SQL PROJECT

Presented by : Jyotiprova Ghosh

CONCEPT

- Gross Sales
- Pre Invoice Discount
- Post Invoice Discount
- Net Invoice sales
- Net sales
- Market Share %
- Stored Procedure
- Functions
- Views

OVERVIEW

- function fiscal quarter
- function fiscal year
- monthly sales report for customer Croma
- yearly sales report for customer Croma
- stored procedure for monthly gross sales
- stored procedure for market badge
- pre invoice discount [views]
- post invoice discount [views]
- gross sales [views]
- region wise market share %
- customer wise net sales contribution %

FINANCE ANALYTICS

INTRODUCTION

- Objective : Developed and executed SQL queries to analyze large datasets for generating actionable insights in the finance domain.
- Data Management : Managed and optimized financial datasets, including net sales, gross sales , and market data, ensuring data accuracy and integrity.
- Analytical Techniques : Utilized advanced SQL techniques (joins, subqueries, window functions, and CTEs) , stored procedure , views , functions to conduct in-depth analysis, identify key financial trends, and support strategic decision-making.
- Results : Delivered comprehensive financial reports and dashboards that enhanced business intelligence and improved financial forecasting accuracy.
- Tools & Technologies : Proficiently used MySQL , Excel to handle data management, perform complex queries, and automate reporting processes.

FUNCTIONS

CREATE A FUNCTION FOR GET 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);

    IF m IN (9, 10, 11) THEN
        SET qtr = 'Q1';
    ELSEIF m IN (12, 1, 2) THEN
        SET qtr = 'Q2';
    ELSEIF m IN (3, 4, 5) THEN
        SET qtr = 'Q3';
    ELSE
        SET qtr = 'Q4';
    END IF ;
    RETURN qtr ;
END
```

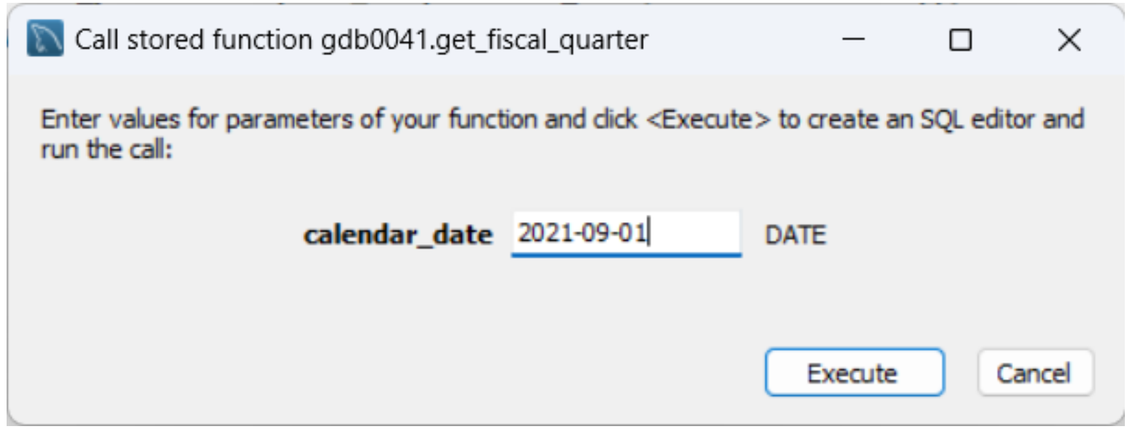
NOTE

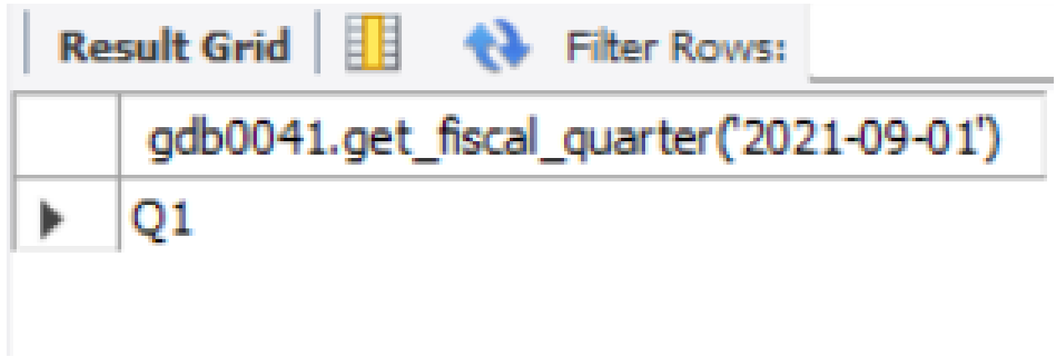
MONTH (9,10,11) - Q1

MONTH (12,1,2) - Q2

MONTH (3,4,5) - Q3

MONTH (6,7,8) - Q4





FUNCTIONS

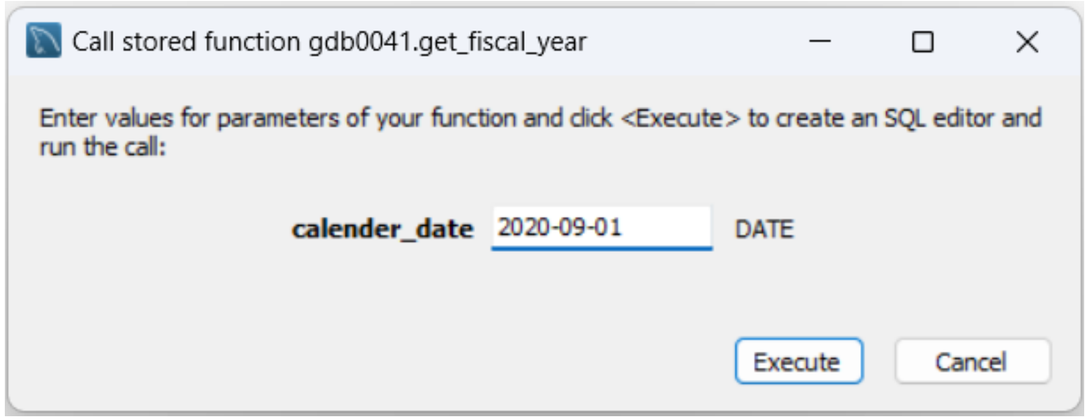
CREATE A FUNCTION FOR GET FISCAL YEAR

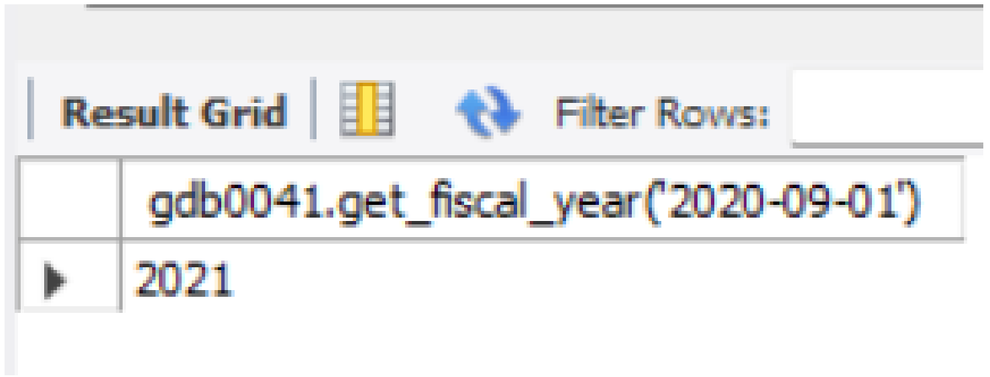
```
CREATE DEFINER=`root`@`localhost` FUNCTION `get_fiscal_year`(
calender_date DATE) RETURNS int
    DETERMINISTIC

