ATLIQ HARDWARE

FINANCE AND SUPPLY CHAIN ANALYTICS

PROBLEM STATEMENT

Atliq Hardware, a leading provider of computer hardware such as PCs, printers, and mice, faced challenges in analyzing its growing sales and market data. Relying on large Excel files for data processing caused performance bottlenecks, limiting the ability to derive actionable insights and make timely decisions. The objective of this project was to utilize MySQL for transforming raw sales data into meaningful insights, enabling a deeper understanding of sales performance, market dynamics, customer behavior, and forecasting supply chain trends.

FINANCE ANALYTICS

Queries

1. Customer-Specific Sales Analysis

a. **Retrieve customer details:**

```
SELECT *
FROM dim_customer
WHERE customer LIKE "%croma%" AND market = "india";
```

b. **Extract all sales transactions for a specific customer for the fiscal year 2021:**

```
SELECT *
FROM fact_sales_monthly
WHERE customer_code = 90002002
  AND YEAR(DATE_ADD(date, INTERVAL 4 MONTH)) = 2021
ORDER BY date ASC
LIMIT 100000;
```

date	fiscal_year	product_code	customer_code	sold_quantity
2020-09-01	2021	A0118150101	90002002	202
2020-09-01	2021	A0118150102	90002002	162
2020-09-01	2021	A0118150103	90002002	193
2020-09-01	2021	A0118150104	90002002	146
2020-09-01	2021	A0219150201	90002002	149
2020-09-01	2021	A0219150202	90002002	107
2020-09-01	2021	A0220150203	90002002	123
2020-09-01	2021	A0320150301	90002002	146
2020-09-01	2021	A0321150302	90002002	236
2020-09-01	2021	A0321150303	90002002	137
2020-09-01	2021	A0418150103	90002002	23
2020-09-01	2021	A0418150104	90002002	82
2020-09-01	2021	A0418150105	90002002	86
2020-09-01	2021	A0418150106	90002002	48
2020-09-01	2021	A0519150201	90002002	138
2020-09-01	2021	A0519150202	90002002	72
2020-09-01	2021	A0519150203	90002002	38

c. **Create a function to calculate the fiscal year:**

```
CREATE FUNCTION `get_fiscal_year`(calendar_date DATE)
RETURNS INT
DETERMINISTIC
BEGIN
    DECLARE fiscal_year INT;
    SET fiscal_year = YEAR(DATE_ADD(calendar_date, INTERVAL 4 MONTH));
    RETURN fiscal_year;
 END;
```

d. **Use the custom fiscal year function for querying sales transactions:**

```
SELECT *
FROM fact_sales_monthly
WHERE customer_code = 90002002
  AND get_fiscal_year(date) = 2021
```

```
ORDER BY date ASC
LIMIT 100000;
```

2. Gross Sales Reporting

a. **Perform a join to fetch product information:**

```
SELECT s.date, s.product_code, p.product, p.variant, s.sold_quantity
FROM fact_sales_monthly s
JOIN dim_product p ON s.product_code = p.product_code
WHERE customer_code = 90002002
  AND get_fiscal_year(date) = 2021
LIMIT 1000000;
```

b. **Calculate gross sales by joining with the fact_gross_price table:**

```
    SELECT
        s.date,
        s.product_code,
        p.product,
        p.variant,
        s.sold_quantity,
        g.gross_price,
        ROUND(s.sold_quantity * g.gross_price, 2) AS gross_price_total
FROM fact_sales_monthly s
JOIN dim_product p ON s.product_code = p.product_code
JOIN fact_gross_price g ON g.fiscal_year = get_fiscal_year(s.date)
  AND g.product_code = s.product_code
WHERE customer_code = 90002002
  AND get_fiscal_year(s.date) = 2021
LIMIT 1000000;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

date	product_code	product	variant	sold_quantity	gross_price	gross_price_total
2020-09-01	A0118150101	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 RP...	Standard	202	19.0573	3849.57
2020-09-01	A0118150102	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 RP...	Plus	162	21.4565	3475.95
2020-09-01	A0118150103	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 RP...	Premium	193	21.7795	4203.44
2020-09-01	A0118150104	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 RP...	Premium Plus	146	22.9729	3354.04
2020-09-01	A0219150201	AQ WereWolf NAS Internal Hard Drive HDD – 8.8...	Standard	149	23.6987	3531.11
2020-09-01	A0219150202	AQ WereWolf NAS Internal Hard Drive HDD – 8.8...	Plus	107	24.7312	2646.24
2020-09-01	A0220150203	AQ WereWolf NAS Internal Hard Drive HDD – 8.8...	Premium	123	23.6154	2904.69
2020-09-01	A0320150301	AQ Zion Saga	Standard	146	23.7223	3463.46
2020-09-01	A0321150302	AQ Zion Saga	Plus	236	27.1027	6396.24
2020-09-01	A0321150303	AQ Zion Saga	Premium	137	28.0059	3836.81
2020-09-01	A0418150103	AQ Mforce Gen X	Standard 3	23	19.5235	449.04
2020-09-01	A0418150104	AQ Mforce Gen X	Plus 1	82	19.9239	1633.76
2020-09-01	A0418150105	AQ Mforce Gen X	Plus 2	86	20.0766	1726.59
2020-09-01	A0418150106	AQ Mforce Gen X	Plus 3	48	19.9365	956.95
2020-09-01	A0519150201	AQ Mforce Gen Y	Standard 1	138	22.3984	3090.98
2020-09-01	A0519150202	AQ Mforce Gen Y	Standard 2	72	24.9298	1794.95
2020-09-01	A0519150203	AQ Mforce Gen Y	Standard 3	38	26.5871	1010.31

c. Generate a monthly gross sales report for a specific customer:

