**Finance-and-Supply-Chain-Analytics**

**About ATLIQ Hardware**

AtliQ Hardware is a renowned hardware company specializing in PCs, printers, mice, and computers. Their products are globally recognized and available at Croma and Best Buy stores, as well as online on Amazon and Flipkart.

**Project Overview**

* The project's objective is to extract valuable information from the provided database.
* The database includes details about sales, products, customers, and regions for AtliQ Hardware.
* Specific questions related to sales reports, market analysis, customer behavior, and predicting supply chain needs will be addressed.

**Key Takeaways**

* Used Common Table Expression technique (CTE)
* Worked with User-Defined Functions
* Created Database Views
* Employed JOINS
* Implemented Stored Procedures
* Applied Window Functions like OVER, ROW\_NUMBER, RANK, DENSE\_RANK
* Used Temporary Tables
* Worked with Indexes and Database Creation/Update

**Insights**

* Amazon had the highest net sales in fiscal year 2021, reaching 109.03M, followedby AtliQ Exclusive with 79.92M.
* The Indian market brought in the most net sales in fiscal year 2021, totaling210.67M, with the USA following at 132.05M.
* Amazon made up 13.23% of the total net sales across all customers in the fiscalyear 2021.
* In the APAC region, Amazon had the highest percentage of net sales, accountingfor 12.99% among all customers in 2021.
* Among APAC countries, India ranked first in total gross sales amount.

**General Queries**

**1. Croma India Product Wise Sales Report for Fiscal Year -2021**

Select

s.date, s.product\_code, p.product, p.variant, s.sold\_quantity,

g.gross\_price, ROUND(g.gross\_price\*s.sold\_quantity,2) as gross\_price\_total

from fact\_sales\_monthly s

join dim\_product p

on p.product\_code= s.product\_code

join fact\_gross\_price g

on g.product\_code= s.product\_code

and g.fiscal\_year= get\_fiscal\_year(s.date)

where

customer\_code=90002002 AND

get\_fiscal\_year(date)=2021

ORDER BY date asc;

**2. Total Gross Sales Amount Report For Croma - By Monthly**

select

YEAR(s.date) AS year,

monthname(s.date) AS month,

SUM(g.gross\_price\*s.sold\_quantity) as gross\_price\_total

from fact\_sales\_monthly s

join fact\_gross\_price g

on g.product\_code= s.product\_code

and g.fiscal\_year= get\_fiscal\_year(s.date)

where customer\_code = 90002002

group by s.date

order by s.date asc;

**3. Total Gross Sales Amount Report For Croma - By Fiscal Year**

select

get\_fiscal\_year(date) as fiscal\_year,

round(SUM(g.gross\_price\*s.sold\_quantity)/1000000,2) as yearly\_sales\_in\_mln

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 get\_fiscal\_year(date)

order by fiscal\_year;

**4. Top 5 Customers for the Financial Year - 2021**

select

c.customer, round(sum(net\_sales)/1000000,2) as net\_sales\_mln

FROM net\_sales n

join dim\_customer c

on n.customer\_code=c.customer\_code

where fiscal\_year=2021

group by c.customer

order by net\_sales\_mln desc

limit 5;

**5. Top 5 Products for the Financial Year - 2021**

select

product, round(sum(net\_sales)/1000000,2) as net\_sales\_mln

FROM net\_sales

where fiscal\_year=2021

group by product

order by net\_sales\_mln desc

limit 5;

**6. Top 5 Markets for the Financial Year - 2021**

select

market, round(sum(net\_sales)/1000000,2) as net\_sales\_mln

FROM net\_sales

where fiscal\_year=2021

group by market

order by net\_sales\_mln desc

limit 5;

**7. Top 10 Customers by Net Sales % in Financial Year - 2021**

with cte1 as (

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=2021

group by customer)

select \*,

net\_sales\_mln\*100/sum(net\_sales\_mln) over()as net\_sales\_pct

from cte1

order by net\_sales\_pct desc

limit 10;

