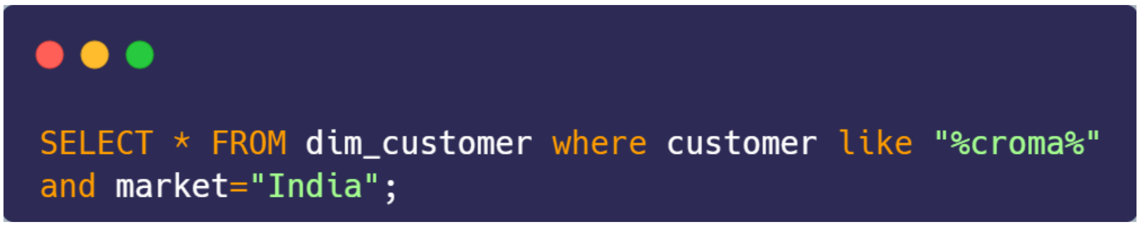
SQL Ad-Hoc Analysis: Consumer Goods

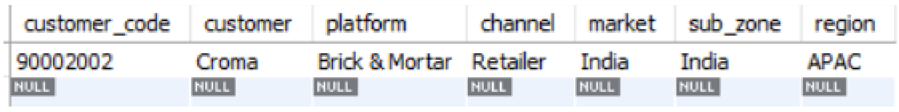
Document by Shwetha Munigala

Extracting the customer code for a specific customer.



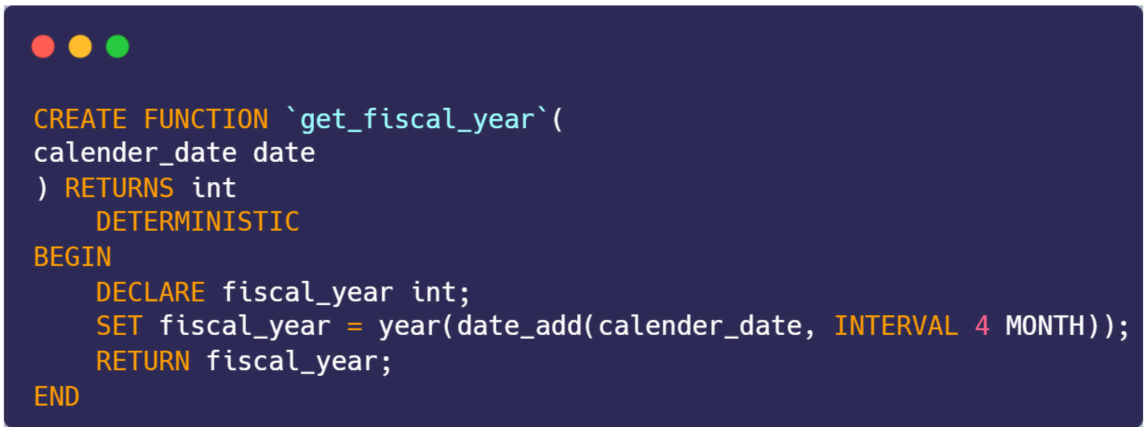
```sql
SELECT * FROM dim_customer where customer like "%croma%"
and market="India";
```

Output:



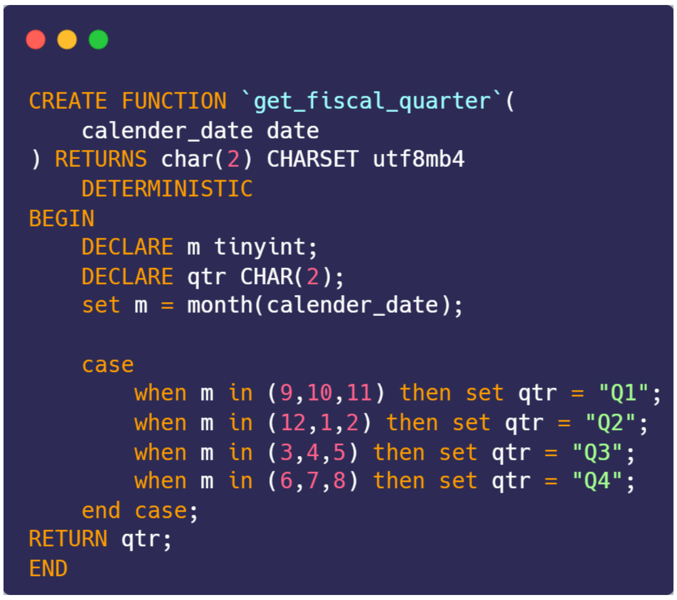
customer_code	customer	platform	channel	market	sub_zone	region
90002002	Croma	Brick & Mortar	Retailer	India	India	APAC
NULL	NULL	NULL	NULL	NULL	NULL	NULL

User Defined Function to get the corresponding fiscal year (AtliQ FY starts from September)



```sql
CREATE 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
```

User Defined Function to get fiscal quarter (AtliQ FY starts from September)

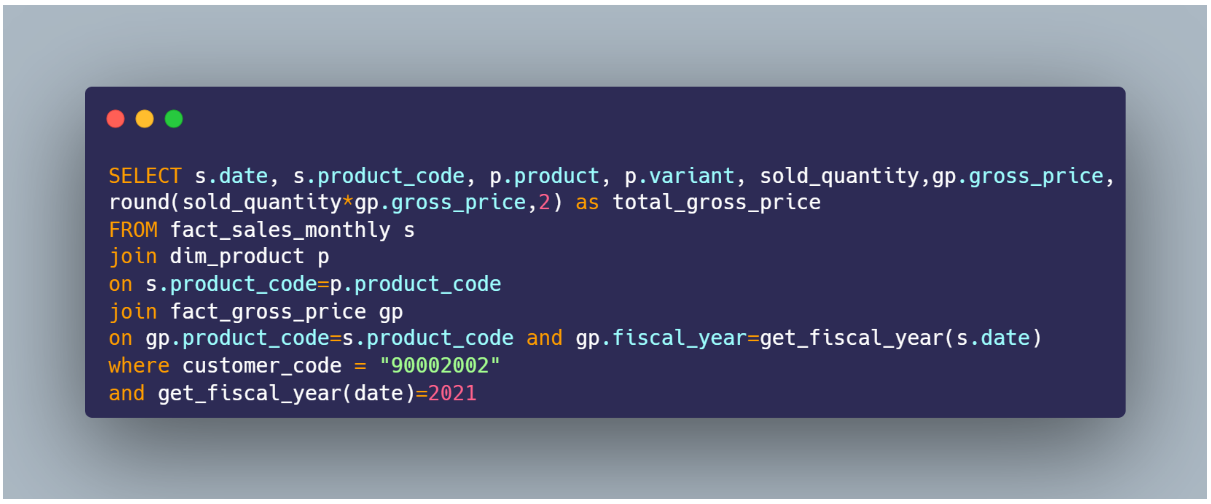


```sql
CREATE FUNCTION `get_fiscal_quarter`(
    calender_date date
) RETURNS char(2) CHARSET utf8mb4
    DETERMINISTIC
BEGIN
    DECLARE m tinyint;
    DECLARE qtr CHAR(2);
    set m = month(calender_date);

    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
```

Document by Shwetha Munigala

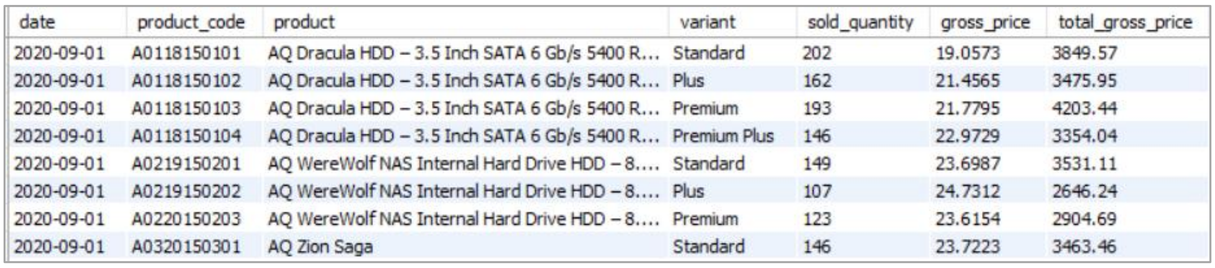
get_fiscal_year and get_fiscal_quarter UDFs were created because these time frames appear in almost every report. By reusing them, we can avoid repetitive date logic, ensuring faster queries and consistent results. UDFs are a go-to for simple logic.