BEGIN
DECLARE fiscal_year INT ;

SET fiscal_year = YEAR(DATE_ADD(calender_date,INTERVAL 4 MONTH));
RETURN fiscal_year ;
END
```

NOTE : Fiscal Year starts from september

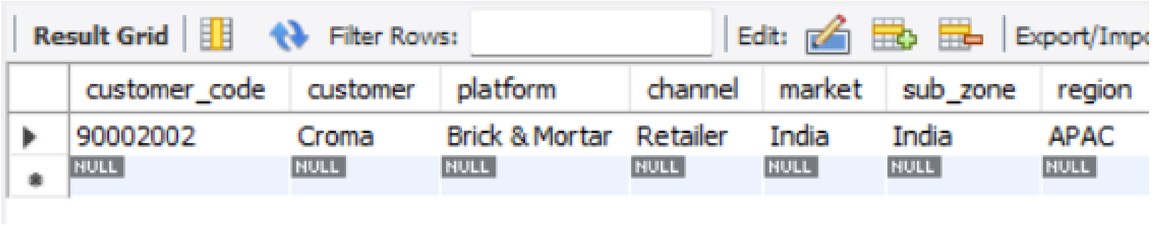




QUERY

FIND OUT THE CUSTOMER CODE FOR CROMA

```
SELECT
  *
FROM
  dim_customer
WHERE
  customer LIKE '%croma%'
```



customer_code	customer	platform	channel	market	sub_zone	region
90002002	Croma	Brick & Mortar	Retailer	India	India	APAC
NULL	NULL	NULL	NULL	NULL	NULL	NULL

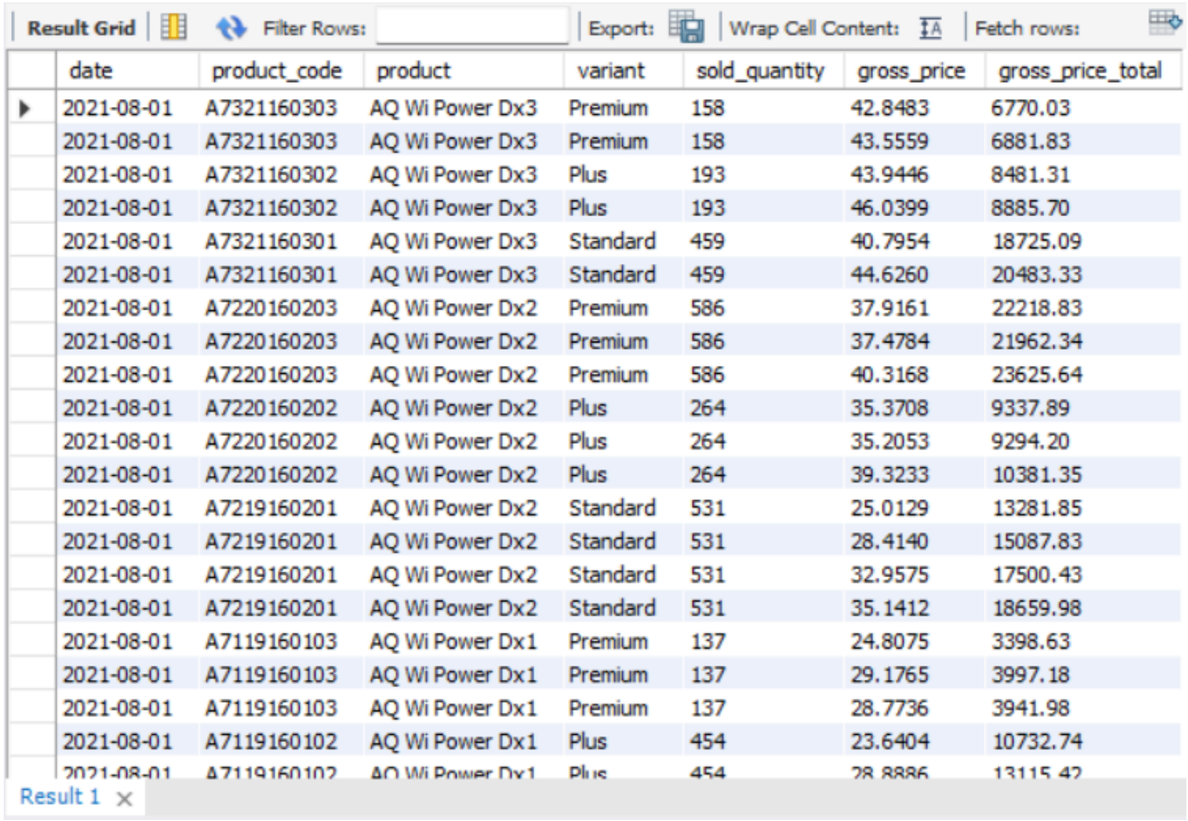
NOTE : WE CLEARLY SEE THAT CUSTOMER_CODE FOR CROMA 90002002

USING THIS CUSTOMER CODE WE CAN DO FURTHER ANALYSIS

QUERY

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 AS s
        JOIN
    dim_product AS p ON p.product_code = s.product_code
        JOIN
    fact_gross_price AS g ON g.product_code = s.product_code
        AND GET_FISCAL_YEAR(s.date)
WHERE
    customer_code = 90002002
        AND GET_FISCAL_YEAR(date) = 2021
ORDER BY date DESC;
```