**8. Top 10 Customers by Net Sales % in APAC Region & Financial Year - 2021**

with cte1 as (

select

c.customer, c.region,

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

and s.market=c.market

where

s.fiscal\_year=2021 AND c.region = "APAC"

group by c.region, c.customer)

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

limit 10;

**9. Top 2 Markets in Across Region w.r.t their Gross Sales in Financial Year - 2021**

SET SQL\_MODE="";

with cte1 as

(select

c.market,

c.region,

round(sum(gross\_price\_total)/1000000,2) as gross\_sales\_mln

from gross\_sales g

join dim\_customer c

on c.customer\_code=g.customer\_code

where fiscal\_year=2021

group by market),

cte2 as

(select

\*,

dense\_rank() over (partition by region order by gross\_sales\_mln desc) as drnk

from cte1)

select \* from cte2 where drnk<=2;

**10. Supply Chain Statistics Financial Year - 2021**

SET SQL\_MODE="";

WITH forecast\_err\_table as (

SELECT

s.customer\_code,

sum(s.sold\_quantity) as total\_sold\_qty,

sum(s.forecast\_quantity) as total\_forecast\_qty,

ROUND(SUM((forecast\_quantity- sold\_quantity)),2) as net\_err,

ROUND(SUM((forecast\_quantity- sold\_quantity))\*100/ SUM(forecast\_quantity),2)as net\_err\_pct,

ROUND(SUM(abs(forecast\_quantity- sold\_quantity)),2) as abs\_err,

ROUND(SUM(abs(forecast\_quantity- sold\_quantity))\*100/ SUM(forecast\_quantity),2) as abs\_err\_pct

FROM gdb0041.fact\_act\_est s

where s.fiscal\_year=2021

group by s.customer\_code

order by abs\_err\_pct desc)

Select

e.\*, c.customer, c.market,

if(abs\_err\_pct > 100, 0 , (100- abs\_err\_pct)) as forecast\_accuracy

from forecast\_err\_table e

join dim\_customer c

using (customer\_code)

order by forecast\_accuracy desc;

**Stored Procedures**

**1. Finding forecast accuracy for a given year**

CREATE DEFINER=`root`@`localhost` PROCEDURE `get\_forecast\_accuracy`(

in\_fiscal\_year INT

)

BEGIN

SET SQL\_MODE="";

WITH forecast\_err\_table as (

SELECT

s.customer\_code,

sum(s.sold\_quantity) as total\_sold\_qty,

sum(s.forecast\_quantity) as total\_forecast\_qty,

ROUND(SUM((forecast\_quantity- sold\_quantity)),2) as net\_err,

ROUND(SUM((forecast\_quantity- sold\_quantity))\*100/ SUM(forecast\_quantity),2)as net\_err\_pct,

ROUND(SUM(abs(forecast\_quantity- sold\_quantity)),2) as abs\_err,

ROUND(SUM(abs(forecast\_quantity- sold\_quantity))\*100/ SUM(forecast\_quantity),2) as abs\_err\_pct

FROM gdb0041.fact\_act\_est s

where s.fiscal\_year=in\_fiscal\_year

group by s.customer\_code

order by abs\_err\_pct desc)

Select

e.\*, c.customer, c.market,

if(abs\_err\_pct > 100, 0 , (100- abs\_err\_pct)) as forecast\_accuracy

from forecast\_err\_table e

join dim\_customer c

using (customer\_code)

order by forecast\_accuracy desc;

END

**2. determine the market badge if tota1\_qty > 5 million, then gold else it is a silver by market and year**

CREATE DEFINER=`root`@`localhost` PROCEDURE `get\_market\_badge`(

IN in\_market varchar(45),

IN in\_fiscal\_year year,

OUT out\_badge varchar(45)

)

BEGIN

declare total\_qty int default 0;

# set default market to be in india

if in\_market="" then

set in\_market= "India";

end if;

# retrieve total qty for a given market and fical year

select

SUM(sold\_quantity) into total\_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

group by c.market ;