Gross Sales Report: Monthly product level sales for Croma for FY 2021



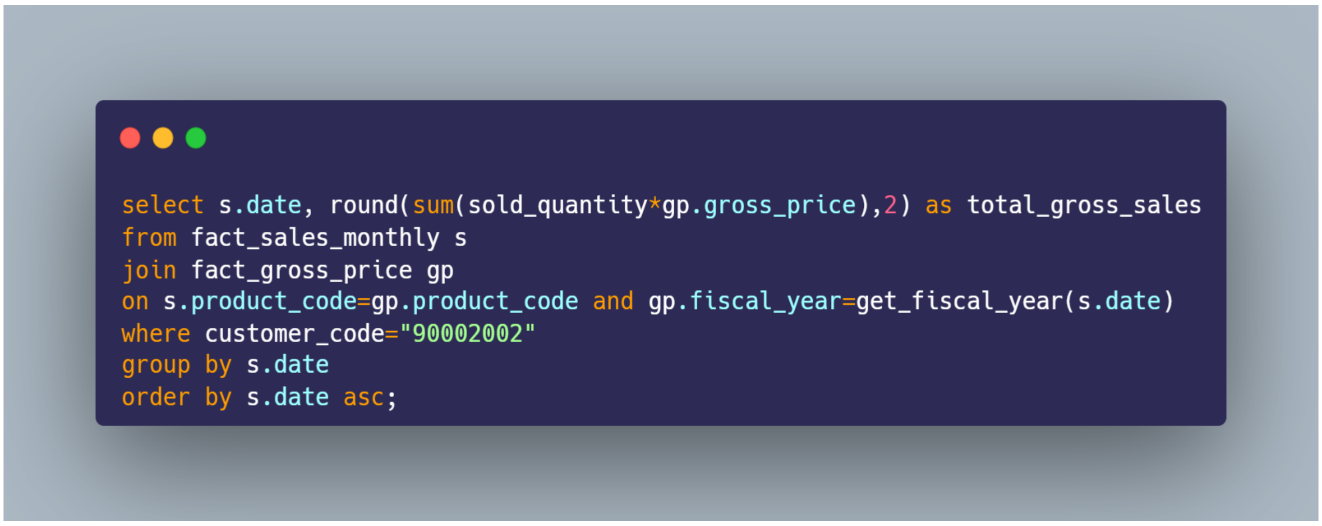
```sql
SELECT s.date, s.product_code, p.product, p.variant, sold_quantity,gp.gross_price,
round(sold_quantity*gp.gross_price,2) as total_gross_price
FROM fact_sales_monthly s
join dim_product p
on s.product_code=p.product_code
join fact_gross_price gp
on gp.product_code=s.product_code and gp.fiscal_year=get_fiscal_year(s.date)
where customer_code = "90002002"
and get_fiscal_year(date)=2021
```

Output:



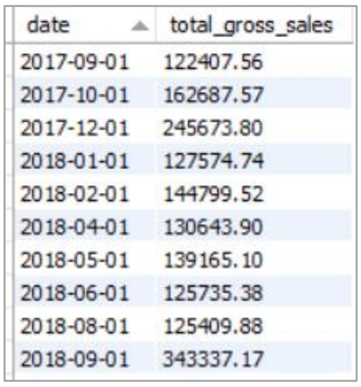
date	product_code	product	variant	sold_quantity	gross_price	total_gross_price
2020-09-01	A0118150101	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Standard	202	19.0573	3849.57
2020-09-01	A0118150102	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Plus	162	21.4565	3475.95
2020-09-01	A0118150103	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Premium	193	21.7795	4203.44
2020-09-01	A0118150104	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Premium Plus	146	22.9729	3354.04
2020-09-01	A0219150201	AQ WereWolf NAS Internal Hard Drive HDD – 8....	Standard	149	23.6987	3531.11
2020-09-01	A0219150202	AQ WereWolf NAS Internal Hard Drive HDD – 8....	Plus	107	24.7312	2646.24
2020-09-01	A0220150203	AQ WereWolf NAS Internal Hard Drive HDD – 8....	Premium	123	23.6154	2904.69
2020-09-01	A0320150301	AQ Zion Saga	Standard	146	23.7223	3463.46

Get total gross sales for 'Croma' per month



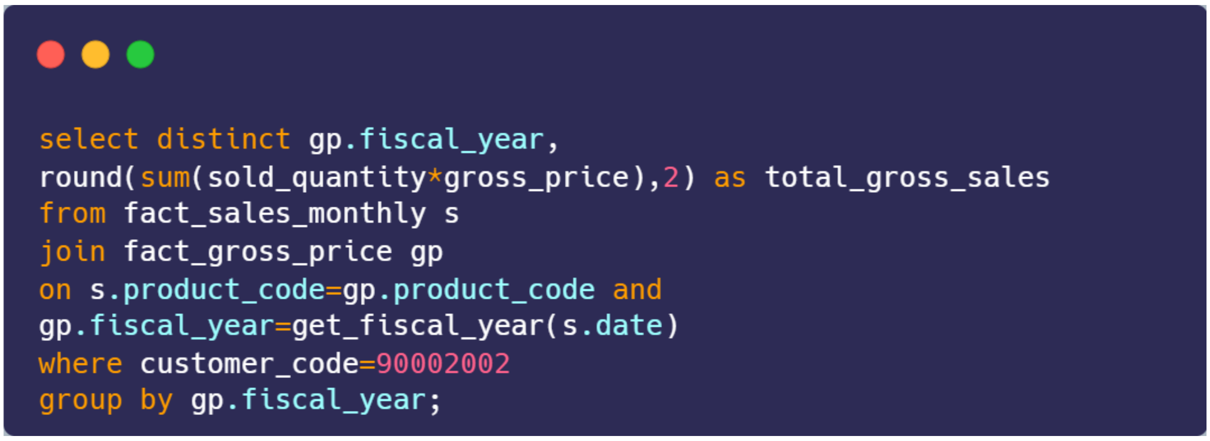
```sql
select s.date, round(sum(sold_quantity*gp.gross_price),2) as total_gross_sales
from fact_sales_monthly s
join fact_gross_price gp
on s.product_code=gp.product_code and gp.fiscal_year=get_fiscal_year(s.date)
where customer_code="90002002"
group by s.date
order by s.date asc;
```

Output:



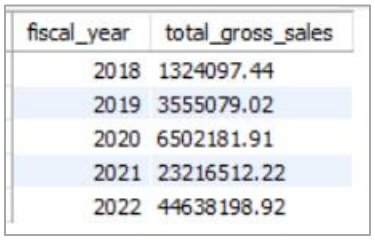
date	total_gross_sales
2017-09-01	122407.56
2017-10-01	162687.57
2017-12-01	245673.80
2018-01-01	127574.74
2018-02-01	144799.52
2018-04-01	130643.90
2018-05-01	139165.10
2018-06-01	125735.38
2018-08-01	125409.88
2018-09-01	343337.17

Generate a yearly sales report for Croma with 2 columns – Fiscal year and Total gross sales in that FY



```
select distinct gp.fiscal_year,
round(sum(sold_quantity*gross_price),2) as total_gross_sales
from fact_sales_monthly s
join fact_gross_price gp
on s.product_code=gp.product_code and
gp.fiscal_year=get_fiscal_year(s.date)
where customer_code=90002002
group by gp.fiscal_year;
```

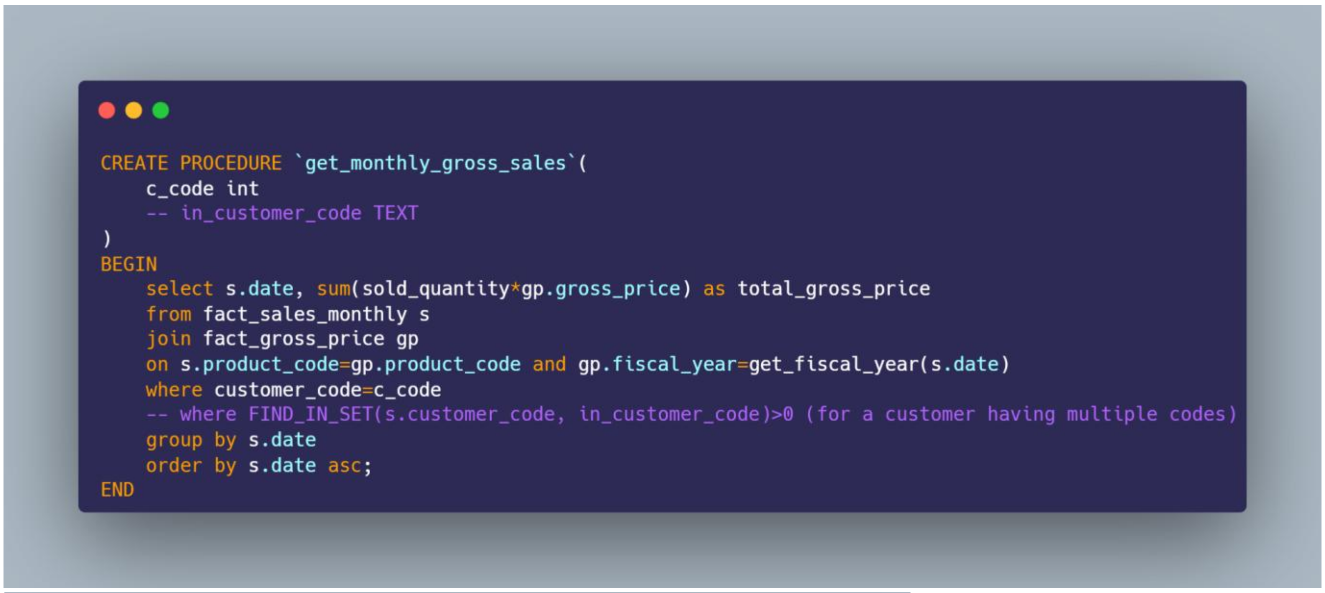
Output:



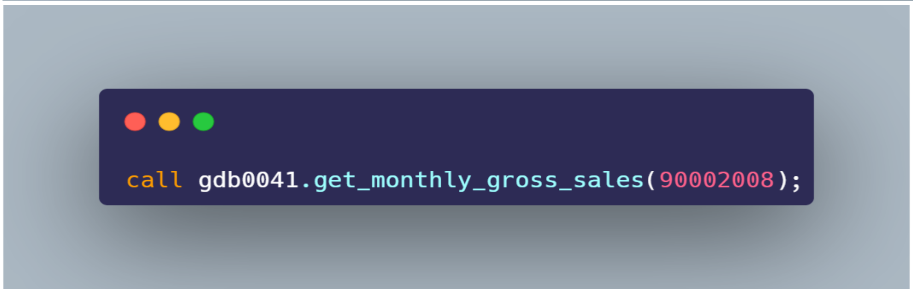
fiscal_year	total_gross_sales
2018	1324097.44
2019	3555079.02
2020	6502181.91
2021	23216512.22
2022	44638198.92

To get the gross sales report for every customer, we can create stored procedures.

Stored procedure to get monthly gross sales report for any customer using the customer code

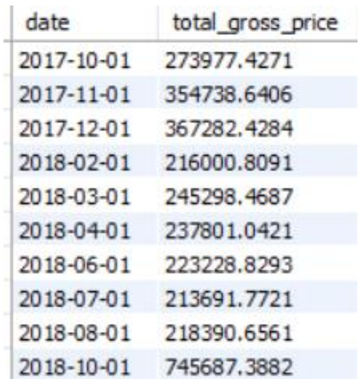


```
CREATE PROCEDURE `get_monthly_gross_sales`(
    c_code int
    -- in_customer_code TEXT
)
BEGIN
    select s.date, sum(sold_quantity*gp.gross_price) as total_gross_price
    from fact_sales_monthly s
    join fact_gross_price gp
    on s.product_code=gp.product_code and gp.fiscal_year=get_fiscal_year(s.date)
    where customer_code=c_code
    -- where FIND_IN_SET(s.customer_code, in_customer_code)>0 (for a customer having multiple codes)
    group by s.date
    order by s.date asc;
END
```



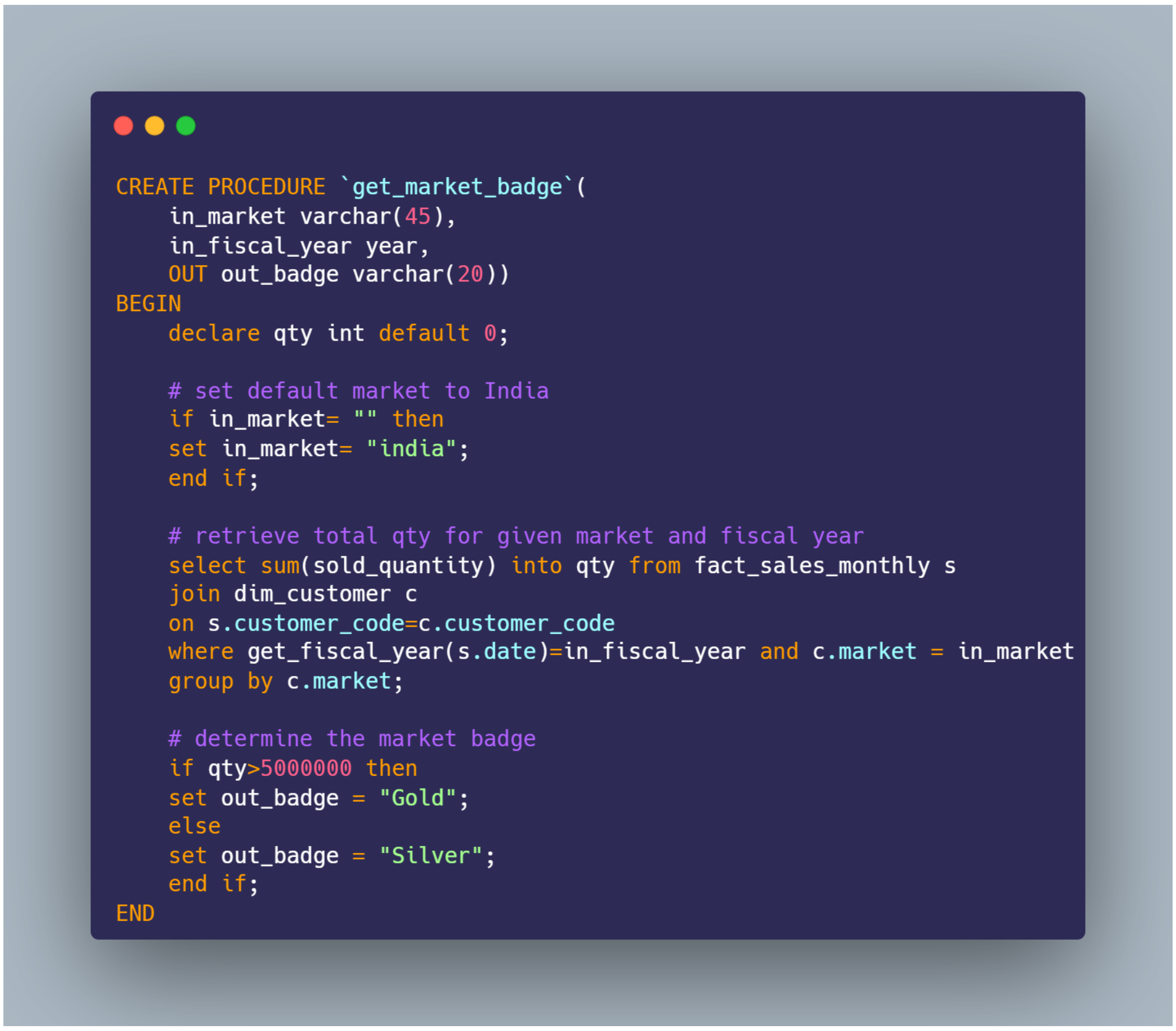
```
call gdb0041.get_monthly_gross_sales(90002008);
```

Output:



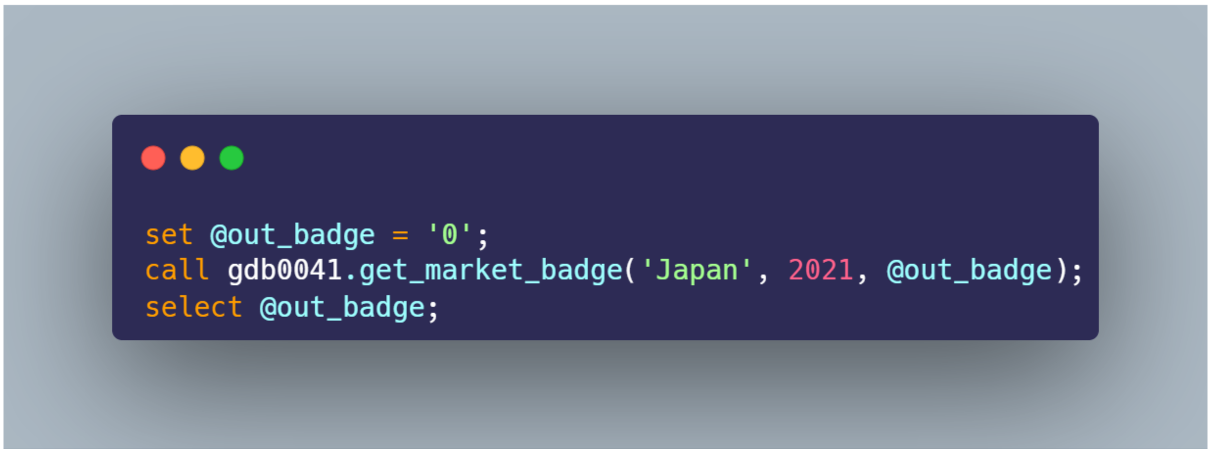
date	total_gross_price
2017-10-01	273977.4271
2017-11-01	354738.6406
2017-12-01	367282.4284
2018-02-01	216000.8091
2018-03-01	245298.4687
2018-04-01	237801.0421
2018-06-01	223228.8293
2018-07-01	213691.7721
2018-08-01	218390.6561
2018-10-01	745687.3882