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

date	product_code	product	variant	sold_quantity	gross_price	gross_price_total
2021-08-01	A7321160303	AQ Wi Power Dx3	Premium	158	42.8483	6770.03
2021-08-01	A7321160303	AQ Wi Power Dx3	Premium	158	43.5559	6881.83
2021-08-01	A7321160302	AQ Wi Power Dx3	Plus	193	43.9446	8481.31
2021-08-01	A7321160302	AQ Wi Power Dx3	Plus	193	46.0399	8885.70
2021-08-01	A7321160301	AQ Wi Power Dx3	Standard	459	40.7954	18725.09
2021-08-01	A7321160301	AQ Wi Power Dx3	Standard	459	44.6260	20483.33
2021-08-01	A7220160203	AQ Wi Power Dx2	Premium	586	37.9161	22218.83
2021-08-01	A7220160203	AQ Wi Power Dx2	Premium	586	37.4784	21962.34
2021-08-01	A7220160203	AQ Wi Power Dx2	Premium	586	40.3168	23625.64
2021-08-01	A7220160202	AQ Wi Power Dx2	Plus	264	35.3708	9337.89
2021-08-01	A7220160202	AQ Wi Power Dx2	Plus	264	35.2053	9294.20
2021-08-01	A7220160202	AQ Wi Power Dx2	Plus	264	39.3233	10381.35
2021-08-01	A7219160201	AQ Wi Power Dx2	Standard	531	25.0129	13281.85
2021-08-01	A7219160201	AQ Wi Power Dx2	Standard	531	28.4140	15087.83
2021-08-01	A7219160201	AQ Wi Power Dx2	Standard	531	32.9575	17500.43
2021-08-01	A7219160201	AQ Wi Power Dx2	Standard	531	35.1412	18659.98
2021-08-01	A7119160103	AQ Wi Power Dx1	Premium	137	24.8075	3398.63
2021-08-01	A7119160103	AQ Wi Power Dx1	Premium	137	29.1765	3997.18
2021-08-01	A7119160103	AQ Wi Power Dx1	Premium	137	28.7736	3941.98
2021-08-01	A7119160102	AQ Wi Power Dx1	Plus	454	23.6404	10732.74
2021-08-01	A7119160102	AQ Wi Power Dx1	Plus	454	28.8886	13115.42

Result 1

QUERY

CROMA INDIA MONTHLY SALES REPORT

```
SELECT YEAR(s.date) as Year,
MONTHNAME(s.date) as month,
SUM(g.gross_price*s.sold_quantity) as Total_gross_price
FROM fact_sales_monthly as s
JOIN fact_gross_price as 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
```

