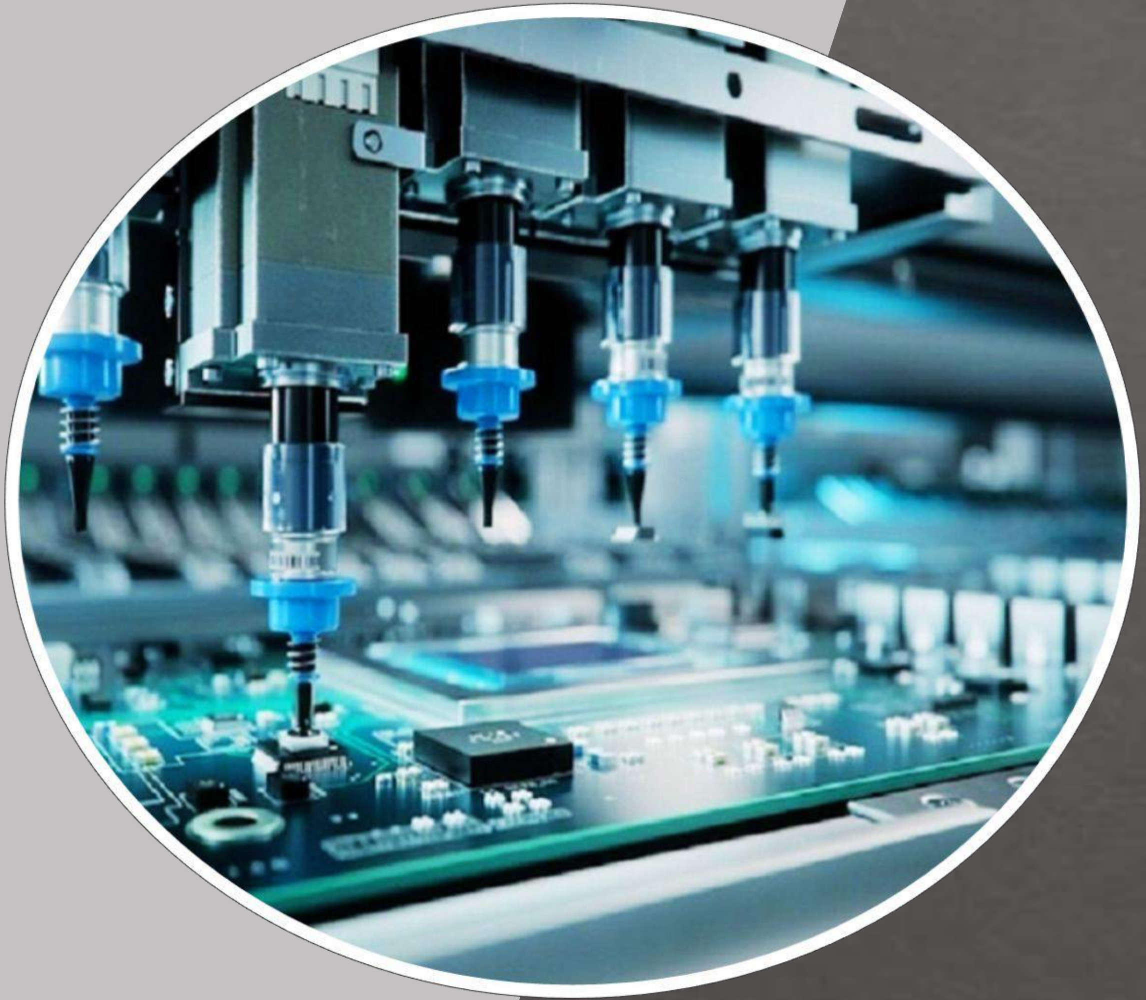


AtliQ Hardware

INSIGHTS FROM AD HOC REQUESTS

Content

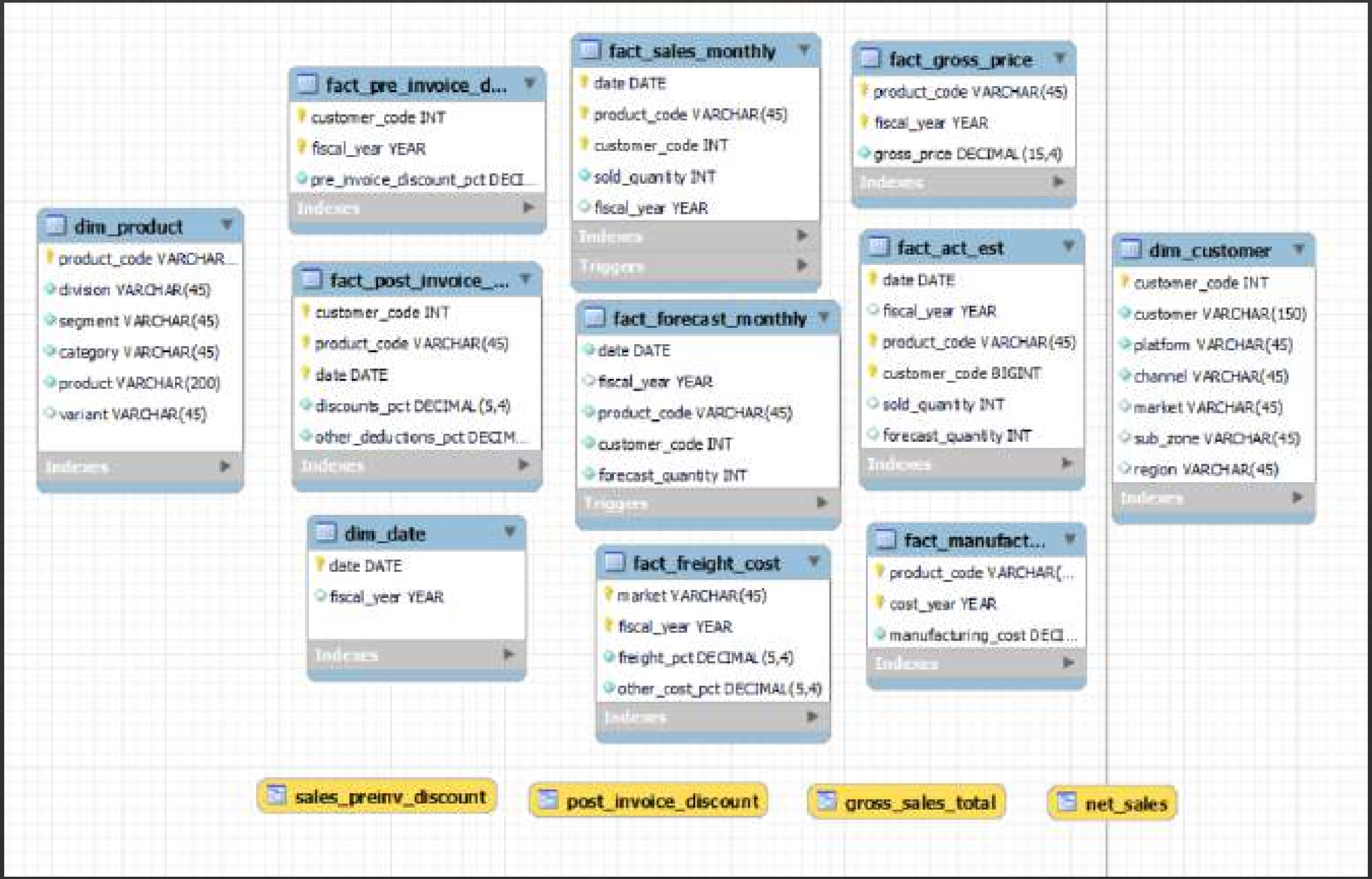
Company Overview

- AtliQ manufactures and sales hardware.
- To customers like Croma, Best Buy, Flipkart, Amazon etc.
- Manufacture → Warehouse → Distribution Centres → Customers → Consumers
- Customers are Brick and mortar and Ecommerce types
- Retailers, Distributors and Direct sales are channels

Problem Statement

- To answer and generate reports for Ad Hoc requests given by Product owner

Dataset and model



dim_product
- product_code VARCHAR...
- division VARCHAR(45)
- segment VARCHAR(45)
- category VARCHAR(45)
- product VARCHAR(200)
- variant VARCHAR(45)

Indexes

fact_pre_invoice_d...
- customer_code INT
- fiscal_year YEAR
- pre_invoice_discount_pct DECI...

Indexes

fact_post_invoice_...
- customer_code INT
- product_code VARCHAR(45)
- date DATE
- discounts_pct DECIMAL(5,4)
- other_deductions_pct DECIM...