Stored procedure to assign market badges to customers based on their sales turnover



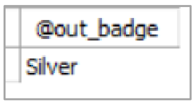
```sql
CREATE PROCEDURE `get_market_badge`(
    in_market varchar(45),
    in_fiscal_year year,
    OUT out_badge varchar(20))
BEGIN
    declare qty int default 0;

    # set default market to India
    if in_market= "" then
    set in_market= "india";
    end if;

    # retrieve total qty for given market and fiscal year
    select sum(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
    group by c.market;

    # determine the market badge
    if qty>5000000 then
    set out_badge = "Gold";
    else
    set out_badge = "Silver";
    end if;
END
```

When we call the store procedure, a window is popped up asking for the values ‘market’, ‘fiscal year’. Once we enter the values and click execute,



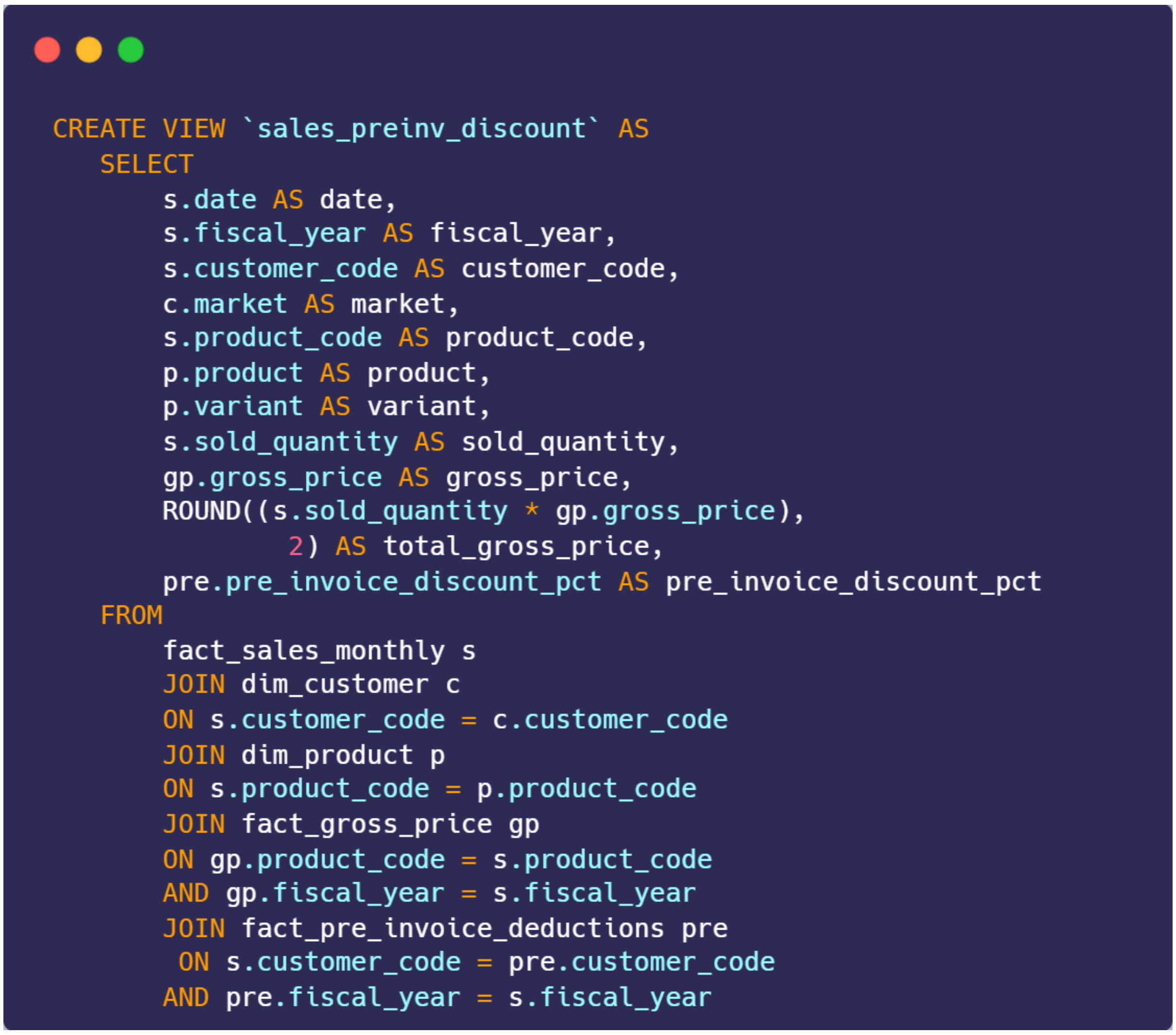
```sql
set @out_badge = '0';
call gdb0041.get_market_badge('Japan', 2021, @out_badge);
select @out_badge;
```

We get the output as –



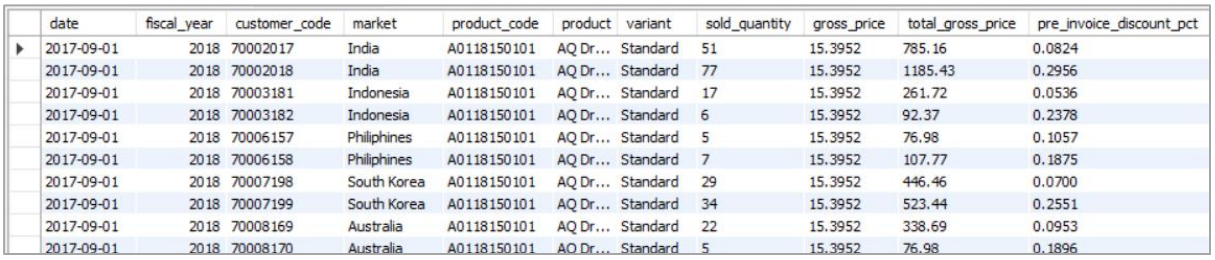
@out_badge
Silver

Create a view named 'sales_preinv_discount' to generate reports easily for future use.



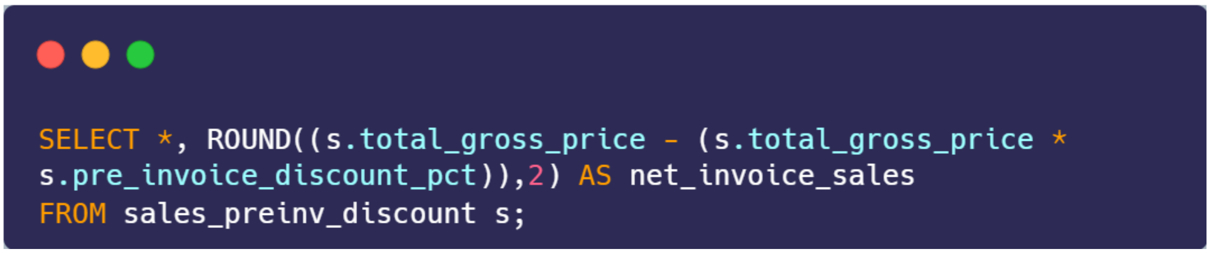
```sql
CREATE VIEW `sales_preinv_discount` AS
   SELECT
       s.date AS date,
       s.fiscal_year AS fiscal_year,
       s.customer_code AS customer_code,
       c.market AS market,
       s.product_code AS product_code,
       p.product AS product,
       p.variant AS variant,
       s.sold_quantity AS sold_quantity,
       gp.gross_price AS gross_price,
       ROUND((s.sold_quantity * gp.gross_price),
               2) AS total_gross_price,
       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 gp
       ON gp.product_code = s.product_code
       AND gp.fiscal_year = s.fiscal_year
       JOIN fact_pre_invoice_deductions pre
        ON s.customer_code = pre.customer_code
       AND pre.fiscal_year = s.fiscal_year
```

Output:



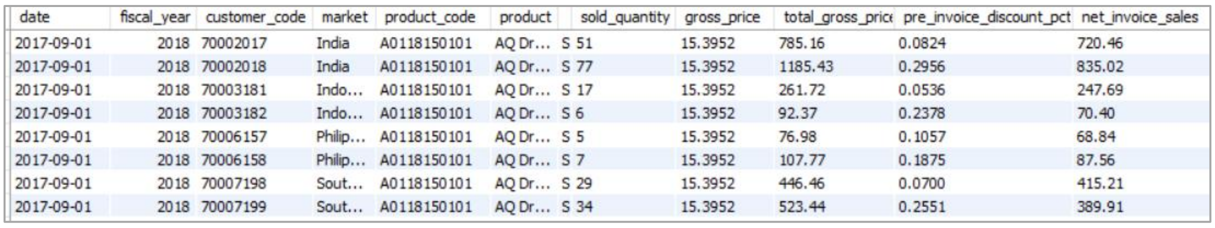
date	fiscal_year	customer_code	market	product_code	product	variant	sold_quantity	gross_price	total_gross_price	pre_invoice_discount_pct
2017-09-01	2018	70002017	India	A0118150101	AQ Dr...	Standard	51	15.3952	785.16	0.0824
2017-09-01	2018	70002018	India	A0118150101	AQ Dr...	Standard	77	15.3952	1185.43	0.2956
2017-09-01	2018	70003181	Indonesia	A0118150101	AQ Dr...	Standard	17	15.3952	261.72	0.0536
2017-09-01	2018	70003182	Indonesia	A0118150101	AQ Dr...	Standard	6	15.3952	92.37	0.2378
2017-09-01	2018	70006157	Philiphines	A0118150101	AQ Dr...	Standard	5	15.3952	76.98	0.1057
2017-09-01	2018	70006158	Philiphines	A0118150101	AQ Dr...	Standard	7	15.3952	107.77	0.1875
2017-09-01	2018	70007198	South Korea	A0118150101	AQ Dr...	Standard	29	15.3952	446.46	0.0700
2017-09-01	2018	70007199	South Korea	A0118150101	AQ Dr...	Standard	34	15.3952	523.44	0.2551
2017-09-01	2018	70008169	Australia	A0118150101	AQ Dr...	Standard	22	15.3952	338.69	0.0953
2017-09-01	2018	70008170	Australia	A0118150101	AQ Dr...	Standard	5	15.3952	76.98	0.1896

By using this view, we generate net invoice sales data –



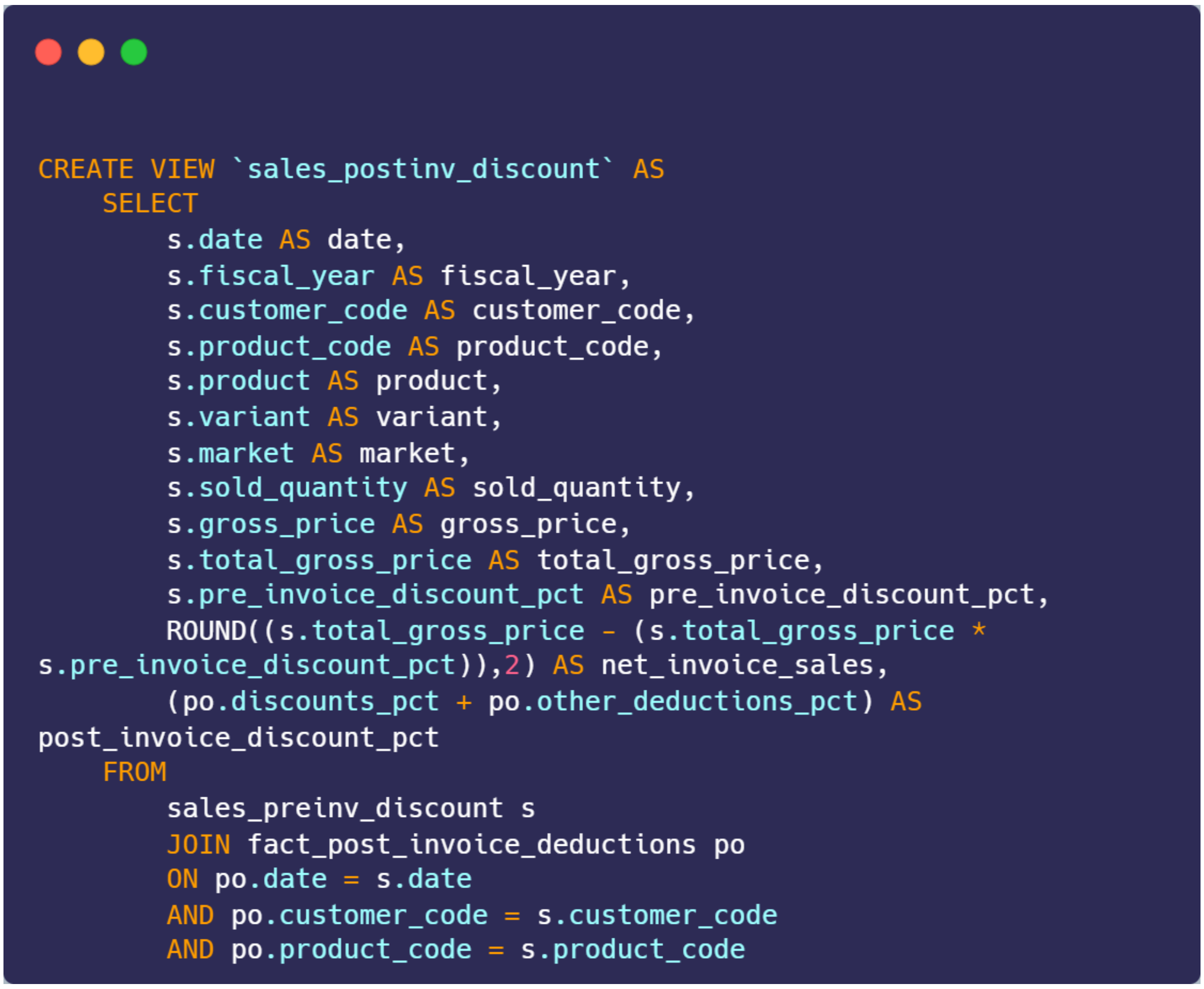
```sql
SELECT *, ROUND((s.total_gross_price - (s.total_gross_price *
s.pre_invoice_discount_pct)),2) AS net_invoice_sales
FROM sales_preinv_discount s;
```

Output:



date	fiscal_year	customer_code	market	product_code	product		sold_quantity	gross_price	total_gross_pric	pre_invoice_discount_pct	net_invoice_sales
2017-09-01	2018	70002017	India	A0118150101	AQ Dr...	S	51	15.3952	785.16	0.0824	720.46
2017-09-01	2018	70002018	India	A0118150101	AQ Dr...	S	77	15.3952	1185.43	0.2956	835.02
2017-09-01	2018	70003181	Indo...	A0118150101	AQ Dr...	S	17	15.3952	261.72	0.0536	247.69
2017-09-01	2018	70003182	Indo...	A0118150101	AQ Dr...	S	6	15.3952	92.37	0.2378	70.40
2017-09-01	2018	70006157	Philip...	A0118150101	AQ Dr...	S	5	15.3952	76.98	0.1057	68.84
2017-09-01	2018	70006158	Philip...	A0118150101	AQ Dr...	S	7	15.3952	107.77	0.1875	87.56
2017-09-01	2018	70007198	Sout...	A0118150101	AQ Dr...	S	29	15.3952	446.46	0.0700	415.21
2017-09-01	2018	70007199	Sout...	A0118150101	AQ Dr...	S	34	15.3952	523.44	0.2551	389.91

Create a view named `sales_postinv_discount` for generating report easily for future use.