# determine the market badge if tota\_qty > 5 million, then gold

# else it is a silver

if total\_qty > 5000000 then

set out\_badge = "Gold";

else

set out\_badge = "Silver";

end if;

END

**3. Getting monthly gross sales for customers**

CREATE DEFINER=`root`@`localhost` PROCEDURE `get\_monthly\_gross\_sales\_for\_customer`(in\_customer\_codes TEXT)

BEGIN

select s.date,SUM(Round(g.gross\_price\*s.sold\_quantity,2)) as monthly\_sales

from fact\_sales\_monthly s

join fact\_gross\_price g

on g.product\_code= s.product\_code

and g.fiscal\_year= get\_fiscal\_year(s.date)

where FIND\_IN\_SET(s.customer\_code, in\_customer\_codes)>0

group by date;

END

**4. Determining top N customers w.r.t net sales using fiscal year**

CREATE DEFINER=`root`@`localhost` PROCEDURE `get\_top\_n\_customers\_by\_net\_sales`(

in\_fiscal\_year INT,

in\_top\_n INT

)

BEGIN

select

c.customer, round(sum(net\_sales)/1000000,2) as net\_sales\_mln

FROM net\_sales n

join dim\_customer c

on n.customer\_code=c.customer\_code

where fiscal\_year= in\_fiscal\_year

group by c.customer

order by net\_sales\_mln desc

limit in\_top\_n;

END

**5. Determining Top N customers w.r.t market and net sales**

CREATE DEFINER=`root`@`localhost` PROCEDURE `get\_top\_n\_customers\_w.r.t\_market\_by\_net\_sales`(

in\_market VARCHAR(45),

in\_fiscal\_year INT,

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

**6. Determining Top N markets by net sales**

CREATE DEFINER=`root`@`localhost` PROCEDURE `get\_top\_n\_markets\_by\_net\_sales`(

in\_fiscal\_year INT,

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

**7. Determining Top N products by net sales**

CREATE DEFINER=`root`@`localhost` PROCEDURE `get\_top\_n\_products\_by\_net\_sales`(

in\_fiscal\_year INT,

in\_top\_n INT

)

BEGIN

select

product, round(sum(net\_sales)/1000000,2) as net\_sales\_mln

FROM net\_sales

where fiscal\_year= in\_fiscal\_year

group by product

order by net\_sales\_mln desc

limit in\_top\_n;

END

**8. Determining Top N products per Division by Sold Quantity**

CREATE DEFINER=`root`@`localhost` PROCEDURE `get\_top\_n\_products\_per\_division\_by\_qty\_sold`(

in\_fiscal\_year INT,

in\_top\_n INT

)

BEGIN

with cte1 as

(select

p.division,

p.product,

sum(sold\_quantity) as total\_qty

from fact\_sales\_monthly s

join dim\_product p

on p.product\_code=s.product\_code

where fiscal\_year=in\_fiscal\_year

group by 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

**9. Determining Top N markets by Net Sales**

CREATE DEFINER=`root`@`localhost` PROCEDURE `get\_top\_n\_markets\_by\_net\_sales`(

in\_fiscal\_year INT,

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

**Views**

**1. Gross Sales**

CREATE

ALGORITHM = UNDEFINED

DEFINER = `root`@`localhost`

SQL SECURITY DEFINER

VIEW `gross\_sales` AS

SELECT

`s`.`date` AS `date`,

`s`.`fiscal\_year` AS `fiscal\_year`,

`c`.`market` AS `market`,

`s`.`customer\_code` AS `customer\_code`,

`s`.`product\_code` AS `product\_code`,

`p`.`product` AS `product`,

`p`.`variant` AS `variant`,

`s`.`sold\_quantity` AS `sold\_quantity`,

`g`.`gross\_price` AS `gross\_price\_per\_item`,

ROUND((`s`.`sold\_quantity` \* `g`.`gross\_price`),

2) AS `gross\_price\_total`

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` `pre` ON (((`pre`.`customer\_code` = `s`.`customer\_code`)