Indexes

dim_date
- date DATE
- fiscal_year YEAR

Indexes

fact_sales_monthly
- date DATE
- product_code VARCHAR(45)
- customer_code INT
- sold_quantity INT
- fiscal_year YEAR

Indexes

Triggers

fact_forecast_monthly
- date DATE
- fiscal_year YEAR
- product_code VARCHAR(45)
- customer_code INT
- forecast_quantity INT

Triggers

fact_freight_cost
- market VARCHAR(45)
- fiscal_year YEAR
- freight_pct DECIMAL(5,4)
- other_cost_pct DECIMAL(5,4)

Indexes

fact_gross_price
- product_code VARCHAR(45)
- fiscal_year YEAR
- gross_price DECIMAL(15,4)

Indexes

fact_act_est
- date DATE
- fiscal_year YEAR
- product_code VARCHAR(45)
- customer_code BIGINT
- sold_quantity INT
- forecast_quantity INT

Indexes

fact_manufact...
- product_code VARCHAR(...
- cost_year YEAR
- manufacturing_cost DECI...

Indexes

dim_customer
- customer_code INT
- customer VARCHAR(150)
- platform VARCHAR(45)
- channel VARCHAR(45)
- market VARCHAR(45)
- sub_zone VARCHAR(45)
- region VARCHAR(45)

Indexes

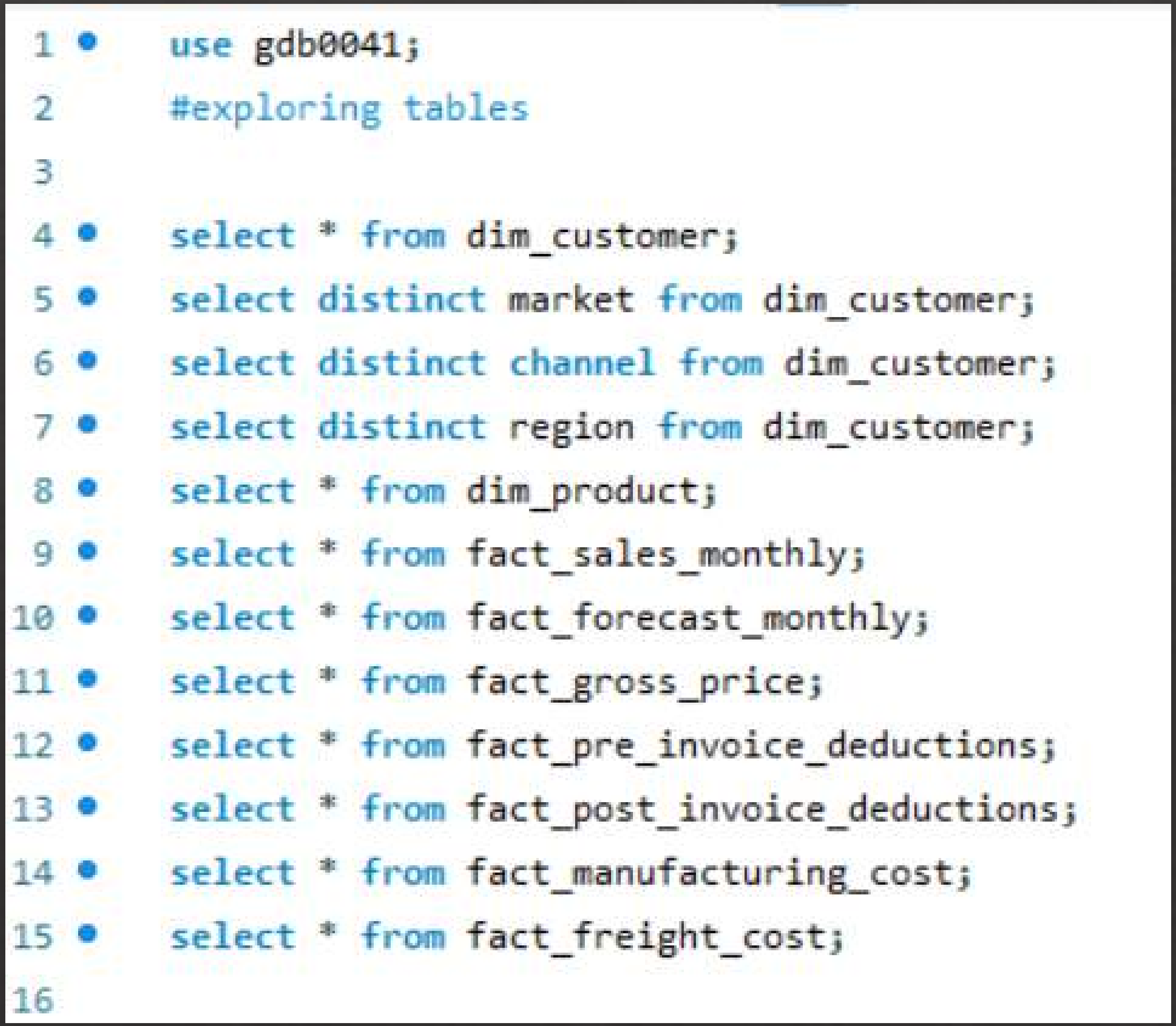
sales_preinv_discount

post_invoice_discount

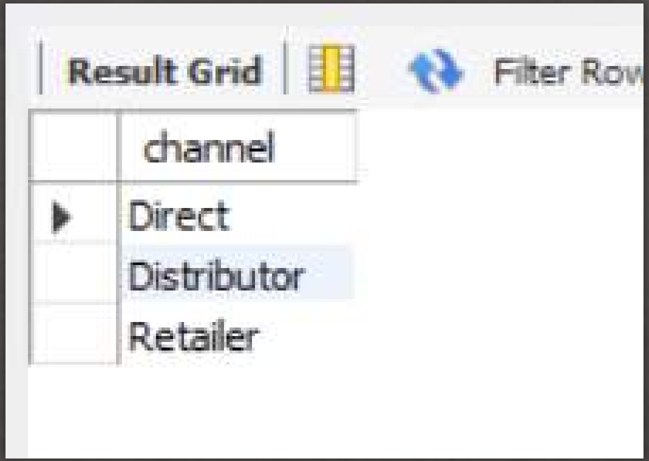
gross_sales_total

net_sales

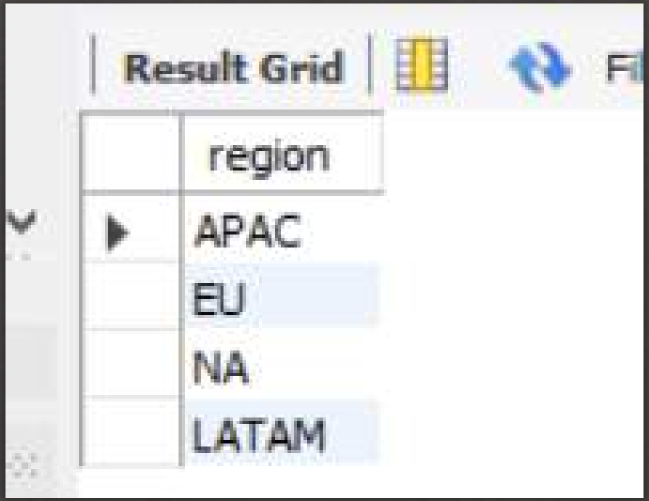
Exploring Data



```sql
use gdb0041;
#exploring tables

select * from dim_customer;
select distinct market from dim_customer;
select distinct channel from dim_customer;
select distinct region from dim_customer;
select * from dim_product;
select * from fact_sales_monthly;
select * from fact_forecast_monthly;
select * from fact_gross_price;
select * from fact_pre_invoice_deductions;
select * from fact_post_invoice_deductions;
select * from fact_manufacturing_cost;
select * from fact_freight_cost;
```



channel
Direct
Distributor
Retailer



region
APAC
EU
NA
LATAM

Ad-hoc requests, output and insights

1. Gross sales report

Product wise sales report for the customer Croma India for fiscal year 2021

Details: Generate report of individual product sales (aggregated on monthly basis at product code level for Croma India customer for FY=2021 to track individual product sales and run further analysis in excel. The report should have following fields