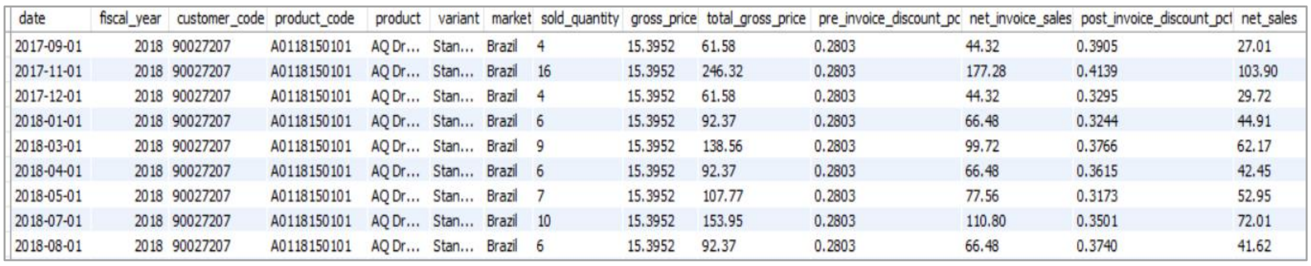


```sql
CREATE VIEW `sales_postinv_discount` AS
    SELECT
        s.date AS date,
        s.fiscal_year AS fiscal_year,
        s.customer_code AS customer_code,
        s.product_code AS product_code,
        s.product AS product,
        s.variant AS variant,
        s.market AS market,
        s.sold_quantity AS sold_quantity,
        s.gross_price AS gross_price,
        s.total_gross_price AS total_gross_price,
        s.pre_invoice_discount_pct AS pre_invoice_discount_pct,
        ROUND((s.total_gross_price - (s.total_gross_price *
s.pre_invoice_discount_pct)),2) 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.date = s.date
        AND po.customer_code = s.customer_code
        AND po.product_code = s.product_code
```

Now, using this view we calculate ‘Net Sales’

```
SELECT *, ROUND(((1-post_invoice_discount_pct)*net_invoice_sales),2)
AS net_sales
FROM sales_postinv_discount;
```

Output:



date	fiscal_year	customer_code	product_code	product	variant	market	sold_quantity	gross_price	total_gross_price	pre_invoice_discount_pc	net_invoice_sales	post_invoice_discount_pct	net_sales
2017-09-01	2018	90027207	A0118150101	AQ Dr...	Stan...	Brazil	4	15.3952	61.58	0.2803	44.32	0.3905	27.01
2017-11-01	2018	90027207	A0118150101	AQ Dr...	Stan...	Brazil	16	15.3952	246.32	0.2803	177.28	0.4139	103.90
2017-12-01	2018	90027207	A0118150101	AQ Dr...	Stan...	Brazil	4	15.3952	61.58	0.2803	44.32	0.3295	29.72
2018-01-01	2018	90027207	A0118150101	AQ Dr...	Stan...	Brazil	6	15.3952	92.37	0.2803	66.48	0.3244	44.91
2018-03-01	2018	90027207	A0118150101	AQ Dr...	Stan...	Brazil	9	15.3952	138.56	0.2803	99.72	0.3766	62.17
2018-04-01	2018	90027207	A0118150101	AQ Dr...	Stan...	Brazil	6	15.3952	92.37	0.2803	66.48	0.3615	42.45
2018-05-01	2018	90027207	A0118150101	AQ Dr...	Stan...	Brazil	7	15.3952	107.77	0.2803	77.56	0.3173	52.95
2018-07-01	2018	90027207	A0118150101	AQ Dr...	Stan...	Brazil	10	15.3952	153.95	0.2803	110.80	0.3501	72.01
2018-08-01	2018	90027207	A0118150101	AQ Dr...	Stan...	Brazil	6	15.3952	92.37	0.2803	66.48	0.3740	41.62

For quick report generation, we can create a view for ‘net sales’

```
CREATE VIEW `net_sales` AS
SELECT *, ROUND(((1-post_invoice_discount_pct)*net_invoice_sales),2)
AS net_sales
FROM sales_postinv_discount;
```

Get the customer yearly sales

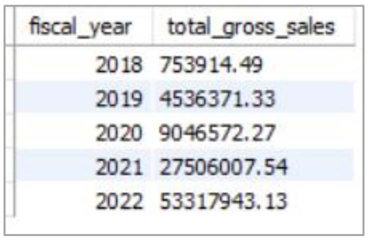
Here, we are using stored procedure -

```
CREATE PROCEDURE `customer_yearly_sales`(
    c_code int)
BEGIN
    select distinct gp.fiscal_year,
    round(sum(sold_quantity*gross_price),2) as total_gross_sales
    from fact_sales_monthly s
    join fact_gross_price gp
    on s.product_code=gp.product_code and
gp.fiscal_year=get_fiscal_year(s.date)
    where customer_code=c_code
    group by gp.fiscal_year;
END
```

Now, when we call the procedure, a small pop-up window appears asking for the input parameter values. Once you enter the input value and click on execute, we see the output.

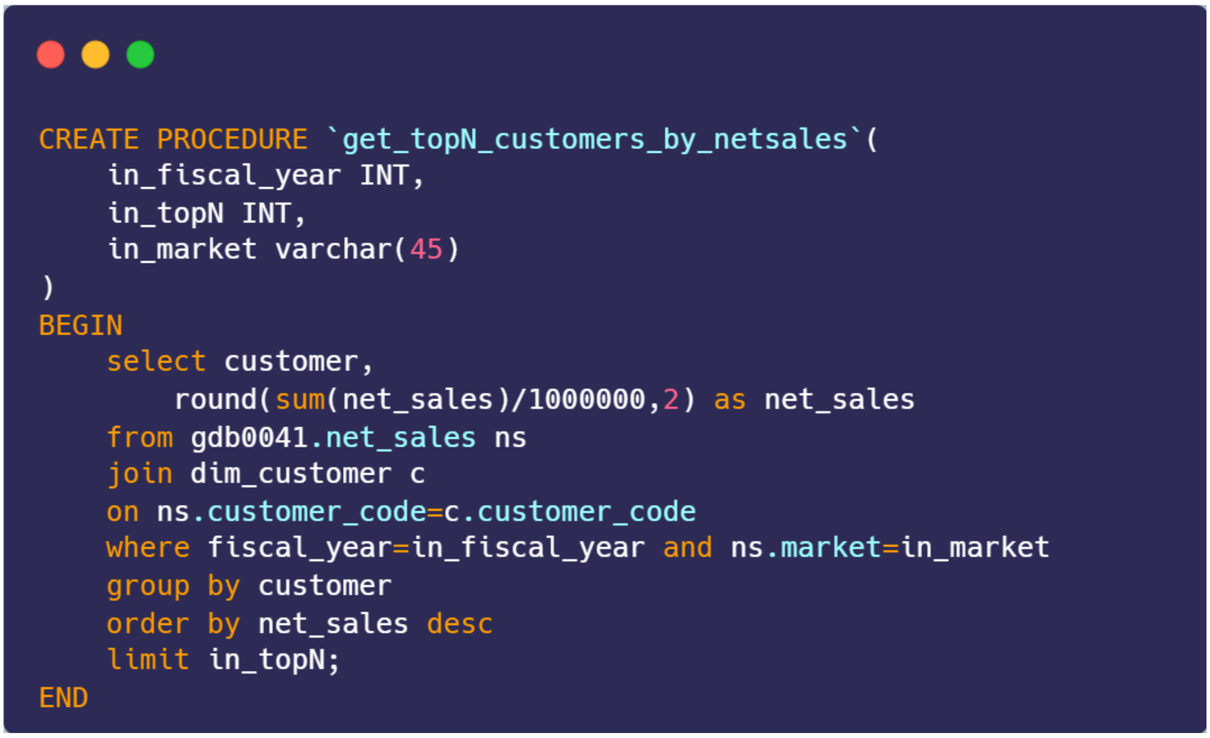
```
call gdb0041.customer_yearly_sales(80006155);
```

Output:



fiscal_year	total_gross_sales
2018	753914.49
2019	4536371.33
2020	9046572.27
2021	27506007.54
2022	53317943.13

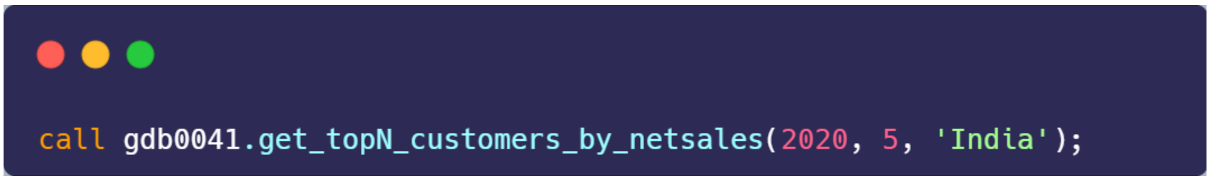
Find out Top N customers by net sales



```sql
CREATE PROCEDURE `get_topN_customers_by_netsales`(
    in_fiscal_year INT,
    in_topN INT,
    in_market varchar(45)
)
BEGIN
    select customer,
        round(sum(net_sales)/1000000,2) as net_sales
    from gdb0041.net_sales ns
    join dim_customer c
    on ns.customer_code=c.customer_code
    where fiscal_year=in_fiscal_year and ns.market=in_market
    group by customer
    order by net_sales desc
    limit in_topN;
END
```

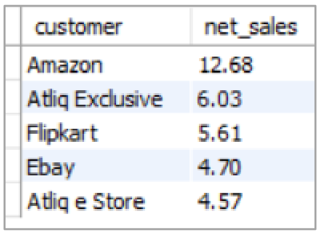
Now, when we call the procedure, a small pop-up window appears asking for the input parameter values.

