and sales.
  - Here It is partition by Region and order by sales amount
  - So Rank 1 is for East region and Electrical product.

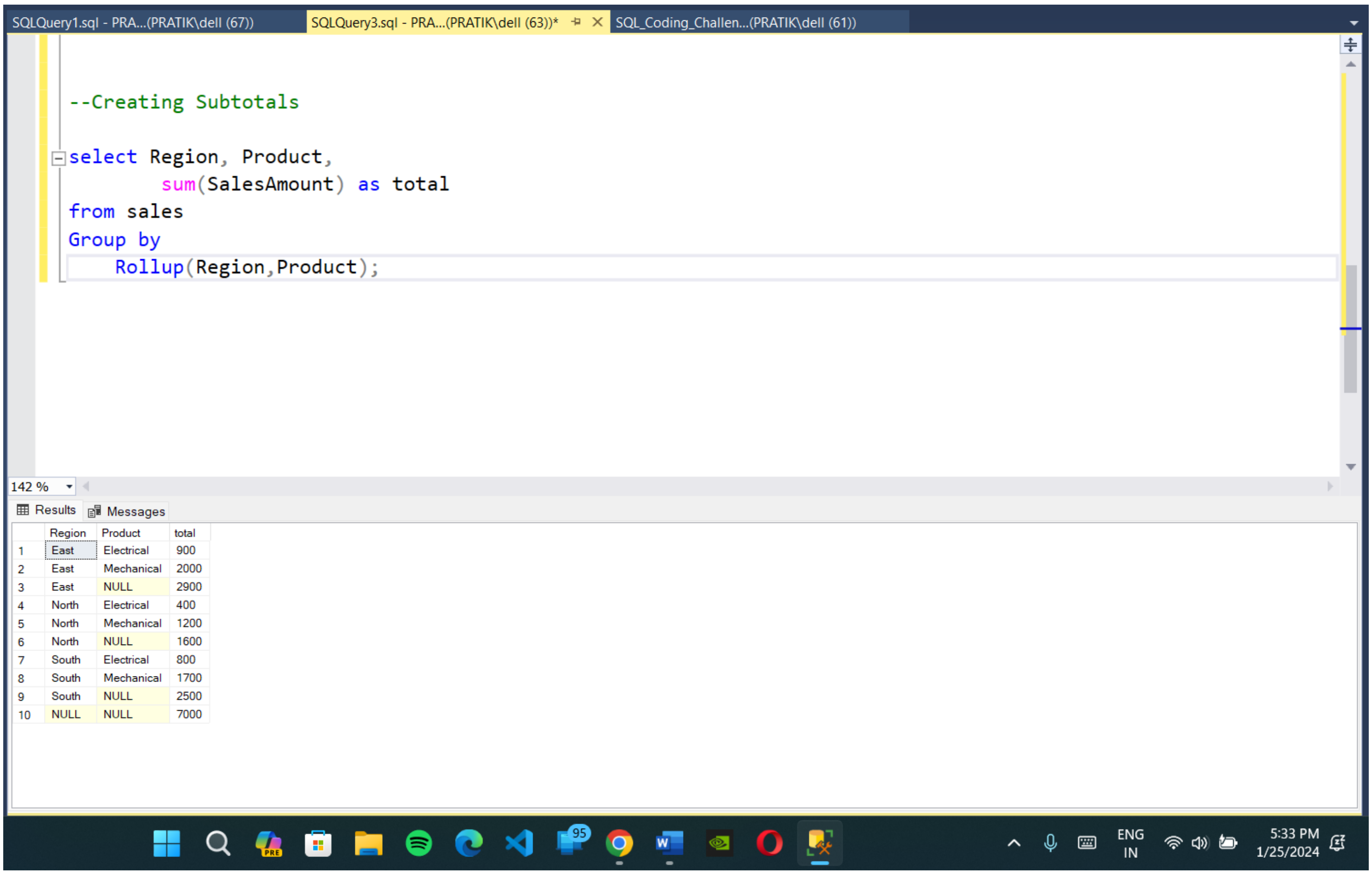


SQLQuery1.sql - PRA...(PRATIK\dell (67))
SQLQuery3.sql - PRA...(PRATIK\dell (63))*
SQL_Coding_Challen...(PRATIK\dell (61))
142 %
Results
Messages

```
--OVER and PARTITION BY Clause in SQL Queries

select Region, Product, SalesAmount,
Rank() over(partition by Region order by SalesAmount) Rank
from sales;
```