-- Month

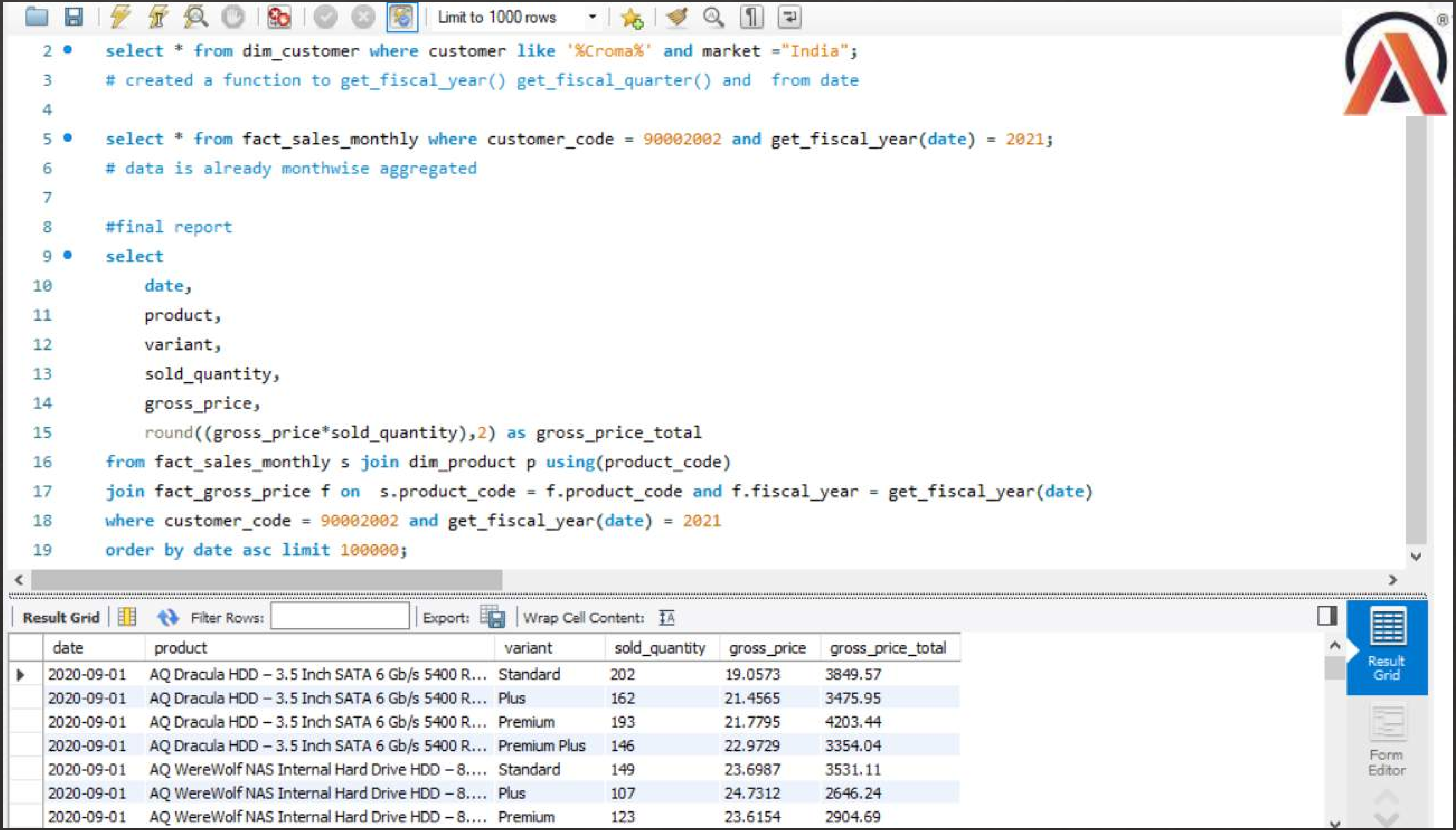
-- Product name

-- Variant

-- Sold quantity

-- Gross price per item

-- Gross price total



```sql
select * from dim_customer where customer like '%Croma%' and market ="India";
# created a function to get_fiscal_year() get_fiscal_quarter() and  from date

select * from fact_sales_monthly where customer_code = 90002002 and get_fiscal_year(date) = 2021;
# data is already monthwise aggregated

#final report
select
    date,
    product,
    variant,
    sold_quantity,
    gross_price,
    round((gross_price*sold_quantity),2) as gross_price_total
from fact_sales_monthly s join dim_product p using(product_code)
join fact_gross_price f on  s.product_code = f.product_code and f.fiscal_year = get_fiscal_year(date)
where customer_code = 90002002 and get_fiscal_year(date) = 2021
order by date asc limit 100000;
```

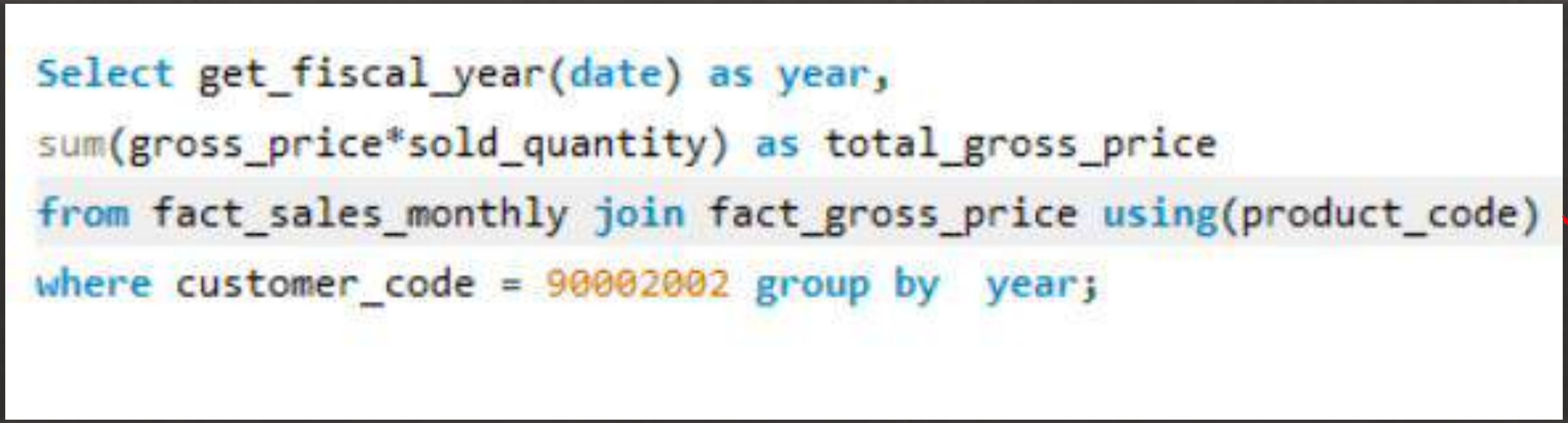
date	product	variant	sold_quantity	gross_price	gross_price_total
2020-09-01	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Standard	202	19.0573	3849.57
2020-09-01	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Plus	162	21.4565	3475.95
2020-09-01	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Premium	193	21.7795	4203.44
2020-09-01	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Premium Plus	146	22.9729	3354.04
2020-09-01	AQ WereWolf NAS Internal Hard Drive HDD – 8....	Standard	149	23.6987	3531.11
2020-09-01	AQ WereWolf NAS Internal Hard Drive HDD – 8....	Plus	107	24.7312	2646.24
2020-09-01	AQ WereWolf NAS Internal Hard Drive HDD – 8....	Premium	123	23.6154	2904.69

2. Yearly sales report

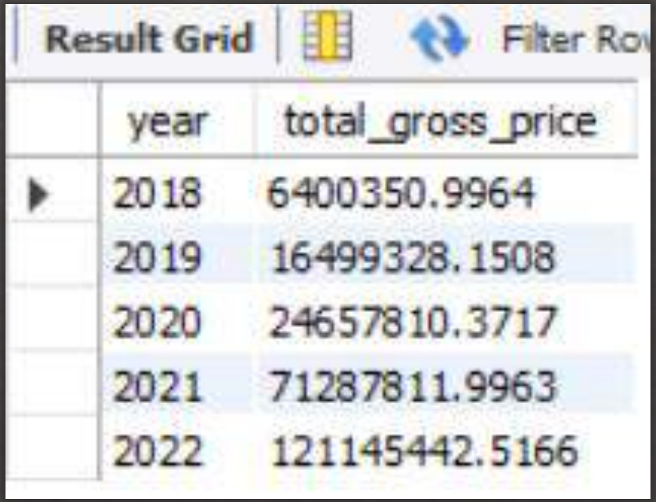
Generate a yearly report for Croma India with columns

-- Fiscal year

-- Total gross sales amount in that year from Croma



```
Select get_fiscal_year(date) as year,
sum(gross_price*sold_quantity) as total_gross_price
from fact_sales_monthly join fact_gross_price using(product_code)
where customer_code = 90002002 group by  year;
```



```
year   total_gross_price
2018   6400350.9964
2019   16499328.1508
2020   24657810.3717
2021   71287811.9963
2022   121145442.5166
```