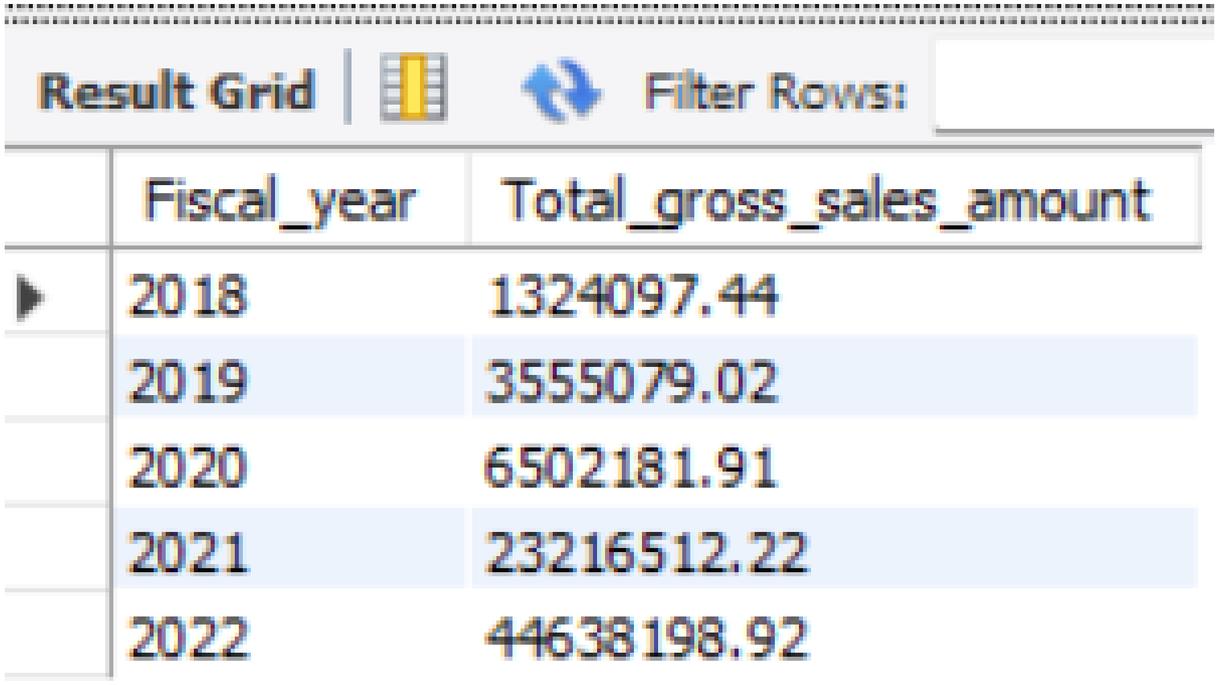
Result Grid | Filter Rows:

Year	month	Total_gross_price
2017	September	122407.5582
2017	October	162687.5716
2017	December	245673.8042
2018	January	127574.7372
2018	February	144799.5182
2018	April	130643.8976
2018	May	139165.0975
2018	June	125735.3786
2018	August	125409.8801
2018	September	343337.1651
2018	October	440562.0754
2018	December	653944.7486
2019	January	359025.0186
2019	February	356607.1729
2019	April	379549.6850
2019	May	340152.2349

QUERY

CROMA INDIA YEARLY SALES REPORT

```
SELECT
    get_fiscal_year(date) AS Fiscal_year,
    ROUND(SUM(g.gross_price*sold_quantity),2) as Total_gross_sales_amount
FROM
    fact_sales_monthly AS s
        JOIN
    fact_gross_price AS g ON g.product_code = s.product_code
        AND g.fiscal_year = GET_FISCAL_YEAR(s.date)
WHERE
    customer_code = 90002002
GROUP BY get_fiscal_year(date)
ORDER BY Fiscal_year ASC
```

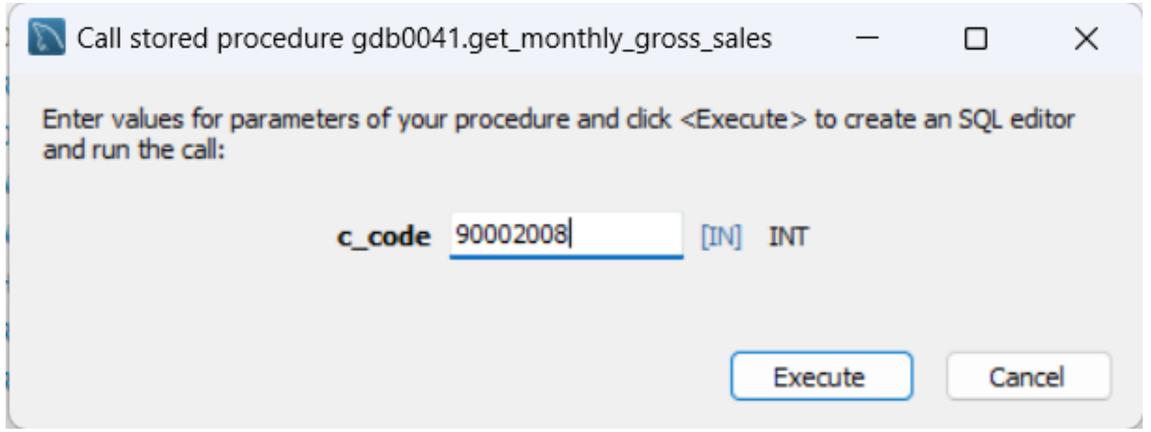


Fiscal_year	Total_gross_sales_amount
2018	1324097.44
2019	3555079.02
2020	6502181.91
2021	23216512.22
2022	44638198.92