```
SELECT
    s.date,
    SUM(ROUND(s.sold_quantity * g.gross_price, 2)) AS monthly_sales
FROM fact_sales_monthly s
JOIN fact_gross_price g ON g.fiscal_year = get_fiscal_year(s.date)
  AND g.product_code = s.product_code
WHERE customer_code = 90002002
GROUP BY date;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

date	monthly_sales
2017-09-01	122407.57
2017-10-01	162687.56
2017-12-01	245673.84
2018-01-01	127574.73
2018-02-01	144799.54
2018-04-01	130643.92
2018-05-01	139165.06
2018-06-01	125735.36
2018-08-01	125409.90
2018-09-01	343337.14
2018-10-01	440562.10
2018-12-01	653944.72
2019-01-01	359025.06
2019-02-01	356607.19
2019-04-01	379549.74
2019-05-01	340152.29
2019-06-01	343792.08

3. Stored Procedures

a. Create a stored procedure to generate a gross sales report for any customer:

```
CREATE PROCEDURE `get_monthly_gross_sales_for_customer`(
    IN in_customer_codes TEXT
)
BEGIN
    SELECT
        s.date,
        SUM(ROUND(s.sold_quantity * g.gross_price, 2)) AS monthly_sales
    FROM fact_sales_monthly s
    JOIN fact_gross_price g ON g.fiscal_year = get_fiscal_year(s.date)
    AND g.product_code = s.product_code
    WHERE FIND_IN_SET(s.customer_code, in_customer_codes) > 0
    GROUP BY s.date
    ORDER BY s.date DESC;
END;
```

b. Market classification based on total sales:

```
    CREATE PROCEDURE `get_market_badge`(
        IN in_market VARCHAR(45),
        IN in_fiscal_year YEAR,
        OUT out_level VARCHAR(45)
)
BEGIN
  DECLARE qty INT DEFAULT 0;

        IF in_market = "" THEN
          SET in_market = "India";
        END IF;

  SELECT
    SUM(s.sold_quantity) INTO qty
  FROM fact_sales_monthly s
  JOIN dim_customer c ON s.customer_code = c.customer_code
  WHERE get_fiscal_year(s.date) = in_fiscal_year
   AND c.market = in_market;

  IF qty > 5000000 THEN
    SET out_level = 'Gold';
  ELSE
    SET out_level = 'Silver';
  END IF;
END;
```

Identifying Top Products, Customers, and Markets

- Identifying top customers, products, and markets.
- Generating pre- and post-invoice discount reports.
- Performance optimization of SQL queries.
- Leveraging window functions and stored procedures for advanced analytics.

Pre-Invoice Discount Report

Pre-invoice deductions for a specific customer

```
SELECT
        s.date,
        s.product_code,
        p.product,
        p.variant,
        s.sold_quantity,
        g.gross_price AS gross_price_per_item,
        ROUND(s.sold_quantity * g.gross_price, 2) AS gross_price_total,
        pre.pre_invoice_discount_pct
FROM fact_sales_monthly s
JOIN dim_product p ON s.product_code = p.product_code
JOIN fact_gross_price g
  ON g.fiscal_year = get_fiscal_year(s.date)
  AND g.product_code = s.product_code
JOIN fact_pre_invoice_deductions AS pre
  ON pre.customer_code = s.customer_code
  AND pre.fiscal_year = get_fiscal_year(s.date)
WHERE
  s.customer_code = 90002002
  AND get_fiscal_year(s.date) = 2021
LIMIT 1000000;
```