AND (`pre`.`fiscal\_year` = `s`.`fiscal\_year`))))

**2. Net Sales**

CREATE

ALGORITHM = UNDEFINED

DEFINER = `root`@`localhost`

SQL SECURITY DEFINER

VIEW `net\_sales` AS

SELECT

`sales\_postinv\_discount`.`date` AS `date`,

`sales\_postinv\_discount`.`fiscal\_year` AS `fiscal\_year`,

`sales\_postinv\_discount`.`customer\_code` AS `customer\_code`,

`sales\_postinv\_discount`.`market` AS `market`,

`sales\_postinv\_discount`.`product\_code` AS `product\_code`,

`sales\_postinv\_discount`.`product` AS `product`,

`sales\_postinv\_discount`.`variant` AS `variant`,

`sales\_postinv\_discount`.`sold\_quantity` AS `sold\_quantity`,

`sales\_postinv\_discount`.`gross\_price\_total` AS `gross\_price\_total`,

`sales\_postinv\_discount`.`pre\_invoice\_discount\_pct` AS `pre\_invoice\_discount\_pct`,

`sales\_postinv\_discount`.`net\_invoice\_sales` AS `net\_invoice\_sales`,

`sales\_postinv\_discount`.`post\_invoice\_discount\_pct` AS `post\_invoice\_discount\_pct`,

(`sales\_postinv\_discount`.`net\_invoice\_sales` \* (1 - `sales\_postinv\_discount`.`post\_invoice\_discount\_pct`)) AS `net\_sales`

FROM

`sales\_postinv\_discount`

**3. Sales ( Post Invoice Discount )**

CREATE

ALGORITHM = UNDEFINED

DEFINER = `root`@`localhost`

SQL SECURITY DEFINER

VIEW `sales\_postinv\_discount` AS

SELECT

`s`.`date` AS `date`,

`s`.`fiscal\_year` AS `fiscal\_year`,

`s`.`customer\_code` AS `customer\_code`,

`s`.`market` AS `market`,

`s`.`product\_code` AS `product\_code`,

`s`.`product` AS `product`,

`s`.`variant` AS `variant`,

`s`.`sold\_quantity` AS `sold\_quantity`,

`s`.`gross\_price\_total` AS `gross\_price\_total`,

`s`.`pre\_invoice\_discount\_pct` AS `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`))))

**4. Sales ( Pre Invoice Discount )**

CREATE

ALGORITHM = UNDEFINED

DEFINER = `root`@`localhost`

SQL SECURITY DEFINER

VIEW `sales\_preinv\_discount` AS

SELECT

`s`.`date` AS `date`,

`s`.`fiscal\_year` AS `fiscal\_year`,

`s`.`customer\_code` AS `customer\_code`,

`c`.`market` AS `market`,

`s`.`product\_code` AS `product\_code`,

`p`.`product` AS `product`,

`p`.`variant` AS `variant`,

`s`.`sold\_quantity` AS `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` AS `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` `pre` ON (((`pre`.`customer\_code` = `s`.`customer\_code`)

AND (`pre`.`fiscal\_year` = `s`.`fiscal\_year`))))

**Functions**

**1. Determining Fiscal Quarter**

CREATE DEFINER=`root`@`localhost` FUNCTION `get\_fiscal\_quarter`(

calendar\_date date

) RETURNS char(2) CHARSET utf8mb4

DETERMINISTIC

BEGIN

declare m TINYINT;

declare qtr CHAR(2);

SET m=month(calendar\_date) ;

CASE

when m in (9,10,11) then

SET qtr= "Q1";

when m in (12,1,2) then

SET qtr= "Q2";

when m in (3,4,5) then

SET qtr= "Q3";

when m in (6,7,8) then

SET qtr= "Q4";

END CASE;

return qtr;

END

**2. Determing Fiscal year**

CREATE DEFINER=`root`@`localhost` 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