FY – 2020, Top N customers – 5, Market - India



```sql
call gdb0041.get_topN_customers_by_netsales(2020, 5, 'India');
```

Output:



customer	net_sales
Amazon	12.68
Atliq Exclusive	6.03
Flipkart	5.61
Ebay	4.70
Atliq e Store	4.57

Note: net sales values in millions

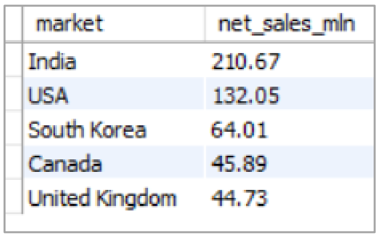
Find out Top N markets by net sales

```
CREATE PROCEDURE `get_topN_markets_by_netsales`(
    in_fiscal_year INT,
    in_topN INT
)
BEGIN
    select market,
        round(sum(net_sales)/1000000,2) as net_sales_mln
    from gdb0041.net_sales
    where fiscal_year=in_fiscal_year
    group by market
    order by net_sales_mln desc
    limit in_topN;
END
```

Now, when we call the procedure, a small pop-up window appears asking for the input parameter values. As we want to find the top 5 markets, let us find out for FY = 2021.

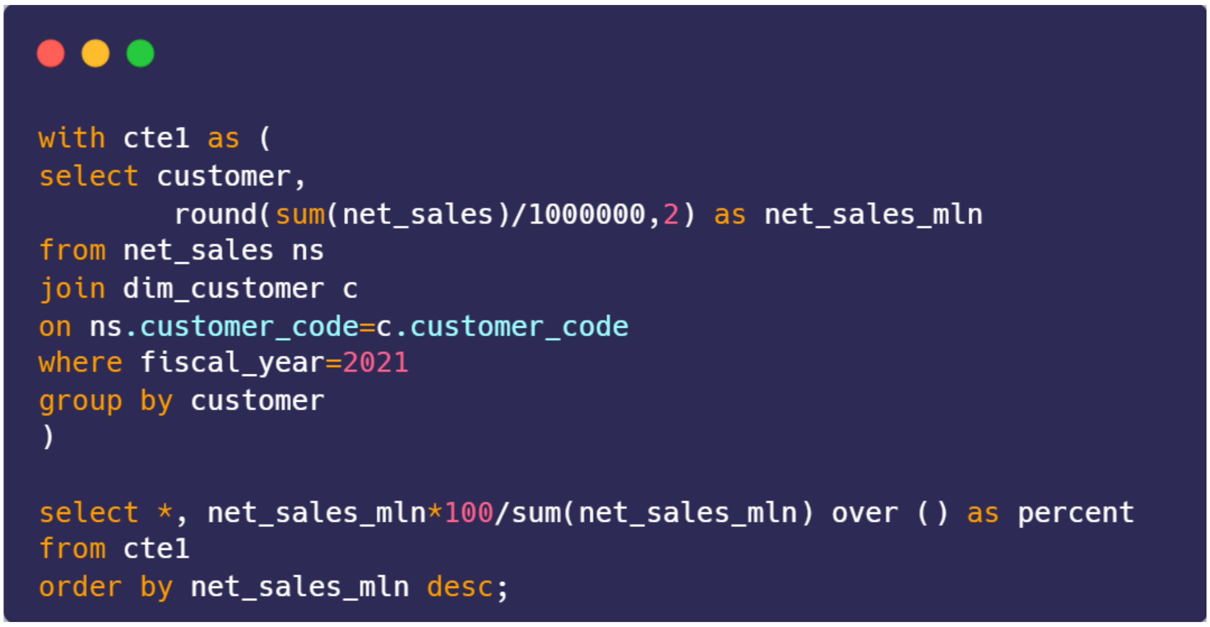
```
call gdb0041.get_topN_markets_by_netsales(2021, 5);
```

Output:



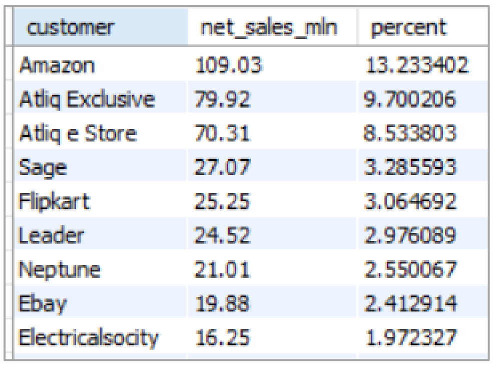
market	net_sales_mln
India	210.67
USA	132.05
South Korea	64.01
Canada	45.89
United Kingdom	44.73

Get the net sales contribution per customer and plot a chart for top 10 customers.

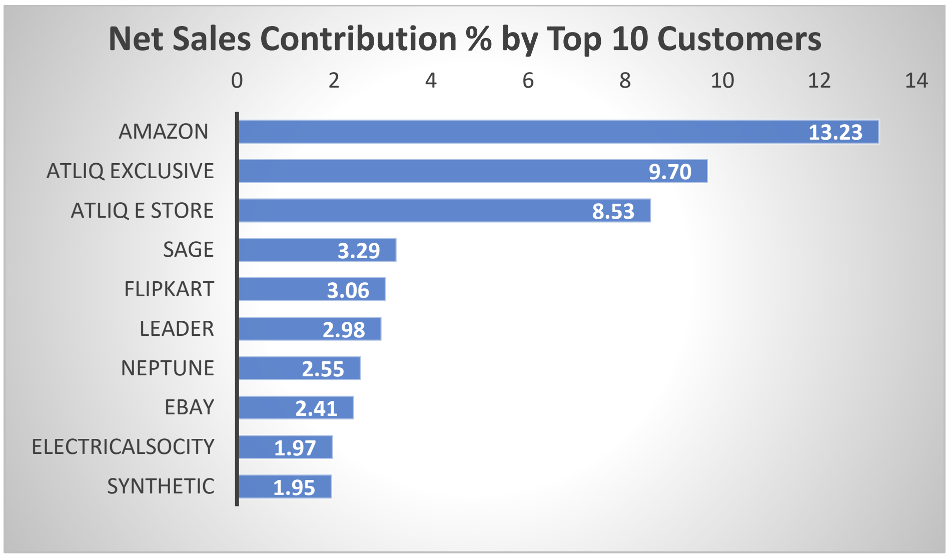


```
with cte1 as (
select customer,
        round(sum(net_sales)/1000000,2) as net_sales_mln
from net_sales ns
join dim_customer c
on ns.customer_code=c.customer_code
where fiscal_year=2021
group by customer
)

select *, net_sales_mln*100/sum(net_sales_mln) over () as percent
from cte1
order by net_sales_mln desc;
```

Output:

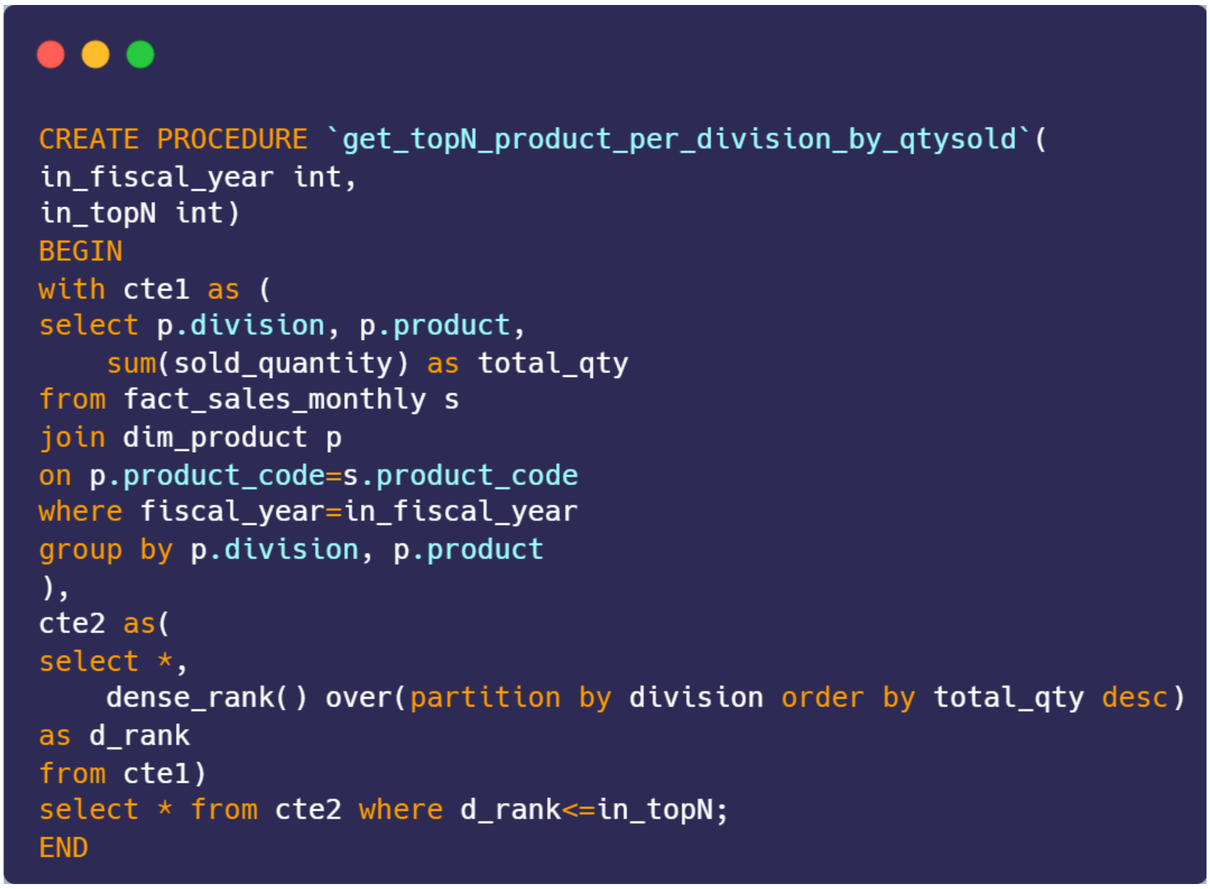


customer	net_sales_mln	percent
Amazon	109.03	13.233402
Atliq Exclusive	79.92	9.700206
Atliq e Store	70.31	8.533803
Sage	27.07	3.285593
Flipkart	25.25	3.064692
Leader	24.52	2.976089
Neptune	21.01	2.550067
Ebay	19.88	2.412914
Electricalsocity	16.25	1.972327

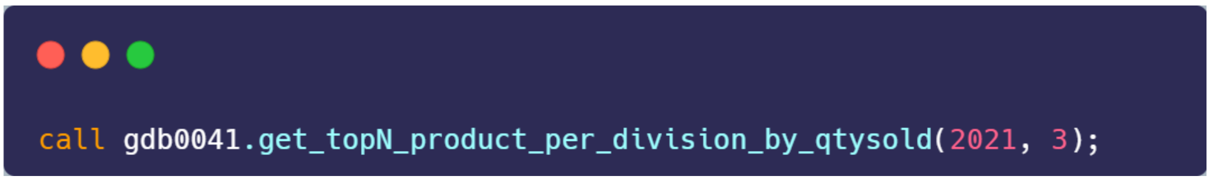


Document by Shwetha Munigala

Write a stored procedure for Top N products in each division by their sold quantity in a given financial year

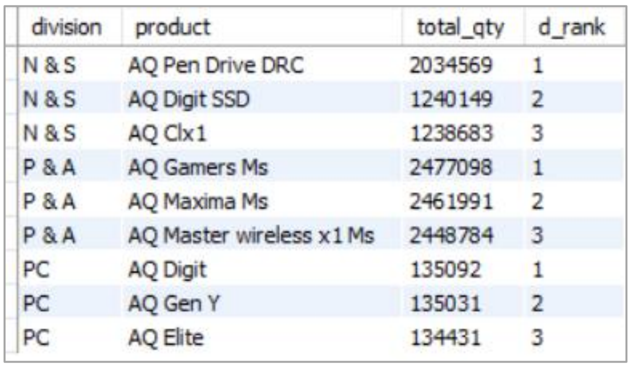


```
CREATE PROCEDURE `get_topN_product_per_division_by_qtysold`(
in_fiscal_year int,
in_topN int)
BEGIN
with cte1 as (
select p.division, p.product,
    sum(sold_quantity) as total_qty
from fact_sales_monthly s
join dim_product p
on p.product_code=s.product_code
where fiscal_year=in_fiscal_year
group by p.division, p.product
),
cte2 as(
select *,
    dense_rank() over(partition by division order by total_qty desc)
as d_rank
from cte1)
select * from cte2 where d_rank<=in_topN;
END
```



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

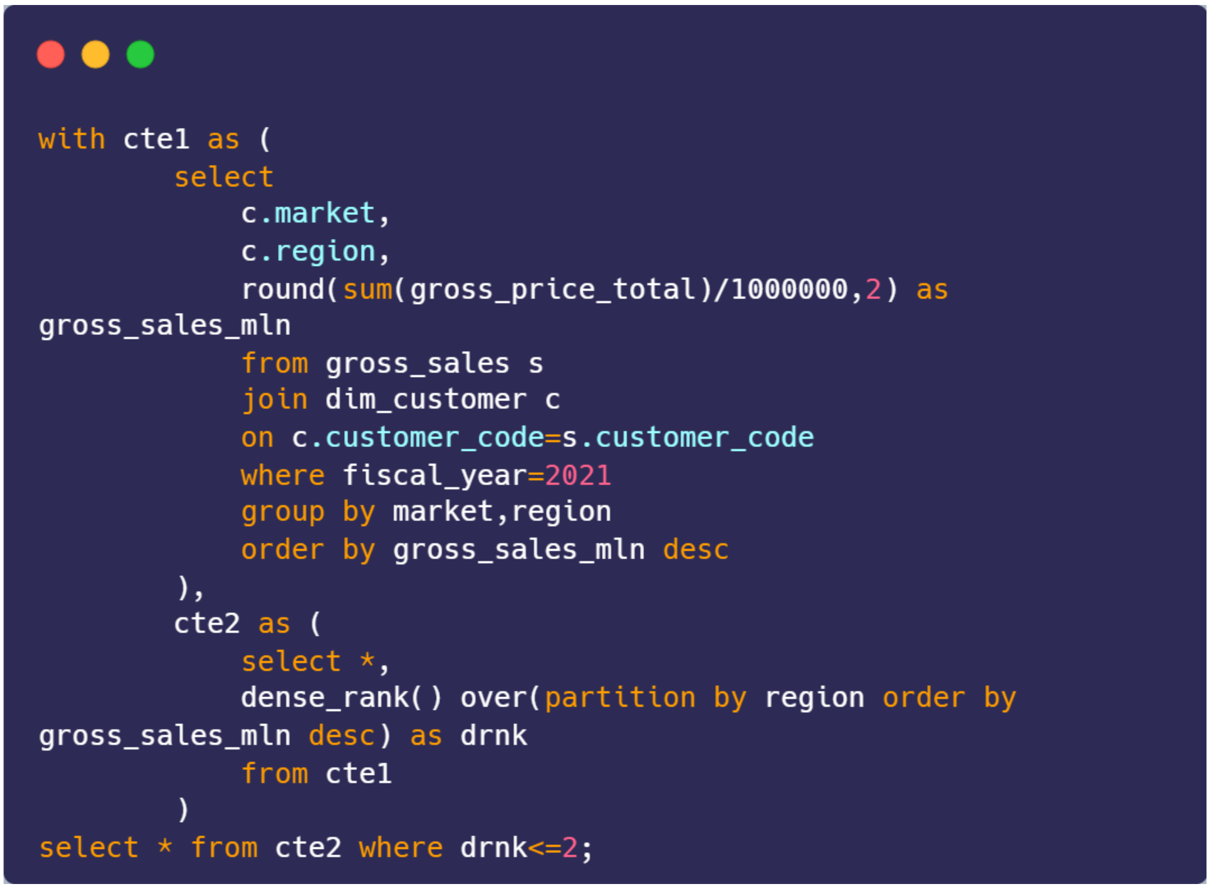
Output:



division	product	total_qty	d_rank
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

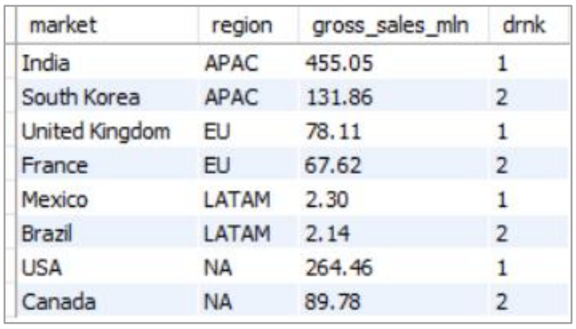
Document by Shwetha Munigala

Retrieve the top 2 markets in every region by their gross sales amount for FY:2021



```
with cte1 as (
        select
            c.market,
            c.region,
            round(sum(gross_price_total)/1000000,2) as
gross_sales_mln
            from gross_sales s
            join dim_customer c
            on c.customer_code=s.customer_code
            where fiscal_year=2021
            group by market,region
            order by gross_sales_mln desc
        ),
        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;
```

Output:



market	region	gross_sales_mln	drnk
India	APAC	455.05	1
South Korea	APAC	131.86	2
United Kingdom	EU	78.11	1
France	EU	67.62	2
Mexico	LATAM	2.30	1
Brazil	LATAM	2.14	2
USA	NA	264.46	1
Canada	NA	89.78	2