STORE PROCEDURE

CREATE A STORE PROCEDURE FOR MONTHLY GROSS SALES

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `get_monthly_gross_sales`(
c_code INT
)
BEGIN
 SELECT YEAR(s.date) as Year,
 MONTHNAME(s.date) as month,
 ROUND(SUM(g.gross_price*s.sold_quantity),2) as Total_gross_price
 FROM fact_sales_monthly as s
 JOIN fact_gross_price as g
 ON g.product_code = s.product_code
 AND g.fiscal_year = get_fiscal_year(s.date)
 WHERE customer_code = c_code
 GROUP BY s.date
 ORDER BY s.date ASC;
END
```



Result Grid | Filter Rows:

Year	month	Total_gross_price
2017	October	273977.43
2017	November	354738.64
2017	December	367282.43
2018	February	216000.81
2018	March	245298.47
2018	April	237801.04
2018	June	223228.83
2018	July	213691.77
2018	August	218390.66
2018	October	745687.39
2018	November	935890.85
2018	December	1026330.72

Result 2

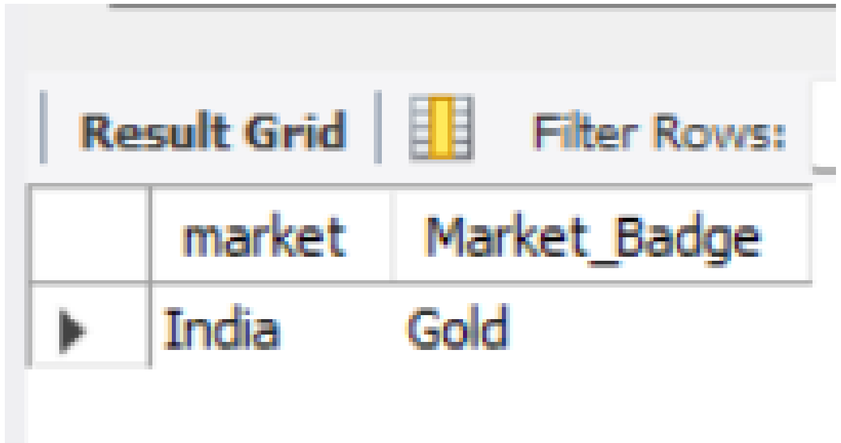
STORE PROCEDURE

CREATE A STORE PROCEDURE FOR MARKET BADGE

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `get_market_badge`(
input_market varchar(20),
input_fiscal_year YEAR)
BEGIN

# If anyone does not put market then the default market should be India
IF input_market = " " THEN
SET input_market = "India";
END IF ;

 SELECT
   d.market,
 CASE WHEN SUM(sold_quantity) > 5000000 THEN "Gold"
 ELSE "Silver"
 END AS Market_Badge
 FROM dim_customer AS d
 JOIN fact_sales_monthly as s
 ON d.customer_code = s.customer_code
   WHERE d.market = input_market AND get_fiscal_year(s.date)= input_fiscal_year
 GROUP BY d.market;
END
```



VIEWS

CREATE A VIEW FOR 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`))))
```

	variant	sold_quantity	gross_price_per_item	gross_price_total	pre_invoice_discount_pct
....	Standard	51	15.3952	785.16	0.0824
....	Standard	77	15.3952	1185.43	0.2956
....	Standard	17	15.3952	261.72	0.0536
....	Standard	6	15.3952	92.37	0.2378
....	Standard	5	15.3952	76.98	0.1057
....	Standard	7	15.3952	107.77	0.1875
....	Standard	29	15.3952	446.46	0.0700
....	Standard	34	15.3952	523.44	0.2551
....	Standard	22	15.3952	338.69	0.0953
....	Standard	5	15.3952	76.98	0.1896
....	Standard	10	15.3952	153.95	0.0521
....	Standard	4	15.3952	61.58	0.2046
....	Standard	0	15.3952	0.00	0.0984
....	Standard	0	15.3952	0.00	0.2620
....	Standard	1	15.3952	15.40	0.0587
....	Standard	1	15.3952	15.40	0.2501
....	Standard	1	15.3952	15.40	0.1937
....	Standard	20	15.3952	307.90	0.2025

VIEWS

CREATE A VIEW FOR 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`))))
```

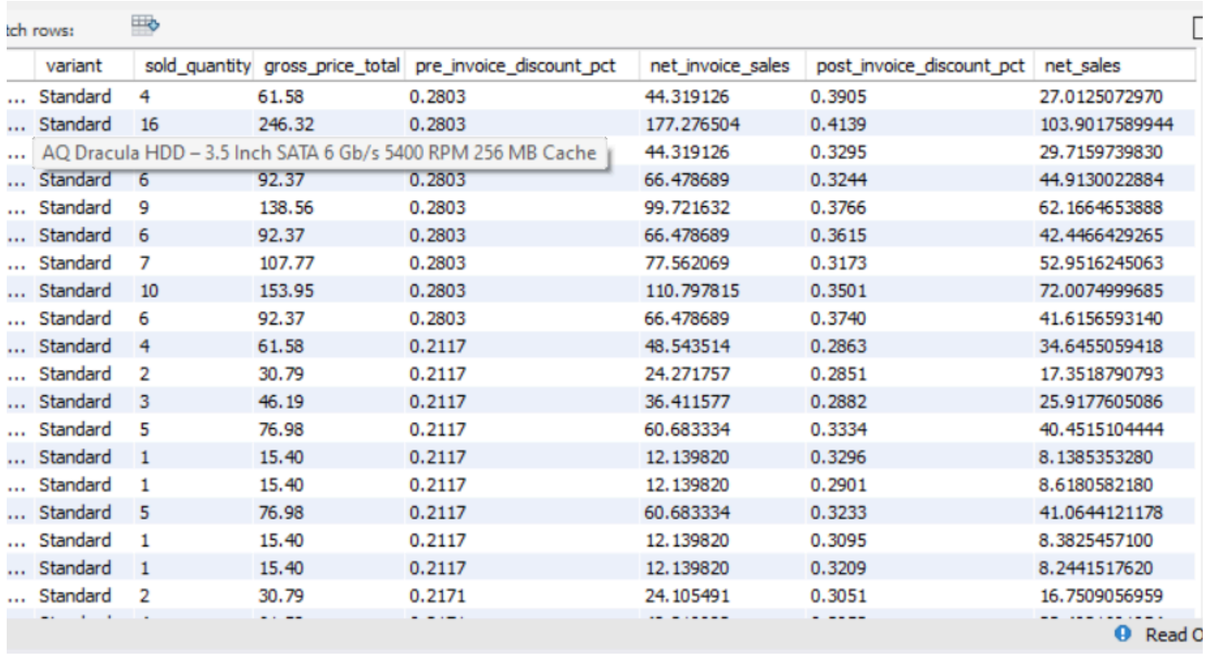
Fetch rows:

	variant	sold_quantity	gross_price_total	pre_invoice_discount_pct	net_invoice_sales	post_invoice_discount_pct
...	Standard	4	61.58	0.2803	44.319126	0.3905
...	Standard	16	246.32	0.2803	177.276504	0.4139
...	Standard	4	61.58	0.2803	44.319126	0.3295
...	Standard	6	92.37	0.2803	66.478689	0.3244
...	Standard	9	138.56	0.2803	99.721632	0.3766
...	Standard	6	92.37	0.2803	66.478689	0.3615
...	Standard	7	107.77	0.2803	77.562069	0.3173
...	Standard	10	153.95	0.2803	110.797815	0.3501
...	Standard	6	92.37	0.2803	66.478689	0.3740
...	Standard	4	61.58	0.2117	48.543514	0.2863
...	Standard	2	30.79	0.2117	24.271757	0.2851
...	Standard	3	46.19	0.2117	36.411577	0.2882
...	Standard	5	76.98	0.2117	60.683334	0.3334
...	Standard	1	15.40	0.2117	12.139820	0.3296
...	Standard	1	15.40	0.2117	12.139820	0.2901
...	Standard	5	76.98	0.2117	60.683334	0.3233
...	Standard	1	15.40	0.2117	12.139820	0.3095
...	Standard	1	15.40	0.2117	12.139820	0.3209
...	Standard	2	30.79	0.2171	24.105491	0.3051

VIEWS

CREATE A VIEW FOR NET SALES

```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `net_sales` AS
    SELECT
     *,
                            ((1    -    `sales_postinv_discount`.`post_invoice_discount_pct`)    *
`sales_postinv_discount`.`net_invoice_sales`) AS `net_sales`
    FROM
        `sales_postinv_discount`
```



tch rows:

	variant	sold_quantity	gross_price_total	pre_invoice_discount_pct	net_invoice_sales	post_invoice_discount_pct	net_sales
...	Standard	4	61.58	0.2803	44.319126	0.3905	27.0125072970
...	Standard	16	246.32	0.2803	177.276504	0.4139	103.9017589944
...	Standard	4	61.58	0.2803	44.319126	0.3295	29.7159739830
...	Standard	6	92.37	0.2803	66.478689	0.3244	44.9130022884
...	Standard	9	138.56	0.2803	99.721632	0.3766	62.1664653888
...	Standard	6	92.37	0.2803	66.478689	0.3615	42.4466429265
...	Standard	7	107.77	0.2803	77.562069	0.3173	52.9516245063
...	Standard	10	153.95	0.2803	110.797815	0.3501	72.0074999685
...	Standard	6	92.37	0.2803	66.478689	0.3740	41.6156593140
...	Standard	4	61.58	0.2117	48.543514	0.2863	34.6455059418
...	Standard	2	30.79	0.2117	24.271757	0.2851	17.3518790793
...	Standard	3	46.19	0.2117	36.411577	0.2882	25.9177605086
...	Standard	5	76.98	0.2117	60.683334	0.3334	40.4515104444
...	Standard	1	15.40	0.2117	12.139820	0.3296	8.1385353280
...	Standard	1	15.40	0.2117	12.139820	0.2901	8.6180582180
...	Standard	5	76.98	0.2117	60.683334	0.3233	41.0644121178
...	Standard	1	15.40	0.2117	12.139820	0.3095	8.3825457100
...	Standard	1	15.40	0.2117	12.139820	0.3209	8.2441517620
...	Standard	2	30.79	0.2171	24.105491	0.3051	16.7509056959

AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 RPM 256 MB Cache

Read O

VIEWS

CREATE A VIEW FOR GROSS SALES

```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `gross_sales` AS
    SELECT
        `s`.`date` AS `date`,`g`.`fiscal_year` AS `fiscal_year`,
        `c`.`customer_code` AS `customer_code`,
        `c`.`customer` AS `customer`, `c`.`market` AS `market`,
        `d`.`product_code` AS `product_code`, `d`.`product` AS `product`,
        `d`.`variant` AS `variant`,
        `s`.`sold_quantity` AS `sold_quantity`,
        `g`.`gross_price` AS `gross_price_per_item`,
        ROUND((`g`.`gross_price` * `s`.`sold_quantity`),
                2) AS `gross_price_total`
    FROM
        (((`fact_sales_monthly` `s`
        JOIN `dim_customer` `c` ON ((`c`.`customer_code` = `s`.`customer_code`)))
        JOIN `fact_gross_price` `g` ON ((`g`.`product_code` = `s`.`product_code`)))
        JOIN `dim_product` `d` ON ((`d`.`product_code` = `g`.`product_code`)))
```

