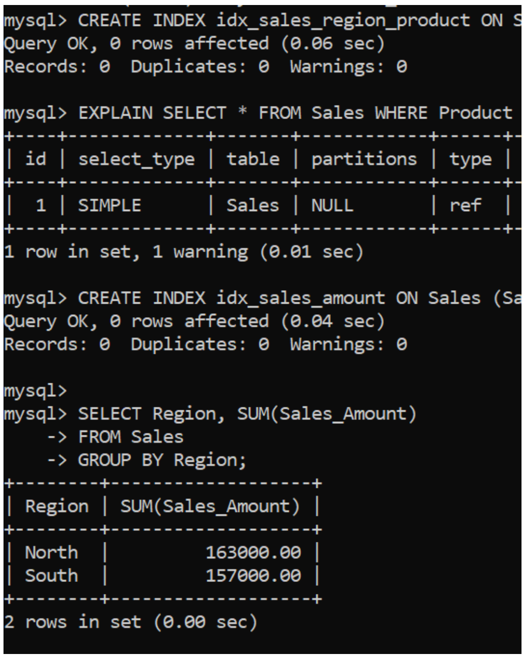
Experiment 7 Output

SQL queries were **optimized** for large-scale data warehouse applications using **indexing, partitioning, and query tuning techniques**.

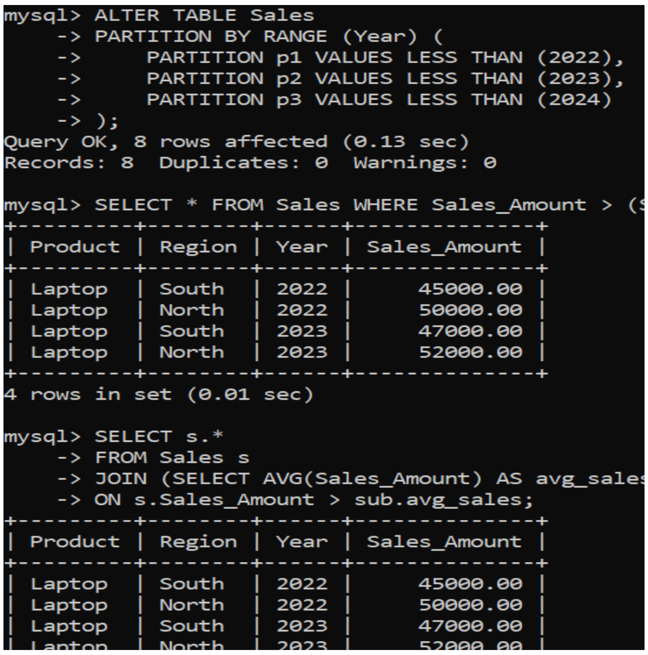
Queries and indexing



```
mysql> CREATE INDEX idx_sales_region_product ON S
Query OK, 0 rows affected (0.06 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> EXPLAIN SELECT * FROM Sales WHERE Product
+----+------------+-------+------------+------+-
| id | select_type | table | partitions | type |
+----+------------+-------+------------+------+-
|  1 | SIMPLE      | Sales | NULL       | ref  |
+----+------------+-------+------------+------+-
1 row in set, 1 warning (0.01 sec)

mysql> CREATE INDEX idx_sales_amount ON Sales (Sa
Query OK, 0 rows affected (0.04 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql>
mysql> SELECT Region, SUM(Sales_Amount)
    -> FROM Sales
    -> GROUP BY Region;
+--------+------------------+
| Region | SUM(Sales_Amount) |
+--------+------------------+
| North  |        163000.00 |
| South  |        157000.00 |
+--------+------------------+
2 rows in set (0.00 sec)
```