3. Market Badge

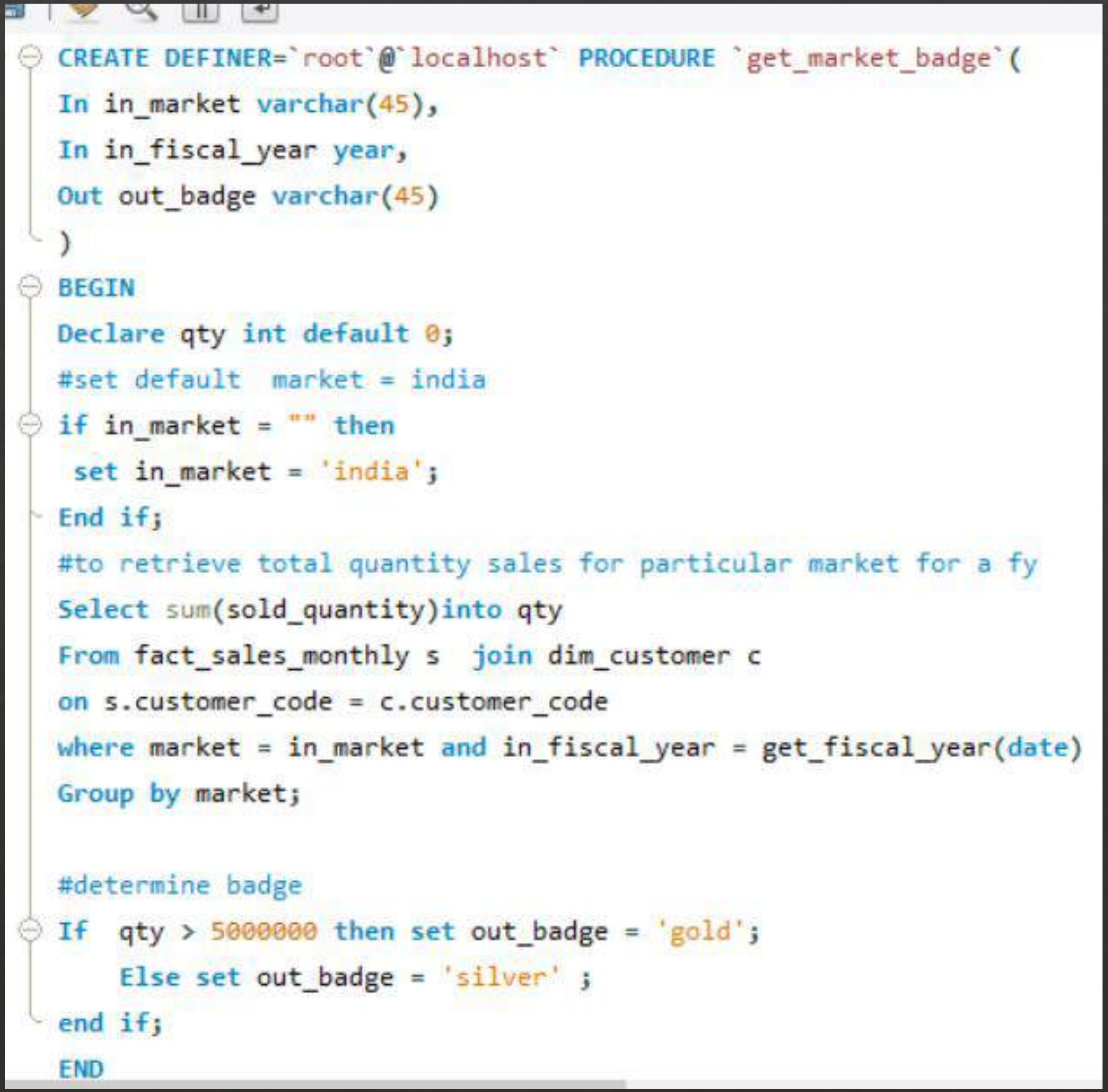
Create a stored procedure that can determine the market badge based on the following logic

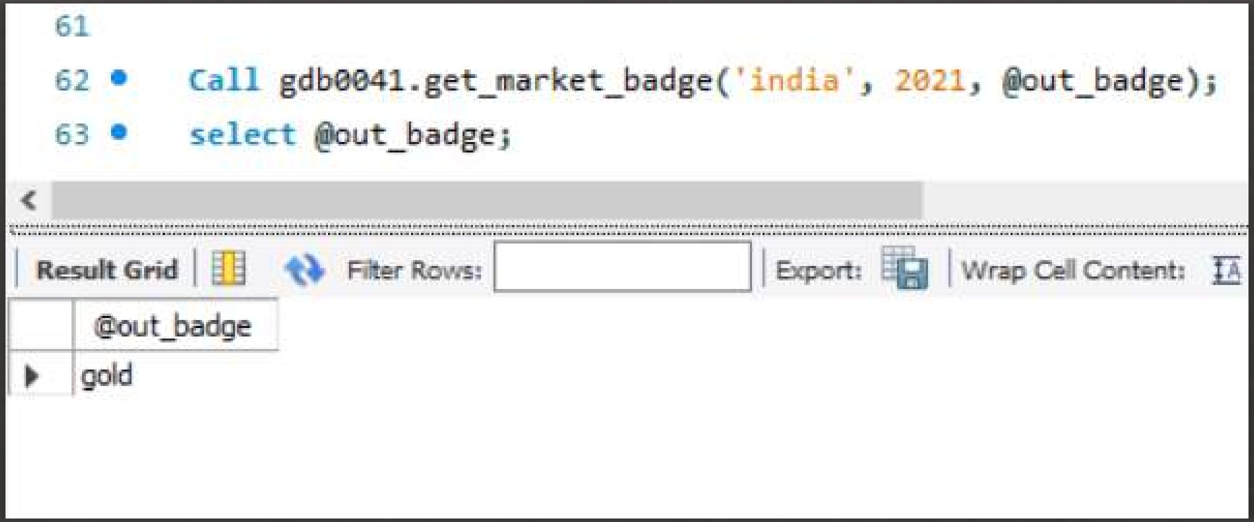
-- If total sales quantity > 5 million that market is considered as gold else it is silver

-- Input market and fiscal year

-- Output badge

-- To know where AtiliQ’s prominent sales are happening


```sql
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 qty int default 0;
#set default  market = india
if in_market = "" then
 set in_market = 'india';
End if;
#to retrieve total quantity sales for particular market for a fy
Select sum(sold_quantity)into qty
From fact_sales_monthly s  join dim_customer c
on s.customer_code = c.customer_code
where market = in_market and in_fiscal_year = get_fiscal_year(date)
Group by market;

#determine badge
If  qty > 5000000 then set out_badge = 'gold';
    Else set out_badge = 'silver' ;
end if;
END
```

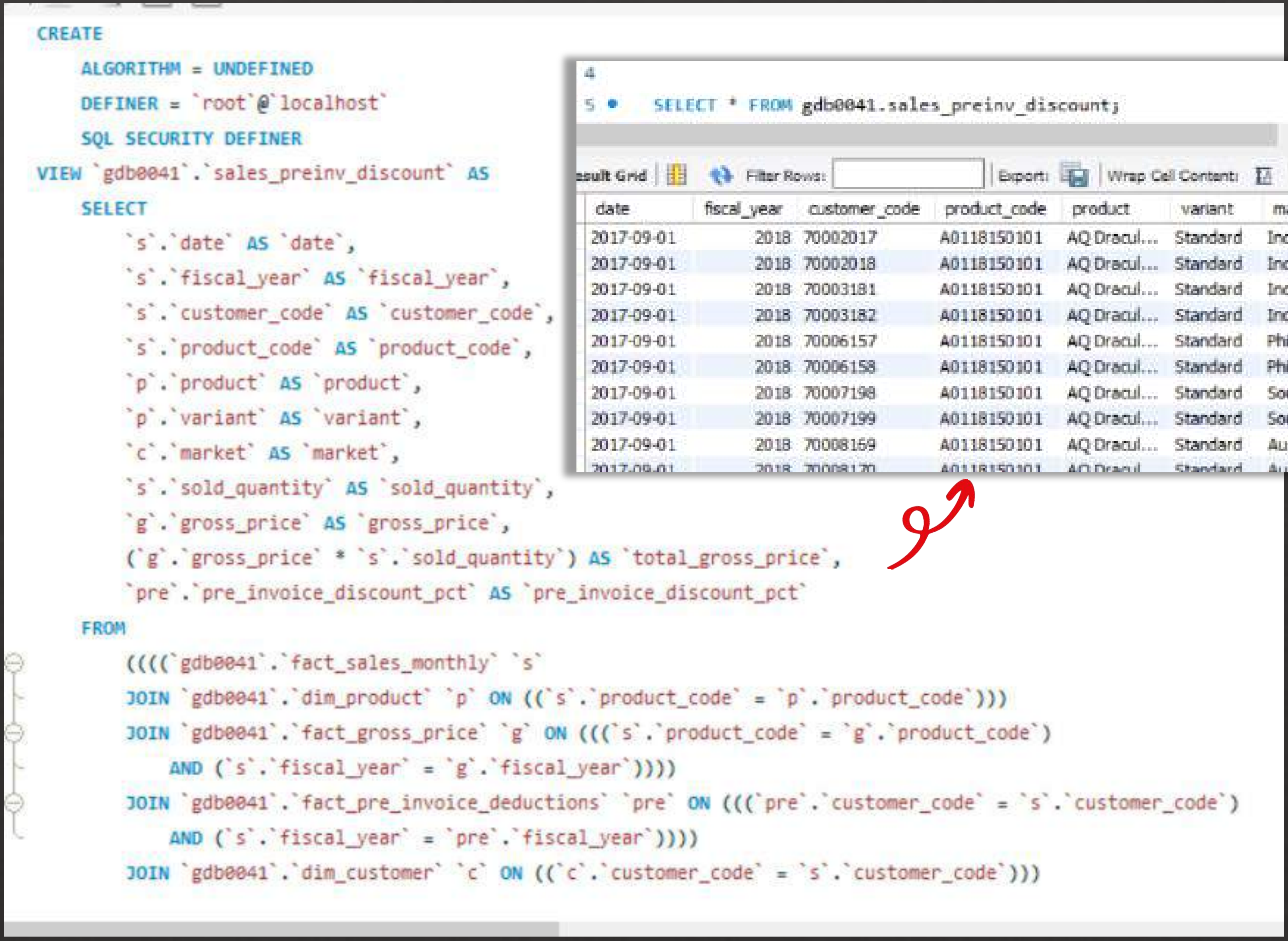

```sql
Call gdb0041.get_market_badge('india', 2021, @out_badge);
select @out_badge;
```

```
@out_badge
gold
```