Pre-invoice deductions for all customers

```
SELECT
        s.date,
        s.product_code,
        p.product,
        p.variant,
        s.sold_quantity,
        g.gross_price AS gross_price_per_item,
        ROUND(s.sold_quantity * g.gross_price, 2) AS gross_price_total,
        pre.pre_invoice_discount_pct
FROM fact_sales_monthly s
JOIN dim_product p ON s.product_code = p.product_code
```

```
JOIN fact_gross_price g
  ON g.fiscal_year = get_fiscal_year(s.date)
  AND g.product_code = s.product_code
JOIN fact_pre_invoice_deductions AS pre
  ON pre.customer_code = s.customer_code
  AND pre.fiscal_year = get_fiscal_year(s.date)
WHERE
  get_fiscal_year(s.date) = 2021
LIMIT 1000000;
```

Performance Improvement #1:

1. **Creating dim_date and Joining it with the Table to Avoid Using the get_fiscal_year() Function:**

```
SELECT
        s.date,
        s.customer_code,
        s.product_code,
        p.product,
        p.variant,
        s.sold_quantity,
        g.gross_price AS gross_price_per_item,
        ROUND(s.sold_quantity * g.gross_price, 2) AS gross_price_total,
        pre.pre_invoice_discount_pct
FROM fact_sales_monthly s
JOIN dim_date dt
  ON dt.calendar_date = s.date
JOIN dim_product p
  ON s.product_code = p.product_code
JOIN fact_gross_price g
  ON g.fiscal_year = dt.fiscal_year
  AND g.product_code = s.product_code
JOIN fact_pre_invoice_deductions AS pre
  ON pre.customer_code = s.customer_code
  AND pre.fiscal_year = dt.fiscal_year
WHERE
  dt.fiscal_year = 2021
```

```
LIMIT 1500000;
```

Performance Improvement #2:

1. **Adding the Fiscal Year in the fact_sales_monthly Table:**

```
SELECT
        s.date,
        s.customer_code,
        s.product_code,
        p.product,
        p.variant,
        s.sold_quantity,
        g.gross_price AS gross_price_per_item,
        ROUND(s.sold_quantity * g.gross_price, 2) AS gross_price_total,
        pre.pre_invoice_discount_pct
FROM fact_sales_monthly s
JOIN dim_product p
  ON s.product_code = p.product_code
JOIN fact_gross_price g
  ON g.fiscal_year = s.fiscal_year
  AND g.product_code = s.product_code
JOIN fact_pre_invoice_deductions AS pre
  ON pre.customer_code = s.customer_code
  AND pre.fiscal_year = s.fiscal_year
WHERE
  s.fiscal_year = 2021
LIMIT 1500000;
```

Get Net Invoice Sales :

```
WITH cte1 AS (
  SELECT
        s.date,
        s.customer_code,
        s.product_code,
        p.product,
        p.variant,
```

```
        s.sold_quantity,
        g.gross_price AS gross_price_per_item,
        ROUND(s.sold_quantity * g.gross_price, 2) AS gross_price_total,
        pre.pre_invoice_discount_pct
    FROM fact_sales_monthly s
    JOIN dim_product p
      ON s.product_code = p.product_code
    JOIN fact_gross_price g
      ON g.fiscal_year = s.fiscal_year
      AND g.product_code = s.product_code
    JOIN fact_pre_invoice_deductions AS pre
      ON pre.customer_code = s.customer_code
      AND pre.fiscal_year = s.fiscal_year
    WHERE
      s.fiscal_year = 2021
)
SELECT
  *,
  (gross_price_total - pre_invoice_discount_pct * gross_price_total) AS net_invoice_sales
FROM cte1
LIMIT 1500000;
```

Creating the View sales_preinv_discount:

```
CREATE VIEW sales_preinv_discount AS
    SELECT
        s.date,
        s.fiscal_year,
        s.customer_code,
        c.market,
        s.product_code,
        p.product,
        p.variant,
        s.sold_quantity,
        g.gross_price AS gross_price_per_item,
```

```
        ROUND(s.sold_quantity * g.gross_price, 2) AS gross_price_total,
        pre.pre_invoice_discount_pct
FROM fact_sales_monthly s
JOIN dim_customer c
  ON s.customer_code = c.customer_code
JOIN dim_product p
  ON s.product_code = p.product_code
JOIN fact_gross_price g
  ON g.fiscal_year = s.fiscal_year
  AND g.product_code = s.product_code
JOIN fact_pre_invoice_deductions AS pre
  ON pre.customer_code = s.customer_code
  AND pre.fiscal_year = s.fiscal_year;
```