Fig 1: Queries output after indexing

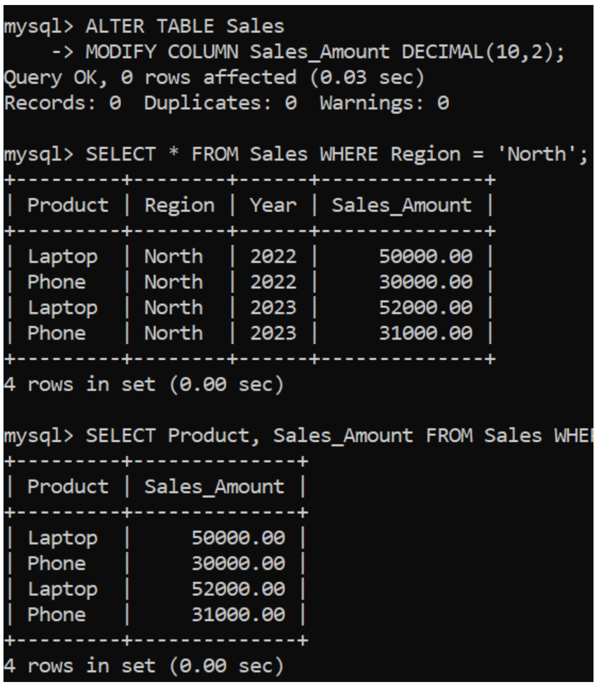
Selecting Tuples from tables



```
mysql> ALTER TABLE Sales
    -> PARTITION BY RANGE (Year) (
    ->     PARTITION p1 VALUES LESS THAN (2022),
    ->     PARTITION p2 VALUES LESS THAN (2023),
    ->     PARTITION p3 VALUES LESS THAN (2024)
    -> );
Query OK, 8 rows affected (0.13 sec)
Records: 8  Duplicates: 0  Warnings: 0

mysql> SELECT * FROM Sales WHERE Sales_Amount > (S
+---------+--------+------+--------------+
| Product | Region | Year | Sales_Amount |
+---------+--------+------+--------------+
| Laptop  | South  | 2022 |     45000.00 |
| Laptop  | North  | 2022 |     50000.00 |
| Laptop  | South  | 2023 |     47000.00 |
| Laptop  | North  | 2023 |     52000.00 |
+---------+--------+------+--------------+
4 rows in set (0.01 sec)

mysql> SELECT s.*
    -> FROM Sales s
    -> JOIN (SELECT AVG(Sales_Amount) AS avg_sales
    -> ON s.Sales_Amount > sub.avg_sales;
+---------+--------+------+--------------+
| Product | Region | Year | Sales_Amount |
+---------+--------+------+--------------+
| Laptop  | South  | 2022 |     45000.00 |
| Laptop  | North  | 2022 |     50000.00 |
| Laptop  | South  | 2023 |     47000.00 |
| Laptop  | North  | 2023 |     52000.00 |
```

Fig 2: Selecting using conditions instead of *

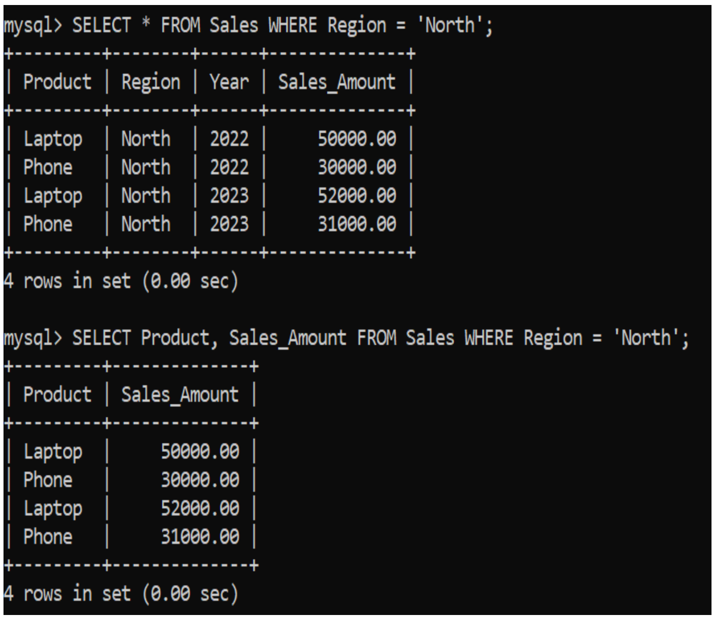
Extracting Data



```
mysql> ALTER TABLE Sales
    -> MODIFY COLUMN Sales_Amount DECIMAL(10,2);
Query OK, 0 rows affected (0.03 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> SELECT * FROM Sales WHERE Region = 'North';
+---------+--------+------+--------------+
| Product | Region | Year | Sales_Amount |
+---------+--------+------+--------------+
| Laptop  | North  | 2022 |     50000.00 |
| Phone   | North  | 2022 |     30000.00 |
| Laptop  | North  | 2023 |     52000.00 |
| Phone   | North  | 2023 |     31000.00 |
+---------+--------+------+--------------+
4 rows in set (0.00 sec)

mysql> SELECT Product, Sales_Amount FROM Sales WHE
+---------+--------------+
| Product | Sales_Amount |
+---------+--------------+
| Laptop  |     50000.00 |
| Phone   |     30000.00 |
| Laptop  |     52000.00 |
| Phone   |     31000.00 |
+---------+--------------+
4 rows in set (0.00 sec)
```

Fig 3: Conditional Queries Output

Loading Data using Columns



```
mysql> SELECT * FROM Sales WHERE Region = 'North';
+---------+--------+------+--------------+
| Product | Region | Year | Sales_Amount |
+---------+--------+------+--------------+
| Laptop  | North  | 2022 |     50000.00 |
| Phone   | North  | 2022 |     30000.00 |
| Laptop  | North  | 2023 |     52000.00 |
| Phone   | North  | 2023 |     31000.00 |
+---------+--------+------+--------------+
4 rows in set (0.00 sec)

mysql> SELECT Product, Sales_Amount FROM Sales WHERE Region = 'North';
+---------+--------------+
| Product | Sales_Amount |
+---------+--------------+
| Laptop  |     50000.00 |
| Phone   |     30000.00 |
| Laptop  |     52000.00 |
| Phone   |     31000.00 |
+---------+--------------+
4 rows in set (0.00 sec)
```

Fig 4: Columns Queries

Madhurima Rawat DS 42