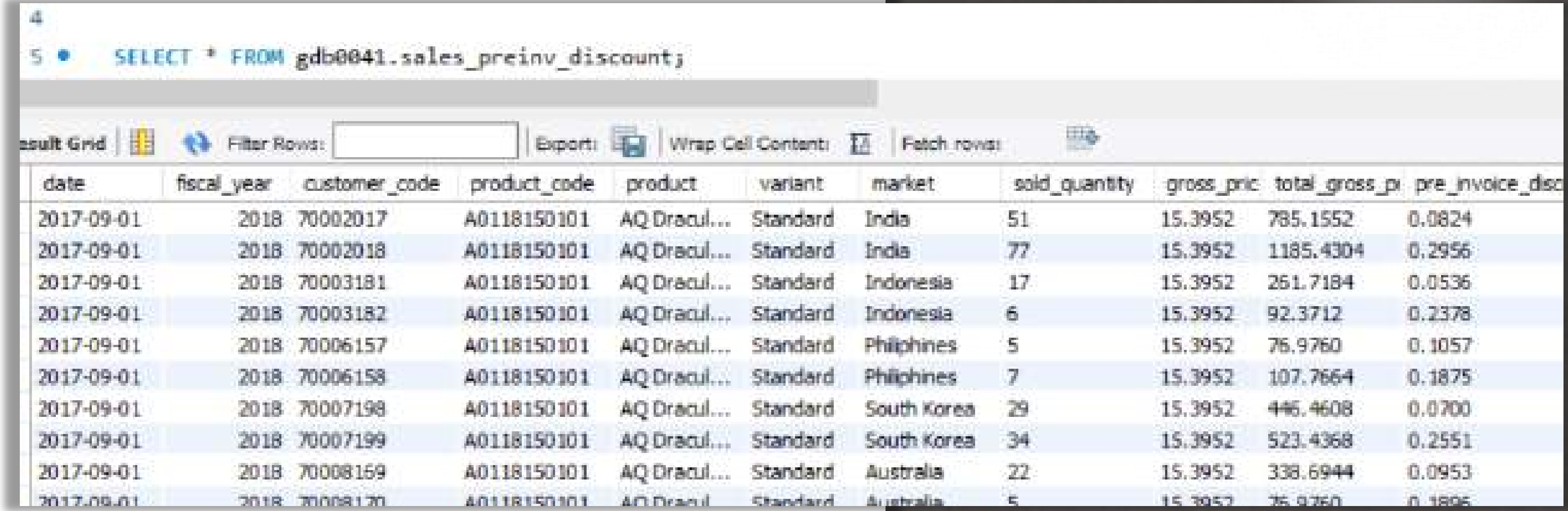
4. Top Markets, Customers, Products based on Net Sales

- Net pre invoice sales = Gross sales – Pre invoice deduction
- Net sales = Net pre invoice sales – Post invoice deduction

Created view for pre invoice discount



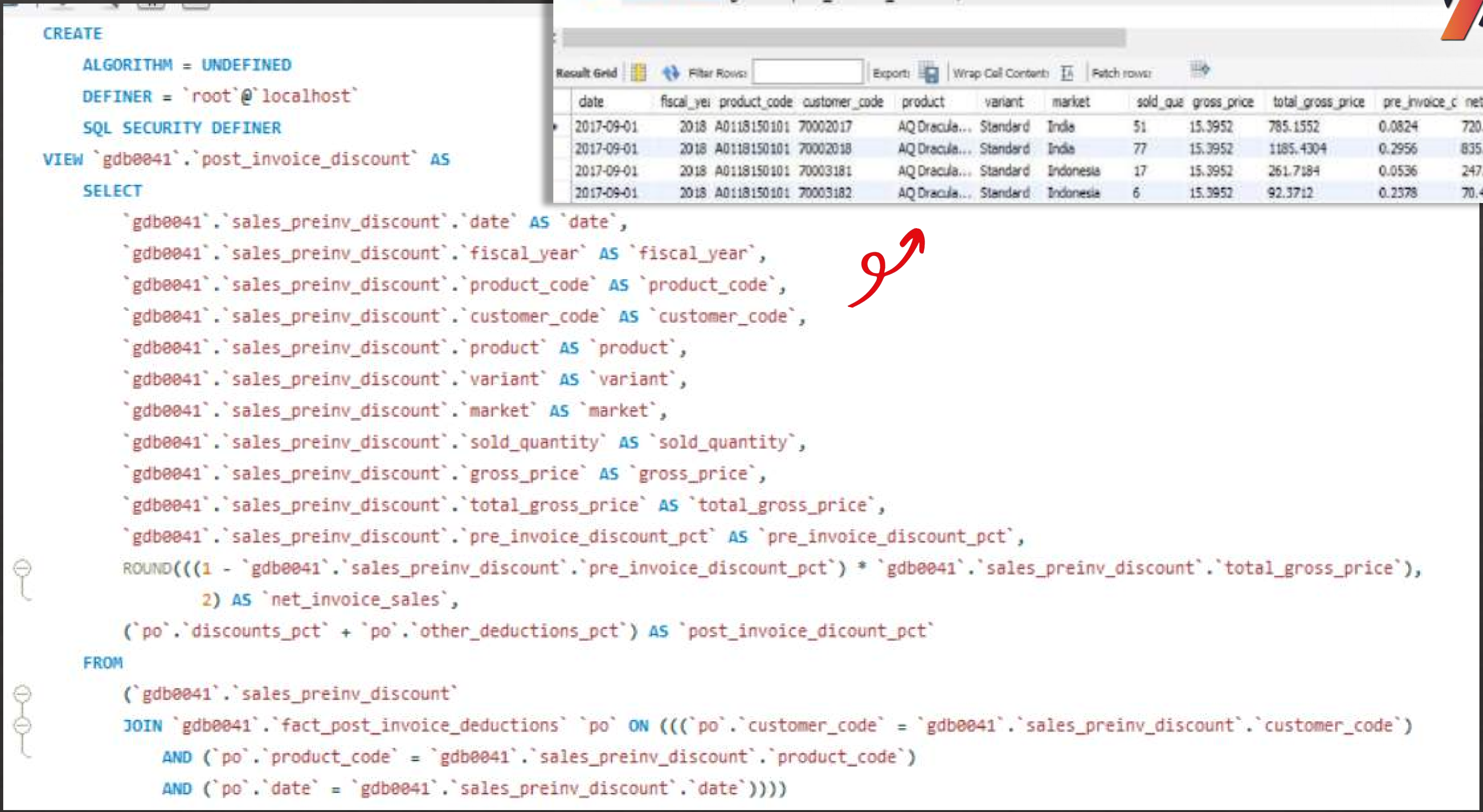
```sql
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `gdb0041`.`sales_preinv_discount` AS
    SELECT
        `s`.`date` AS `date`,
        `s`.`fiscal_year` AS `fiscal_year`,
        `s`.`customer_code` AS `customer_code`,
        `s`.`product_code` AS `product_code`,
        `p`.`product` AS `product`,
        `p`.`variant` AS `variant`,
        `c`.`market` AS `market`,
        `s`.`sold_quantity` AS `sold_quantity`,
        `g`.`gross_price` AS `gross_price`,
        (`g`.`gross_price` * `s`.`sold_quantity`) AS `total_gross_price`,
        `pre`.`pre_invoice_discount_pct` AS `pre_invoice_discount_pct`
    FROM
        ((((`gdb0041`.`fact_sales_monthly` `s`
        JOIN `gdb0041`.`dim_product` `p` ON ((`s`.`product_code` = `p`.`product_code`)))
        JOIN `gdb0041`.`fact_gross_price` `g` ON (((`s`.`product_code` = `g`.`product_code`)
            AND (`s`.`fiscal_year` = `g`.`fiscal_year`))))
        JOIN `gdb0041`.`fact_pre_invoice_deductions` `pre` ON (((`pre`.`customer_code` = `s`.`customer_code`)
            AND (`s`.`fiscal_year` = `pre`.`fiscal_year`))))
        JOIN `gdb0041`.`dim_customer` `c` ON ((`c`.`customer_code` = `s`.`customer_code`)))
```