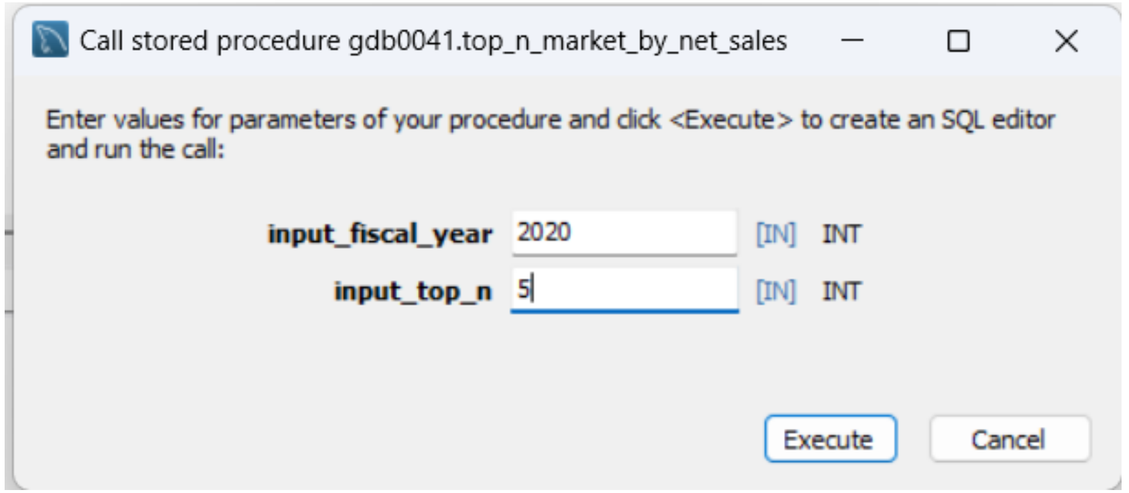
	variant	sold_quantity	gross_price_per_item	gross_price_total
..	Standard	51	15.3952	785.16
..	Standard	51	14.4392	736.40
..	Standard	51	16.2323	827.85
..	Standard	51	19.0573	971.92
..	Standar	77	15.3952	1185.43
..	Standard	77	14.4392	1111.82
..	Standard	77	16.2323	1249.89
..	Standard	77	19.0573	1467.41
..	Standard	17	15.3952	261.72
..	Standard	17	14.4392	245.47
..	Standard	17	16.2323	275.95
..	Standard	17	19.0573	323.97
..	Standard	6	15.3952	92.37
..	Standard	6	14.4392	86.64
..	Standard	6	16.2323	97.39
..	Standard	6	19.0573	114.34
..	Standard	5	15.3952	76.98
..	Standard	5	14.4392	72.20
..	Standard	5	16.2323	81.16

Standard

STORE PROCEDURE

CREATE A STORE PROCEDURE FOR TOP N MARKET BY NET SALES

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `top_n_market_by_net_sales`(
input_fiscal_year INT,
input_top_n INT )
BEGIN
select market,
round(sum(net_sales)/1000000,2) as net_sales_million
from net_sales
where fiscal_year = input_fiscal_year
group by market
order by net_sales_million desc
limit input_top_n;
END
```



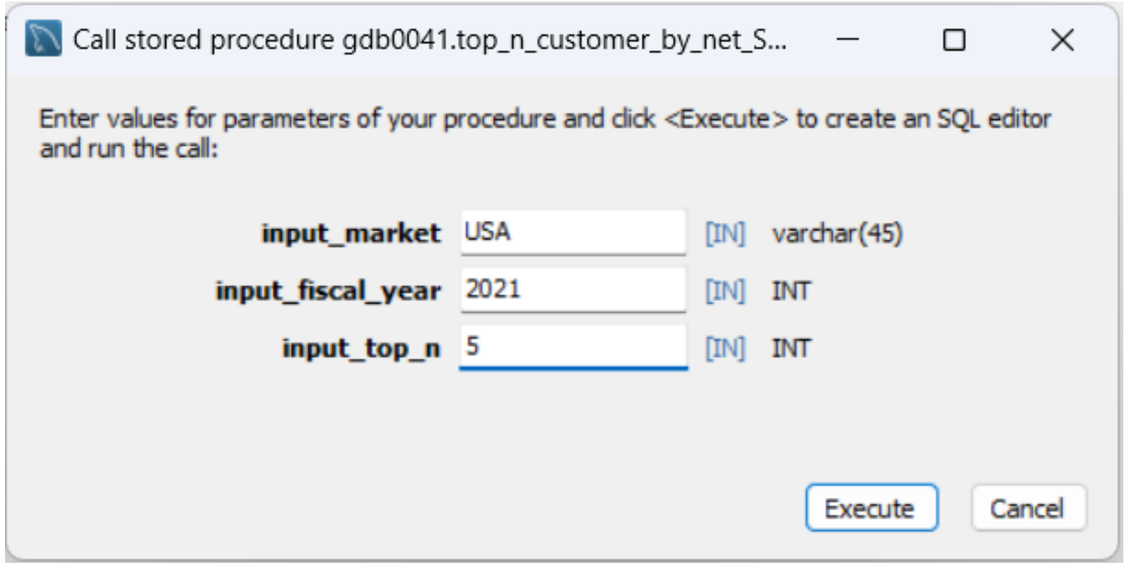
Result Grid | Filter Rows:

market	net_sales_million
India	64.73
USA	46.35
South Korea	22.38
Philiphines	17.45
Canada	15.87

STORE PROCEDURE

CREATE A STORE PROCEDURE FOR TOP N CUSTOMER BY NET SALES

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `top_n_products_by_net_sales`(
input_fiscal_year int,
input_top_n int
)
BEGIN
 select product,
 round(sum(net_sales)/1000000,2) as net_sales_millions
 from net_sales
   where fiscal_year = input_fiscal_year
 group by product
 order by net_sales_millions desc
   limit input_top_n;
END
```