	Region	Product	SalesAmount	Rank
1	East	Electrical	900	1
2	East	Mechanical	2000	2
3	North	Electrical	400	1
4	North	Mechanical	1200	2
5	South	Electrical	800	1
6	South	Mechanical	1700	2

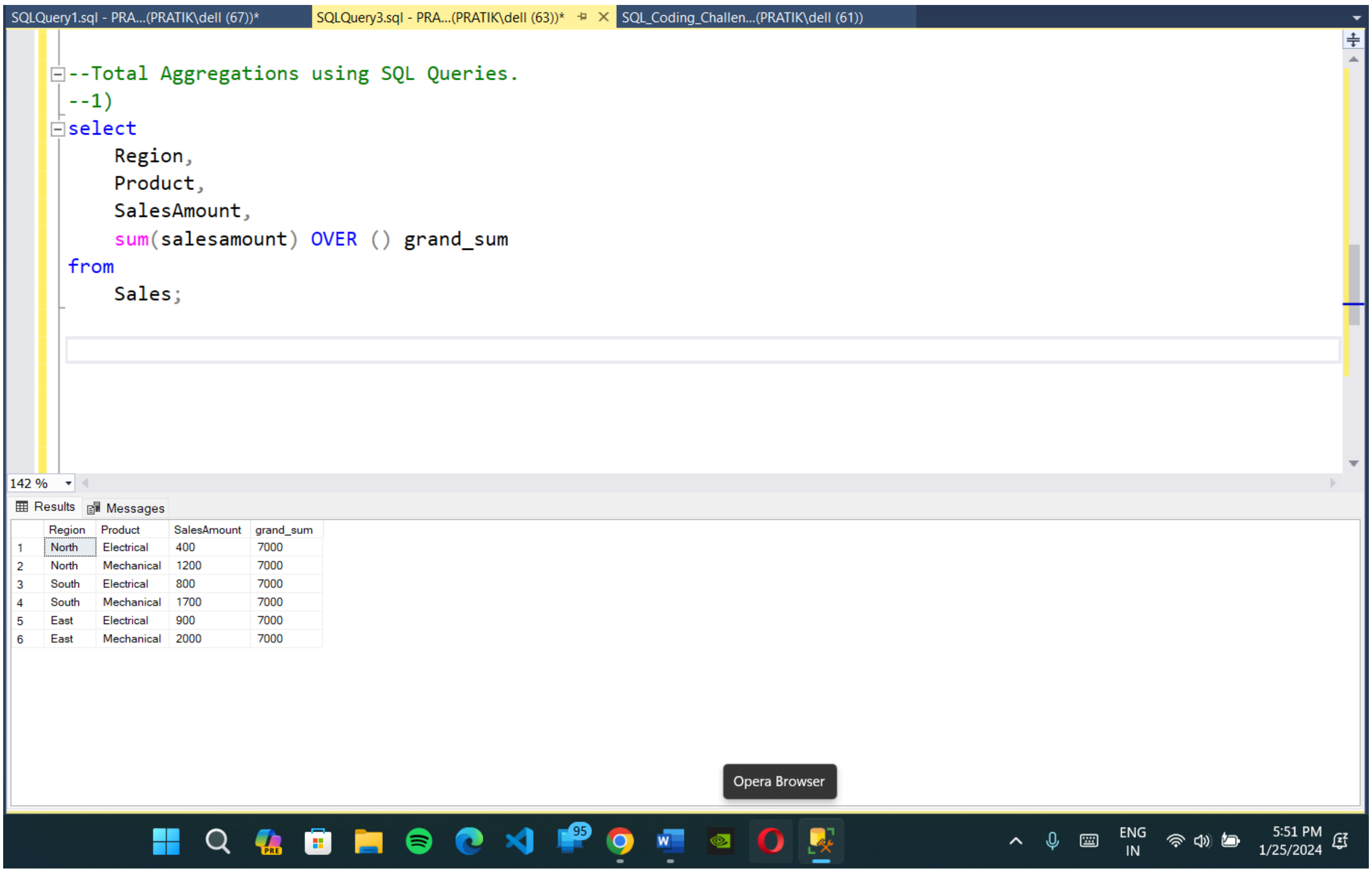
- Creating Subtotals:
    - To create the subtotals we need Group by Rollup.
    - Subtotals is sum for Specific sections
    - Here in these query Subtotal is created for Region wise and Product wise
    - For East Region subtotal = <u>2900</u>
    - For North Region subtotal = <u>1600</u>
    - For South Region subtotal = <u>2500</u>
    - At the end we will also got the grand total as <u>7000.</u>