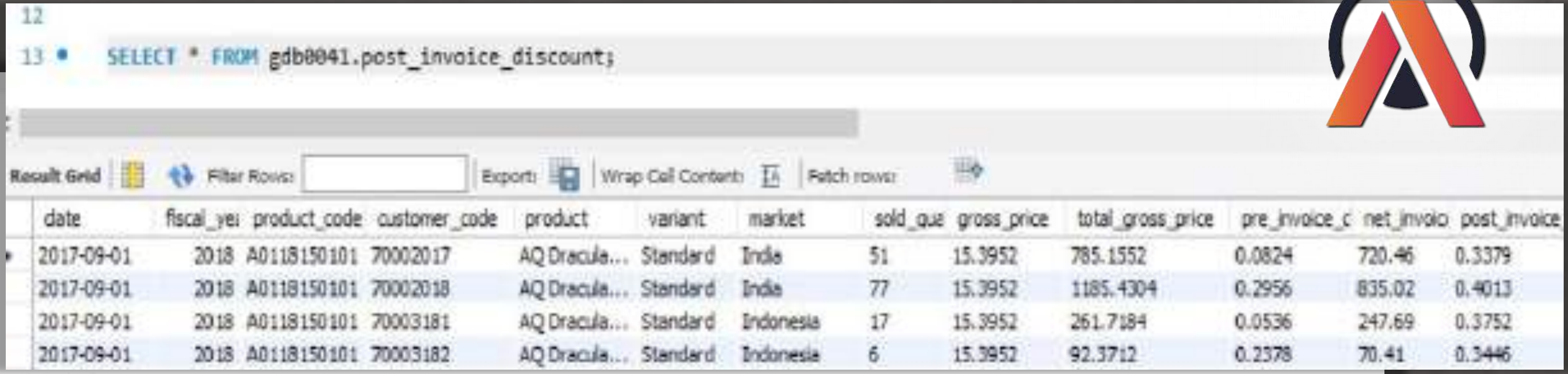
```sql
SELECT * FROM gdb0041.sales_preinv_discount;
```

date	fiscal_year	customer_code	product_code	product	variant	market	sold_quantity	gross_pric	total_gross_pi	pre_invoice_disc
2017-09-01	2018	70002017	A0118150101	AQ Dracul...	Standard	India	51	15.3952	785.1552	0.0824
2017-09-01	2018	70002018	A0118150101	AQ Dracul...	Standard	India	77	15.3952	1185.4304	0.2956
2017-09-01	2018	70003181	A0118150101	AQ Dracul...	Standard	Indonesia	17	15.3952	261.7184	0.0536
2017-09-01	2018	70003182	A0118150101	AQ Dracul...	Standard	Indonesia	6	15.3952	92.3712	0.2378
2017-09-01	2018	70006157	A0118150101	AQ Dracul...	Standard	Philiphines	5	15.3952	76.9760	0.1057
2017-09-01	2018	70006158	A0118150101	AQ Dracul...	Standard	Philiphines	7	15.3952	107.7664	0.1875
2017-09-01	2018	70007198	A0118150101	AQ Dracul...	Standard	South Korea	29	15.3952	446.4608	0.0700
2017-09-01	2018	70007199	A0118150101	AQ Dracul...	Standard	South Korea	34	15.3952	523.4368	0.2551
2017-09-01	2018	70008169	A0118150101	AQ Dracul...	Standard	Australia	22	15.3952	338.6944	0.0953
2017-09-01	2018	70008170	A0118150101	AQ Dracul...	Standard	Australia	5	15.3952	76.9760	0.1896

Created view for post invoice discount



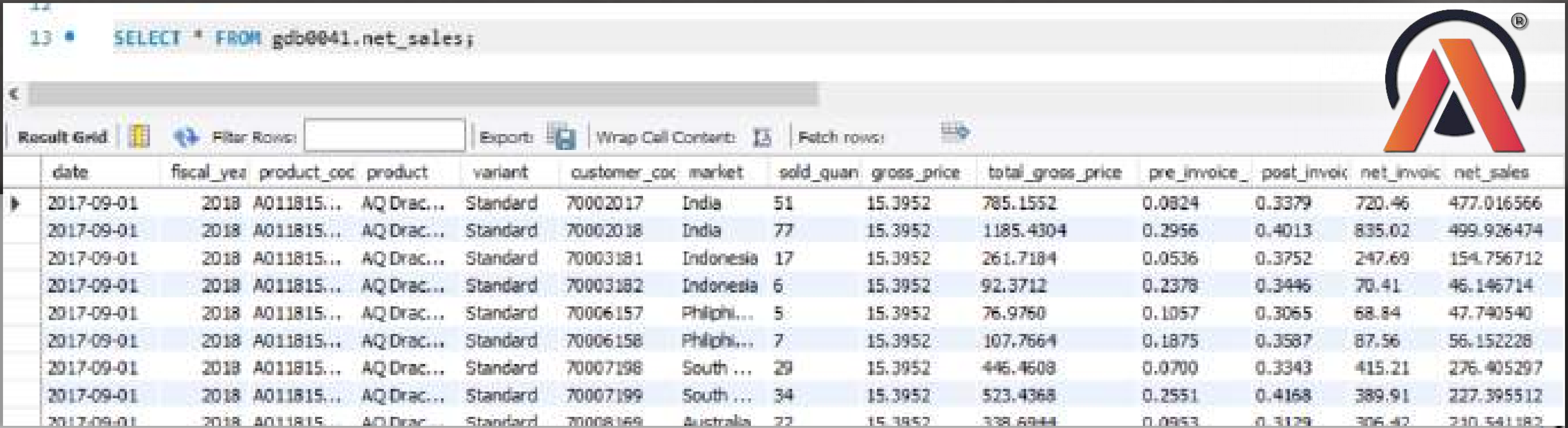
```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `gdb0041`.`post_invoice_discount` AS
    SELECT
        `gdb0041`.`sales_preinv_discount`.`date` AS `date`,
        `gdb0041`.`sales_preinv_discount`.`fiscal_year` AS `fiscal_year`,
        `gdb0041`.`sales_preinv_discount`.`product_code` AS `product_code`,
        `gdb0041`.`sales_preinv_discount`.`customer_code` AS `customer_code`,
        `gdb0041`.`sales_preinv_discount`.`product` AS `product`,
        `gdb0041`.`sales_preinv_discount`.`variant` AS `variant`,
        `gdb0041`.`sales_preinv_discount`.`market` AS `market`,
        `gdb0041`.`sales_preinv_discount`.`sold_quantity` AS `sold_quantity`,
        `gdb0041`.`sales_preinv_discount`.`gross_price` AS `gross_price`,
        `gdb0041`.`sales_preinv_discount`.`total_gross_price` AS `total_gross_price`,
        `gdb0041`.`sales_preinv_discount`.`pre_invoice_discount_pct` AS `pre_invoice_discount_pct`,
        ROUND(((1 - `gdb0041`.`sales_preinv_discount`.`pre_invoice_discount_pct`) * `gdb0041`.`sales_preinv_discount`.`total_gross_price`),
                2) AS `net_invoice_sales`,
        (`po`.`discounts_pct` + `po`.`other_deductions_pct`) AS `post_invoice_dicount_pct`
    FROM
        (`gdb0041`.`sales_preinv_discount`
        JOIN `gdb0041`.`fact_post_invoice_deductions` `po` ON (((`po`.`customer_code` = `gdb0041`.`sales_preinv_discount`.`customer_code`)
            AND (`po`.`product_code` = `gdb0041`.`sales_preinv_discount`.`product_code`)
            AND (`po`.`date` = `gdb0041`.`sales_preinv_discount`.`date`))))
```



```
SELECT * FROM gdb0041.post_invoice_discount;
```