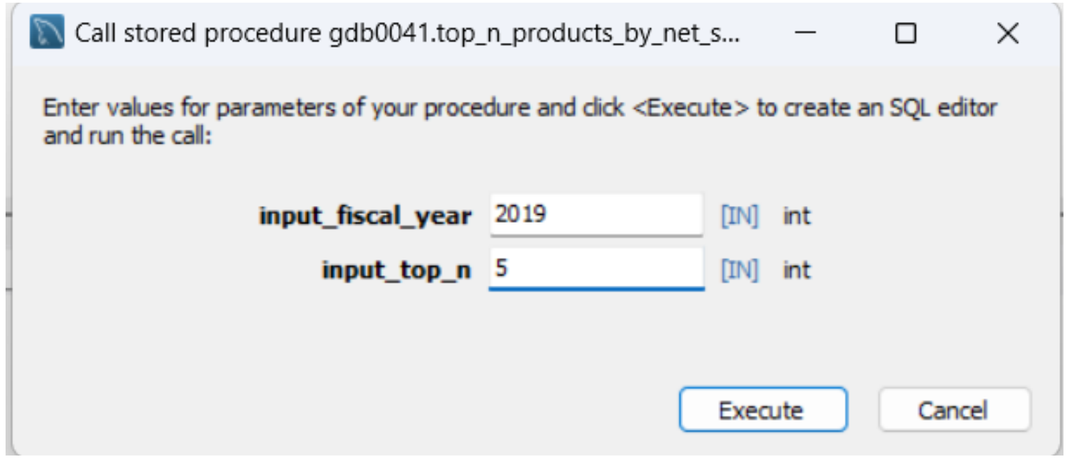
Result Grid | Filter Rows:

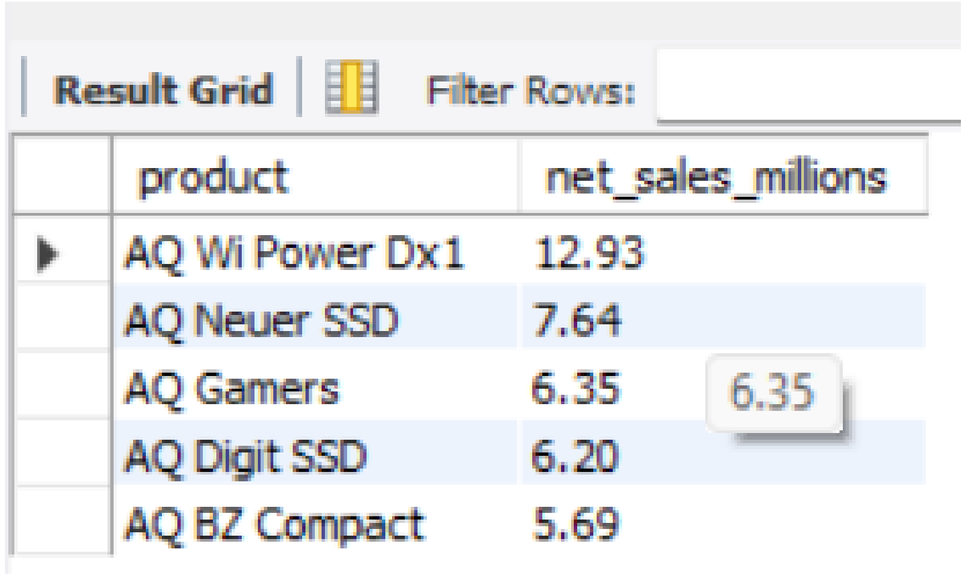
customer	net_sales_millions
Amazon	20.09
Flipkart	10.35
Atliq Exclusive	10.15
Path	9.10
Ebay	8.74

STORE PROCEDURE

CREATE A STORE PROCEDURE FOR TOP N PRODUCTS BY NET SALES

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `top_n_customer_by_net_Sales`(
input_market varchar(45),
input_fiscal_year INT,
input_top_n INT)
BEGIN
 select c.customer,
 round(sum(net_sales)/1000000,2) as net_sales_millions
 from net_sales as s
 join dim_customer as c
 on s.customer_code = c.customer_code
   where fiscal_year = input_fiscal_year and s.market = input_market
 group by c.customer
 order by net_sales_millions desc
   limit input_top_n;
END
```





QUERY

REGION WISE MARKET SHARE %

```
with cte as (select c.customer,c.region,
round(sum(net_sales)/1000000,2) as net_sales_millions
from net_sales as s
join dim_customer as c
on s.customer_code = c.customer_code
where s.fiscal_year = 2021
group by c.customer,c.region)

select *,
net_sales_millions*100/sum(net_sales_millions) over (partition by region) as contribution_pct
from cte
order by region,net_sales_millions desc
```

Result Grid | Filter Rows: | Export: | Wr

customer	region	net_sales_millions	contribution_pct
Amazon	APAC	57.41	12.988688
Atliq Exclusive	APAC	51.58	11.669683
Atliq e Store	APAC	36.97	8.364253
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

Result 2

QUERY

TOP 10 CUSTOMER BY % CONTRIBUTION BY NET SALES

```
with cte1 as (
select customer,
round(sum(net_sales)/1000000,2) as net_Sales_millions
from net_sales as s
join dim_customer as c
on s.customer_code = c.customer_code
where s.fiscal_year = 2021
group by customer)

select customer,
net_sales_millions,
net_sales_millions*100/sum(net_sales_millions) over() as contribution_pct
from cte1
group by customer
order by net_sales_millions desc
limit 10;
```

customer	net_sales_millions	contribution_pct
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

THANK YOU