Generate Net Invoice Sales Using the View sales_preinv_discount:

```
SELECT
  *,
  (gross_price_total - pre_invoice_discount_pct * gross_price_total) AS net_invoice_sales
FROM gdb0041.sales_preinv_discount;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

fiscal_year	customer_code	market	product_code	product	variant	sold_quantity	gross_price_per_item	gross_price_total	pre_invoice_discount_pct	net_invoice_sales
09-01	2018	70002017	India	A0118150101	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Standard	51	15.3952	785.16	0.0824	720.462816
09-01	2018	70002018	India	A0118150101	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Standard	77	15.3952	1185.43	0.2956	835.016892
09-01	2018	70003181	Indonesia	A0118150101	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Standard	17	15.3952	261.72	0.0536	247.691808
09-01	2018	70003182	Indonesia	A0118150101	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Standard	6	15.3952	92.37	0.2378	70.404414
09-01	2018	70006157	Philiphines	A0118150101	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Standard	5	15.3952	76.98	0.1057	68.843214
09-01	2018	70006158	Philiphines	A0118150101	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Standard	7	15.3952	107.77	0.1875	87.563125
09-01	2018	70007198	South Korea	A0118150101	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Standard	29	15.3952	446.46	0.0700	415.207800
09-01	2018	70007199	South Korea	A0118150101	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Standard	34	15.3952	523.44	0.2551	389.910456
09-01	2018	70008169	Australia	A0118150101	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Standard	22	15.3952	338.69	0.0953	306.412843
09-01	2018	70008170	Australia	A0118150101	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Standard	5	15.3952	76.98	0.1896	62.384592
09-01	2018	70011193	France	A0118150101	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Standard	10	15.3952	153.95	0.0521	145.929205
09-01	2018	70011194	France	A0118150101	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Standard	4	15.3952	61.58	0.2046	48.980732
09-01	2018	70012042	Germany	A0118150101	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Standard	0	15.3952	0.00	0.0984	0.000000
09-01	2018	70012043	Germany	A0118150101	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Standard	0	15.3952	0.00	0.2620	0.000000
09-01	2018	70013125	Italy	A0118150101	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Standard	1	15.3952	15.40	0.0587	14.496020

Post Invoice Discount and Net Sales Views:

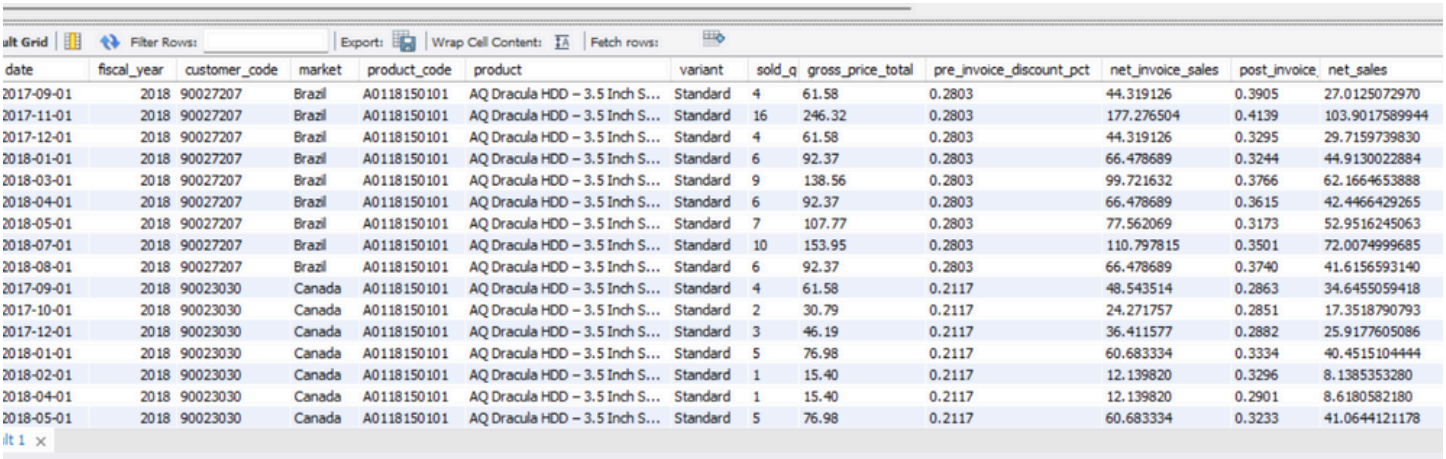
View for Post Invoice Deductions sales_postinv_discount:

```
CREATE VIEW sales_postinv_discount AS
  SELECT
    s.date,
```

```
        s.fiscal_year,
        s.customer_code,
        s.market,
        s.product_code,
        s.product,
        s.variant,
        s.sold_quantity,
        s.gross_price_total,
        s.pre_invoice_discount_pct,
        (s.gross_price_total - s.pre_invoice_discount_pct * s.gross_price_total) AS net_invoice_sales,
        (po.discounts_pct + po.other_deductions_pct) AS post_invoice_discount_pct
    FROM sales_preinv_discount s
    JOIN fact_post_invoice_deductions po
      ON po.customer_code = s.customer_code
      AND po.product_code = s.product_code
      AND po.date = s.date;
```

Create a Report for Net Sales:

```
SELECT
  *,
  net_invoice_sales * (1 - post_invoice_discount_pct) AS net_sales
FROM gdb0041.sales_postinv_discount;
```



date	fiscal_year	customer_code	market	product_code	product	variant	sold_q	gross_price_total	pre_invoice_discount_pct	net_invoice_sales	post_invoice_	net_sales
2017-09-01	2018	90027207	Brazil	A0118150101	AQ Dracula HDD – 3.5 Inch S...	Standard	4	61.58	0.2803	44.319126	0.3905	27.0125072970
2017-11-01	2018	90027207	Brazil	A0118150101	AQ Dracula HDD – 3.5 Inch S...	Standard	16	246.32	0.2803	177.276504	0.4139	103.9017589944
2017-12-01	2018	90027207	Brazil	A0118150101	AQ Dracula HDD – 3.5 Inch S...	Standard	4	61.58	0.2803	44.319126	0.3295	29.7159739830
2018-01-01	2018	90027207	Brazil	A0118150101	AQ Dracula HDD – 3.5 Inch S...	Standard	6	92.37	0.2803	66.478689	0.3244	44.9130022884
2018-03-01	2018	90027207	Brazil	A0118150101	AQ Dracula HDD – 3.5 Inch S...	Standard	9	138.56	0.2803	99.721632	0.3766	62.1664653888
2018-04-01	2018	90027207	Brazil	A0118150101	AQ Dracula HDD – 3.5 Inch S...	Standard	6	92.37	0.2803	66.478689	0.3615	42.4466429265
2018-05-01	2018	90027207	Brazil	A0118150101	AQ Dracula HDD – 3.5 Inch S...	Standard	7	107.77	0.2803	77.562069	0.3173	52.9516245063
2018-07-01	2018	90027207	Brazil	A0118150101	AQ Dracula HDD – 3.5 Inch S...	Standard	10	153.95	0.2803	110.797815	0.3501	72.0074999685
2018-08-01	2018	90027207	Brazil	A0118150101	AQ Dracula HDD – 3.5 Inch S...	Standard	6	92.37	0.2803	66.478689	0.3740	41.6156593140
2017-09-01	2018	90023030	Canada	A0118150101	AQ Dracula HDD – 3.5 Inch S...	Standard	4	61.58	0.2117	48.543514	0.2863	34.6455059418
2017-10-01	2018	90023030	Canada	A0118150101	AQ Dracula HDD – 3.5 Inch S...	Standard	2	30.79	0.2117	24.271757	0.2851	17.3518790793
2017-12-01	2018	90023030	Canada	A0118150101	AQ Dracula HDD – 3.5 Inch S...	Standard	3	46.19	0.2117	36.411577	0.2882	25.9177605086
2018-01-01	2018	90023030	Canada	A0118150101	AQ Dracula HDD – 3.5 Inch S...	Standard	5	76.98	0.2117	60.683334	0.3334	40.4515104444
2018-02-01	2018	90023030	Canada	A0118150101	AQ Dracula HDD – 3.5 Inch S...	Standard	1	15.40	0.2117	12.139820	0.3296	8.1385353280
2018-04-01	2018	90023030	Canada	A0118150101	AQ Dracula HDD – 3.5 Inch S...	Standard	1	15.40	0.2117	12.139820	0.2901	8.6180582180
2018-05-01	2018	90023030	Canada	A0118150101	AQ Dracula HDD – 3.5 Inch S...	Standard	5	76.98	0.2117	60.683334	0.3233	41.0644121178

Create Final View for Net Sales:

```
CREATE VIEW net_sales AS
    SELECT
```

```
        *,
        net_invoice_sales * (1 - post_invoice_discount_pct) AS net_sales
    FROM gdb0041.sales_postinv_discount;
```

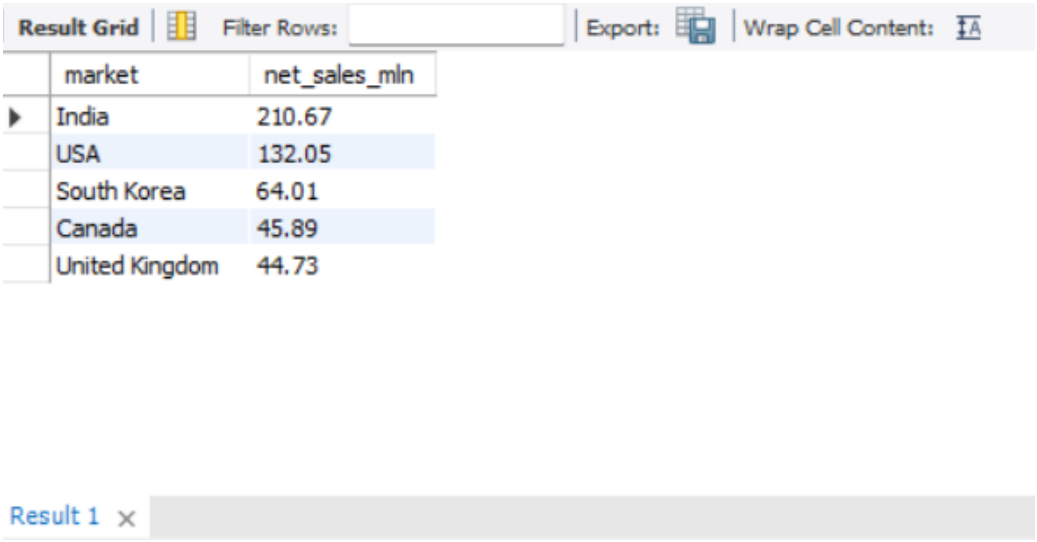
Top Markets and Customers:

Get Top 5 Markets by Net Sales in Fiscal Year 2021:

```
SELECT
        market,
        ROUND(SUM(net_sales) / 1000000, 2) AS net_sales_mln
FROM gdb0041.net_sales
WHERE fiscal_year = 2021
GROUP BY market
ORDER BY net_sales_mln DESC
LIMIT 5;
```

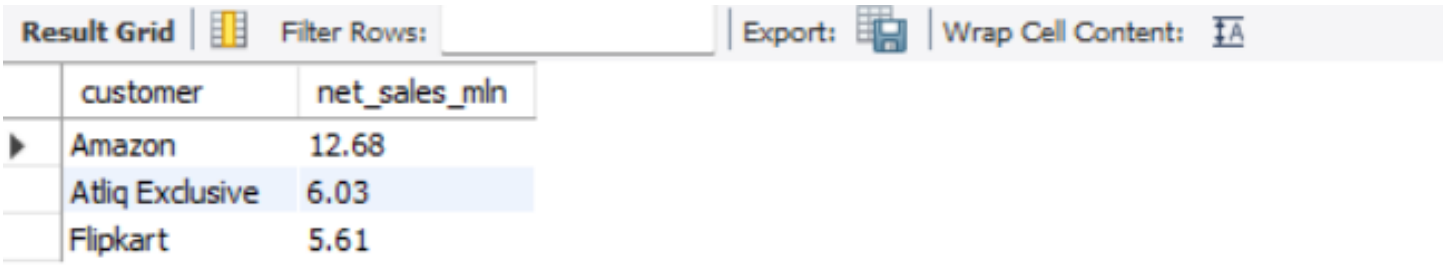
Stored Procedure for Top N Markets by Net Sales for a Given Year:

```
CREATE PROCEDURE get_top_n_markets_by_net_sales(
        IN in_fiscal_year INT,
        IN in_top_n INT
)
BEGIN
        SELECT
                market,
                ROUND(SUM(net_sales) / 1000000, 2) AS net_sales_mln
        FROM net_sales
        WHERE fiscal_year = in_fiscal_year
        GROUP BY market
        ORDER BY net_sales_mln DESC
        LIMIT in_top_n;
END;

call gdb0041.top_n_markets_by_net_sales(2021, 5);
```



Stored Procedure for Top N Customers by Net Sales in a Given Market and Fiscal Year:

```
CREATE PROCEDURE get_top_n_customers_by_net_sales(
        IN in_market VARCHAR(45),
        IN in_fiscal_year INT,
        IN in_top_n INT
)
BEGIN
        SELECT
          customer,
          ROUND(SUM(net_sales) / 1000000, 2) AS net_sales_mln
        FROM net_sales s
        JOIN dim_customer c
          ON s.customer_code = c.customer_code
        WHERE
          s.fiscal_year = in_fiscal_year
      AND s.market = in_market
        GROUP BY customer
        ORDER BY net_sales_mln DESC
        LIMIT in_top_n;
END;


call gdb0041.get_top_n_customers_by_net_sales('India', 2020, 3);
```



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer	net_sales_mln
Amazon	12.68
Atliq Exclusive	6.03
Flipkart	5.61

Result 3

Customer-wise Net Sales Percentage Contribution:

```
WITH cte1 AS (
        SELECT
          c.customer,
          ROUND(SUM(net_sales) / 1000000, 2) AS net_sales_mln
        FROM net_sales s
        JOIN dim_customer c
          ON s.customer_code = c.customer_code
        WHERE s.fiscal_year = 2021
        GROUP BY c.customer
)
SELECT
        *,
        net_sales_mln * 100 / SUM(net_sales_mln) OVER() AS pct_net_sales
FROM cte1
ORDER BY net_sales_mln DESC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer	net_sales_mln	pct_net_sales
Amazon	109.03	13.233402
Atliq Exclusive	79.92	9.700206
Atliq e Store	70.31	8.533803
Sage	27.07	3.285593
Flipkart	25.25	3.064692
Leader	24.52	2.976089
Neptune	21.01	2.550067
Ebay	19.88	2.412914
Electricalsocity	16.25	1.972327
Synthetic	16.10	1.954121
Electricalslytical	15.64	1.898289
Acclaimed Sto...	14.32	1.738075
Propel	14.14	1.716228
Novus	12.91	1.566938
Expression	12.90	1.565724
Reliance Digital	12.75	1.547518