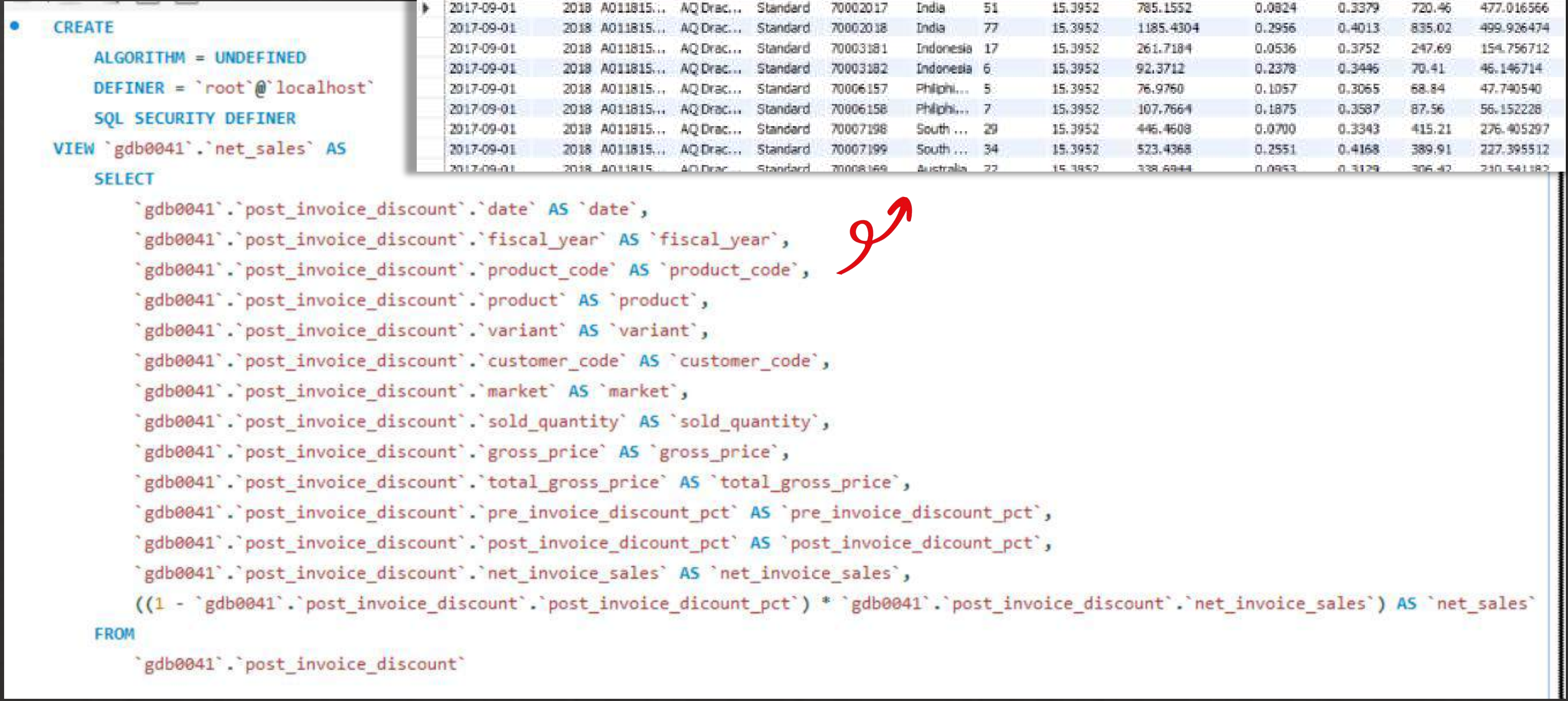
date	fiscal_yea	product_code	customer_code	product	variant	market	sold_qua	gross_price	total_gross_price	pre_invoice_c	net_invoic	post_invoice
2017-09-01	2018	A0118150101	70002017	AQ Dracula...	Standard	India	51	15.3952	785.1552	0.0824	720.46	0.3379
2017-09-01	2018	A0118150101	70002018	AQ Dracula...	Standard	India	77	15.3952	1185.4304	0.2956	835.02	0.4013
2017-09-01	2018	A0118150101	70003181	AQ Dracula...	Standard	Indonesia	17	15.3952	261.7184	0.0536	247.69	0.3752
2017-09-01	2018	A0118150101	70003182	AQ Dracula...	Standard	Indonesia	6	15.3952	92.3712	0.2378	70.41	0.3446

Created view for net sales



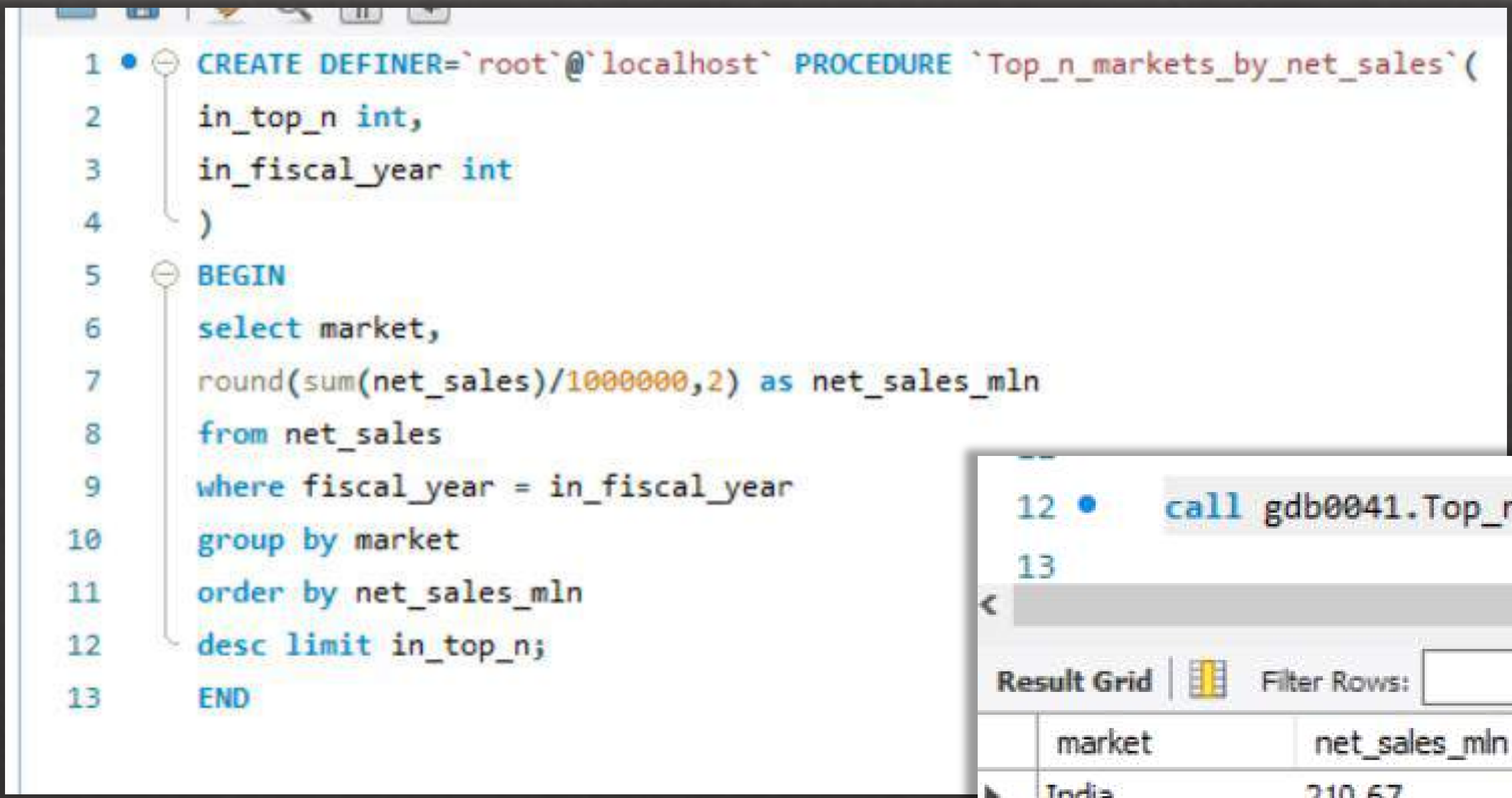
```
SELECT * FROM gdb0041.net_sales;
```

date	fiscal_yea	product_coc	product	variant	customer_coc	market	sold_quan	gross_price	total_gross_price	pre_invoice_	post_invoic	net_invoic	net_sales
2017-09-01	2018	A011815...	AQ Drac...	Standard	70002017	India	51	15.3952	785.1552	0.0824	0.3379	720.46	477.016566
2017-09-01	2018	A011815...	AQ Drac...	Standard	70002018	India	77	15.3952	1185.4304	0.2956	0.4013	835.02	499.926474
2017-09-01	2018	A011815...	AQ Drac...	Standard	70003181	Indonesia	17	15.3952	261.7184	0.0536	0.3752	247.69	154.756712
2017-09-01	2018	A011815...	AQ Drac...	Standard	70003182	Indonesia	6	15.3952	92.3712	0.2378	0.3446	70.41	46.146714
2017-09-01	2018	A011815...	AQ Drac...	Standard	70006157	Philiphi...	5	15.3952	76.9760	0.1057	0.3065	68.84	47.740540
2017-09-01	2018	A011815...	AQ Drac...	Standard	70006158	Philiphi...	7	15.3952	107.7664	0.1875	0.3587	87.56	56.152228
2017-09-01	2018	A011815...	AQ Drac...	Standard	70007198	South ...	29	15.3952	446.4608	0.0700	0.3343	415.21	276.405297
2017-09-01	2018	A011815...	AQ Drac...	Standard	70007199	South ...	34	15.3952	523.4368	0.2551	0.4168	389.91	227.395512
2017-09-01	2018	A011815...	AQ Drac...	Standard	70008169	Australia	22	15.3952	338.6944	0.0953	0.3129	306.42	210.541182