SQLQuery1.sql - PRA...(PRATIK\dell (67))
SQLQuery3.sql - PRA...(PRATIK\dell (63))*
SQL_Coding_Challen...(PRATIK\dell (61))
142 %
Results
Messages

```
--Creating Subtotals

select Region, Product,
        sum(SalesAmount) as total
from sales
Group by
    Rollup(Region,Product);
```

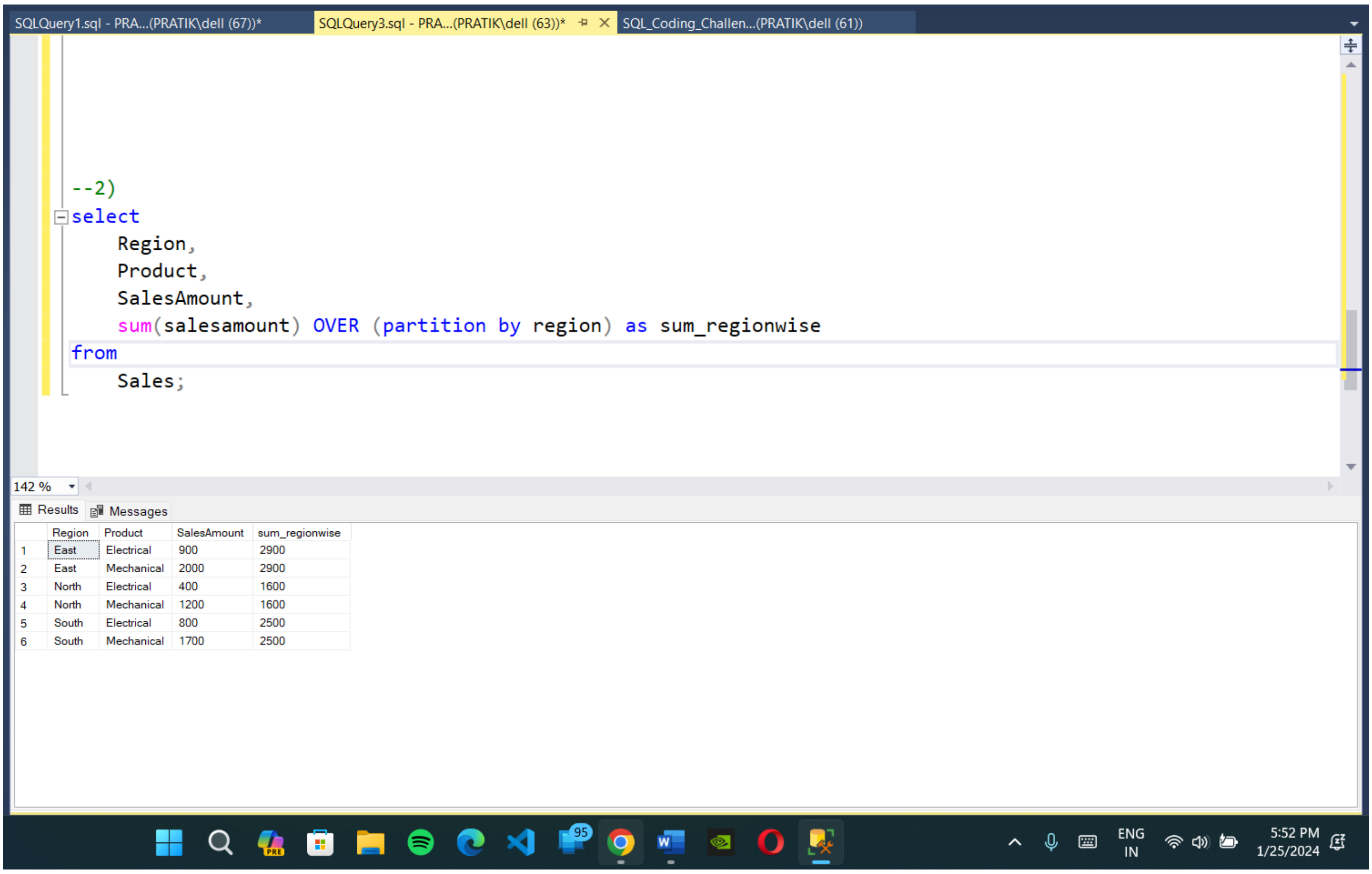
	Region	Product	total
1	East	Electrical	900
2	East	Mechanical	2000
3	East	NULL	2900
4	North	Electrical	400
5	North	Mechanical	1200
6	North	NULL	1600
7	South	Electrical	800
8	South	Mechanical	1700
9	South	NULL	2500
10	NULL	NULL	7000