Result 1

Customer-Wise Net Sales Distribution by Region for FY 2021:

```
WITH cte1 AS (
```

```
    SELECT
        c.customer,
        c.region,
        ROUND(SUM(net_sales) / 1000000, 2) AS net_sales_mln
    FROM gdb0041.net_sales n
    JOIN dim_customer c
      ON n.customer_code = c.customer_code
    WHERE fiscal_year = 2021
    GROUP BY c.customer, c.region
)
SELECT
    *,
    net_sales_mln * 100 / SUM(net_sales_mln) OVER(PARTITION BY region) AS pct_share_region
FROM cte1
ORDER BY region, pct_share_region DESC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer	region	net_sales_mln	pct_share_region
Leader	APAC	24.52	5.547511
Sage	APAC	22.85	5.169683
Neptune	APAC	21.01	4.753394
Electricalsocity	APAC	16.25	3.676471
Propel	APAC	14.14	3.199095
Synthetic	APAC	14.14	3.199095
Flipkart	APAC	12.96	2.932127
Novus	APAC	12.91	2.920814
Expression	APAC	12.90	2.918552
Girias	APAC	11.30	2.556561
Vijay Sales	APAC	11.27	2.549774
Ebay	APAC	11.14	2.520362
Reliance Digital	APAC	11.10	2.511312
Electricalslytical	APAC	11.08	2.506787
Lotus	APAC	10.53	2.382353
Ezone	APAC	10.30	2.330317
Viveks	APAC	10.09	2.282805
Croma	APAC	9.88	2.235294
Zone	APAC	6.91	1.563348
Acclaimed Sto...	APAC	5.79	1.309955
Taobao	APAC	4.31	0.975113
Digimarket	APAC	3.97	0.898190
Forward Stores	APAC	3.83	0.866516
Insight	APAC	3.61	0.816742
Sound	APAC	3.44	0.778281

Result 2

Stored Procedure for Top Products Per Division by Quantity Sold:

```
CREATE PROCEDURE get_top_n_products_per_division_by_qty_sold(
        IN in_fiscal_year INT,
        IN in_top_n INT
)
BEGIN
  WITH cte1 AS (
        SELECT
              p.division,
              p.product,
              SUM(s.sold_quantity) AS total_qty
        FROM fact_sales_monthly s
        JOIN dim_product p
          ON p.product_code = s.product_code
        WHERE s.fiscal_year = in_fiscal_year
        GROUP BY p.division, p.product
  ),
  cte2 AS (
    SELECT
      *,
      DENSE_RANK() OVER (PARTITION BY division ORDER BY total_qty DESC) AS drnk
    FROM cte1
  )
  SELECT *
  FROM cte2
  WHERE drnk <= in_top_n;
END;
```

```
call gdb0041.get_top_n_products_by_division_by_qty_sold(2021, 3);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

division	product	total_qty	drnk
N & S	AQ Pen Drive DRC	2034569	1
N & S	AQ Digit SSD	1240149	2
N & S	AQ Clx1	1238683	3
P & A	AQ Gamers Ms	2477098	1
P & A	AQ Maxima Ms	2461991	2
P & A	AQ Master wireless x1 Ms	2448784	3
PC	AQ Digit	135092	1
PC	AQ Gen Y	135031	2
PC	AQ Elite	134431	3

Result 2 ×

Output

SUPPLY CHAIN ANALYTICS

This section of the project focuses on **Supply Chain Analytics**, with an emphasis on creating a robust dataset and analyzing forecast accuracy. The provided SQL scripts include helper tables, stored procedures, and temporary table operations designed to assess and report on supply chain performance. These queries are integral for understanding forecast reliability, enabling data-driven decisions to optimize inventory management, and improving customer satisfaction.