- Total Aggregations using SQL Queries:
  - When we use aggregations on the over() and order by clause we called it as total aggregation.
  - We can also use aggregations by partitioning the result set
  - If we didn’t use partition by then aggregation is applied on whole set.
  - Here In 1st example sum() is applied on whole set and we got the aggregation for whole result set which comes <u>7000</u>
  - In 2nd example sum() is applied by making partition according to the region so got sum for specific regions
  - For East Region subtotal = <u>2900</u>
  - For North Region subtotal = <u>1600</u>
  - For South Region subtotal = <u>2500</u>


SQLQuery1.sql - PRA...(PRATIK\dell (67))*
SQLQuery3.sql - PRA...(PRATIK\dell (63))*
SQL_Coding_Challen...(PRATIK\dell (61))
142 %
Results
Messages
Opera Browser
ENG IN
5:51 PM
1/25/2024

```sql
--Total Aggregations using SQL Queries.
--1)
select
    Region,
    Product,
    SalesAmount,
    sum(salesamount) OVER () grand_sum
from
    Sales;
```

```
   Region  Product     SalesAmount  grand_sum
1  North   Electrical  400          7000
2  North   Mechanical  1200         7000
3  South   Electrical  800          7000
4  South   Mechanical  1700         7000
5  East    Electrical  900          7000
6  East    Mechanical  2000         7000
```


SQLQuery1.sql - PRA...(PRATIK\dell (67))*
SQLQuery3.sql - PRA...(PRATIK\dell (63))*
SQL_Coding_Challen...(PRATIK\dell (61))
142 %
Results
Messages
ENG IN
5:52 PM
1/25/2024

```sql
--2)
select
    Region,
    Product,
    SalesAmount,
    sum(salesamount) OVER (partition by region) as sum_regionwise
from
    Sales;
```

```
   Region  Product     SalesAmount  sum_regionwise
1  East    Electrical  900          2900
2  East    Mechanical  2000         2900
3  North   Electrical  400          1600
4  North   Mechanical  1200         1600
5  South   Electrical  800          2500
6  South   Mechanical  1700         2500
```