Below are the detailed SQL queries used in this analysis:

Helper Table: fact_act_est Creation

This query creates a consolidated fact table combining actual sales and forecasted quantities for each product and customer. It ensures missing data is accounted for by setting null values to zero.

```
DROP TABLE IF EXISTS fact_act_est;


CREATE TABLE fact_act_est AS
(
    SELECT
        s.date AS date,
```

```
            s.fiscal_year AS fiscal_year,
            s.product_code AS product_code,
            s.customer_code AS customer_code,
            s.sold_quantity AS sold_quantity,
            f.forecast_quantity AS forecast_quantity
        FROM fact_sales_monthly s
        LEFT JOIN fact_forecast_monthly f
        USING (date, customer_code, product_code)
    )
    UNION
(
        SELECT
            f.date AS date,
            f.fiscal_year AS fiscal_year,
            f.product_code AS product_code,
            f.customer_code AS customer_code,
            s.sold_quantity AS sold_quantity,
            f.forecast_quantity AS forecast_quantity
        FROM fact_forecast_monthly f
        LEFT JOIN fact_sales_monthly s
        USING (date, customer_code, product_code)
);

UPDATE fact_act_est
SET sold_quantity = 0
WHERE sold_quantity IS NULL;

UPDATE fact_act_est
SET forecast_quantity = 0
WHERE forecast_quantity IS NULL;
```

Stored Procedure: get_forecast_accuracy

This stored procedure generates a detailed forecast accuracy report for a specified fiscal year. The accuracy is calculated by comparing forecasted and actual sold quantities, and metrics such as net error, absolute error, and accuracy percentage are included.

```
CREATE PROCEDURE `get_forecast_accuracy`(
  IN in_fiscal_year INT
)
BEGIN
  WITH forecast_err_table AS (
    SELECT
        s.customer_code AS customer_code,
        c.customer AS customer_name,
        c.market AS market,
        SUM(s.sold_quantity) AS total_sold_qty,
        SUM(s.forecast_quantity) AS total_forecast_qty,
        SUM(s.forecast_quantity - s.sold_quantity) AS net_error,
            ROUND(SUM(s.forecast_quantity - s.sold_quantity) * 100 / SUM(s.forecast_quantity), 1) AS net_error_pct,
        SUM(ABS(s.forecast_quantity - s.sold_quantity)) AS abs_error,
          ROUND(SUM(ABS(s.forecast_quantity - s.sold_quantity)) * 100 / SUM(s.forecast_quantity), 2) AS abs_error_pct
    FROM fact_act_est s
    JOIN dim_customer c
      ON s.customer_code = c.customer_code
    WHERE s.fiscal_year = in_fiscal_year
    GROUP BY customer_code
  )
  SELECT
    *,
    IF(abs_error_pct > 100, 0, 100.0 - abs_error_pct) AS forecast_accuracy
  FROM forecast_err_table
  ORDER BY forecast_accuracy DESC;
END;

call gdb0041.get_forecast_accuracy(2021);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_code	customer_name	market	total_sold_qty	total_forecast_qty	net_error	net_error_pct	abs_error	abs_error_pct	forecast_accuracy
90013120	Coolblue	Italy	109547	133532	23985	18.0	70467	52.77	47.23
70010048	Atliq e Store	Bangladesh	119439	142010	22571	15.9	75711	53.31	46.69
90023027	Costco	Canada	236189	279962	43773	15.6	149303	53.33	46.67
90023026	Relief	Canada	228988	273492	44504	16.3	146948	53.73	46.27
90017051	Forward Stores	Portugal	86823	118067	31244	26.5	63568	53.84	46.16
90017058	Mbit	Portugal	86860	110195	23335	21.2	59473	53.97	46.03
90023028	walmart	Canada	239081	283323	44242	15.6	153058	54.02	45.98
90023024	Sage	Canada	246397	287233	40836	14.2	155610	54.18	45.82
90013124	Amazon	Italy	110898	136116	25218	18.5	73826	54.24	45.76
90015146	Mbit	Norway	147152	210507	63355	30.1	114189	54.24	45.76
90017054	Flawless Stores	Portugal	84371	114698	30327	26.4	62483	54.48	45.52
70027208	Atliq e Store	Brazil	33713	47321	13608	28.8	25784	54.49	45.51
90015147	Chiptec	Norway	154897	223867	68970	30.8	122100	54.54	45.46
80001019	Neptune	China	1113979	1275248	161269	12.6	695779	54.56	45.44
90015144	Sound	Norway	160074	225637	65563	29.1	123257	54.63	45.37
90009130	Logic Stores	Newzealand	103290	110175	6885	6.2	60225	54.66	45.34
90015149	UniEuro	Norway	142086	212500	70414	33.1	116172	54.67	45.33
90021088	Electricalslytical	United Kin...	224350	323689	99339	30.7	176975	54.67	45.33
90017050	Electricalsara S...	Portugal	85272	114688	29416	25.6	62760	54.72	45.28
70013125	Atliq Exclusive	Italy	101658	123428	21770	17.6	67546	54.73	45.27
90021094	Coolblue	United Kin...	208512	301367	92855	30.8	165043	54.76	45.24
70009134	Atliq e Store	Newzealand	103747	110791	7044	6.4	60726	54.81	45.19
90013118	Fnac-Darty	Italy	101847	126289	24442	19.4	69242	54.83	45.17
70017060	Atliq e Store	Portugal	89925	120744	30819	25.5	66285	54.90	45.10

Result 3