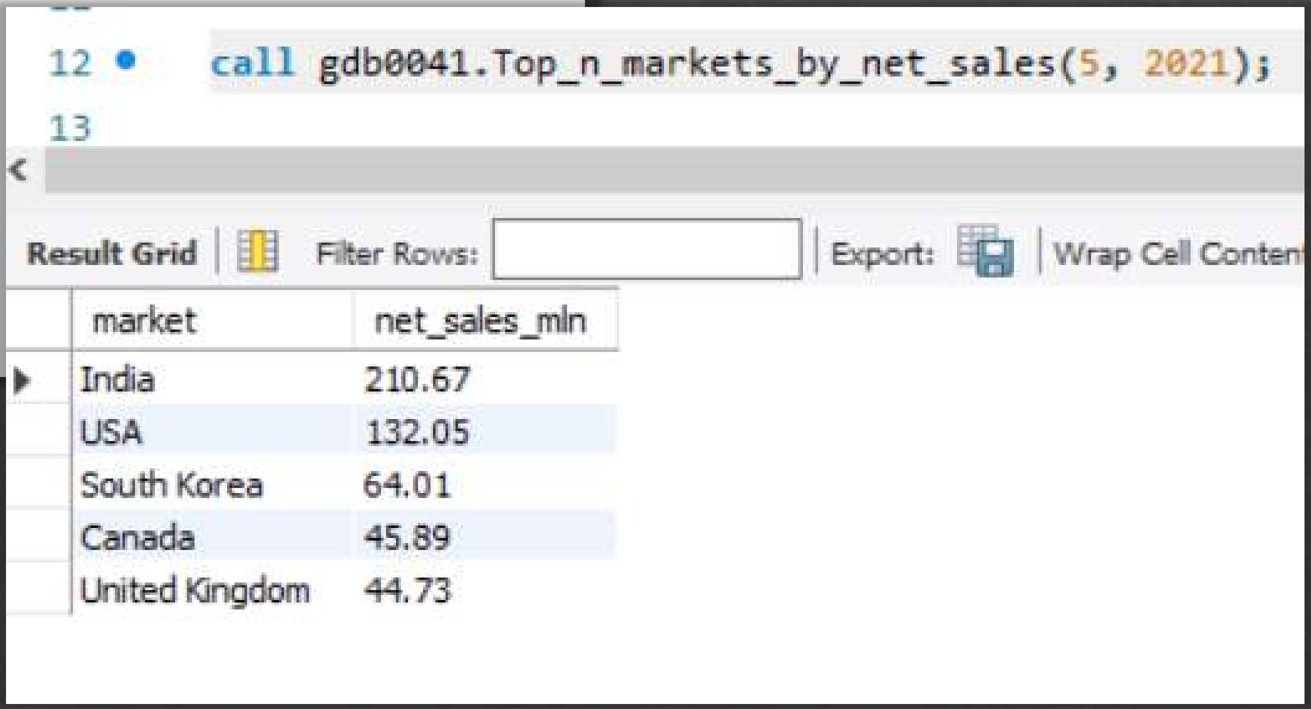


```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `gdb0041`.`net_sales` AS
    SELECT
        `gdb0041`.`post_invoice_discount`.`date` AS `date`,
        `gdb0041`.`post_invoice_discount`.`fiscal_year` AS `fiscal_year`,
        `gdb0041`.`post_invoice_discount`.`product_code` AS `product_code`,
        `gdb0041`.`post_invoice_discount`.`product` AS `product`,
        `gdb0041`.`post_invoice_discount`.`variant` AS `variant`,
        `gdb0041`.`post_invoice_discount`.`customer_code` AS `customer_code`,
        `gdb0041`.`post_invoice_discount`.`market` AS `market`,
        `gdb0041`.`post_invoice_discount`.`sold_quantity` AS `sold_quantity`,
        `gdb0041`.`post_invoice_discount`.`gross_price` AS `gross_price`,
        `gdb0041`.`post_invoice_discount`.`total_gross_price` AS `total_gross_price`,
        `gdb0041`.`post_invoice_discount`.`pre_invoice_discount_pct` AS `pre_invoice_discount_pct`,
        `gdb0041`.`post_invoice_discount`.`post_invoice_dicount_pct` AS `post_invoice_dicount_pct`,
        `gdb0041`.`post_invoice_discount`.`net_invoice_sales` AS `net_invoice_sales`,
        ((1 - `gdb0041`.`post_invoice_discount`.`post_invoice_dicount_pct`) * `gdb0041`.`post_invoice_discount`.`net_invoice_sales`) AS `net_sales`
    FROM
        `gdb0041`.`post_invoice_discount`
```

Top markets by net sales



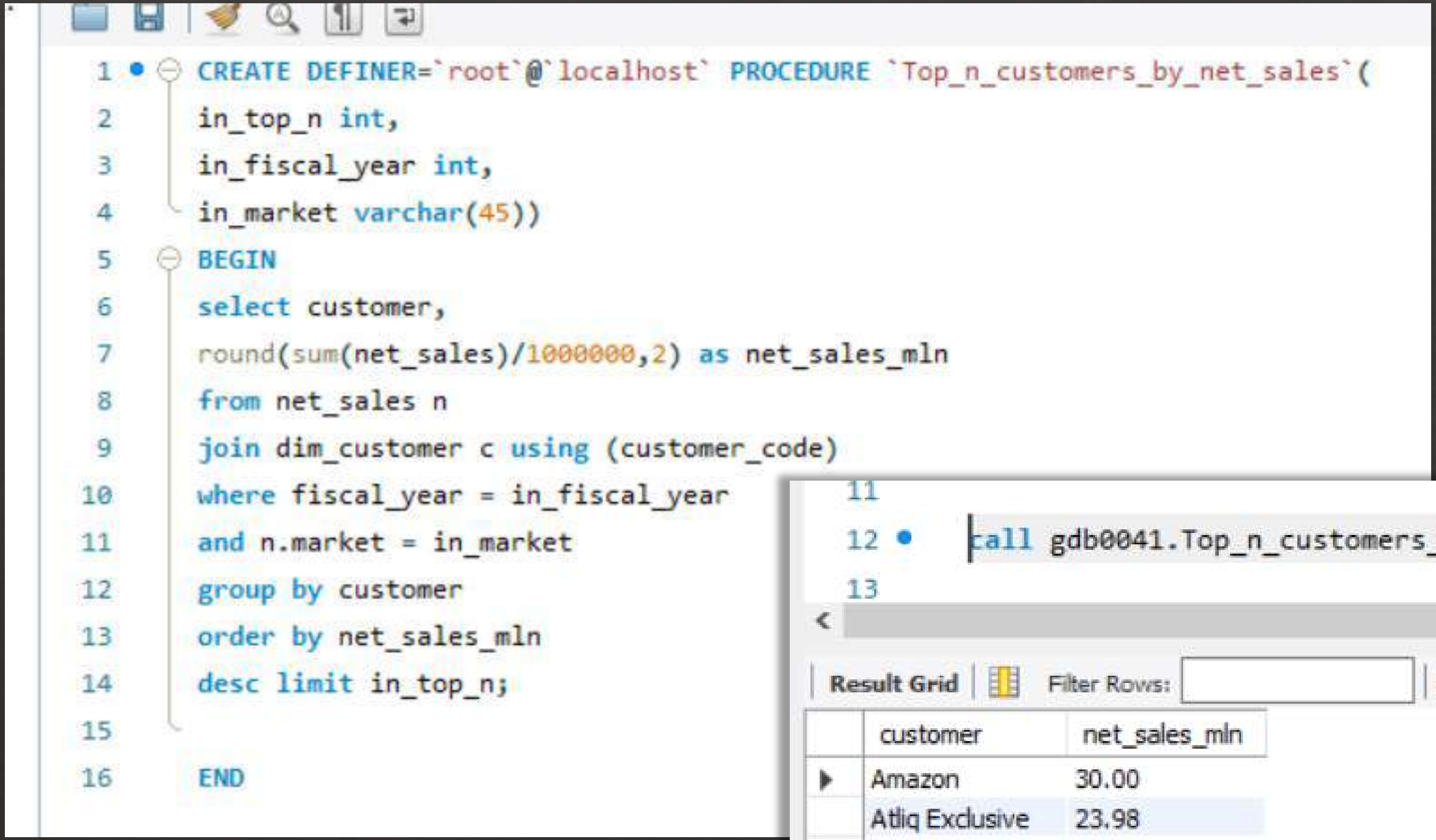
```
CREATE DEFINER=`root`@`localhost` PROCEDURE `Top_n_markets_by_net_sales`(
in_top_n int,
in_fiscal_year int
)
BEGIN
select market,
round(sum(net_sales)/1000000,2) as net_sales_mln
from net_sales
where fiscal_year = in_fiscal_year
group by market
order by net_sales_mln
desc limit in_top_n;
END
```



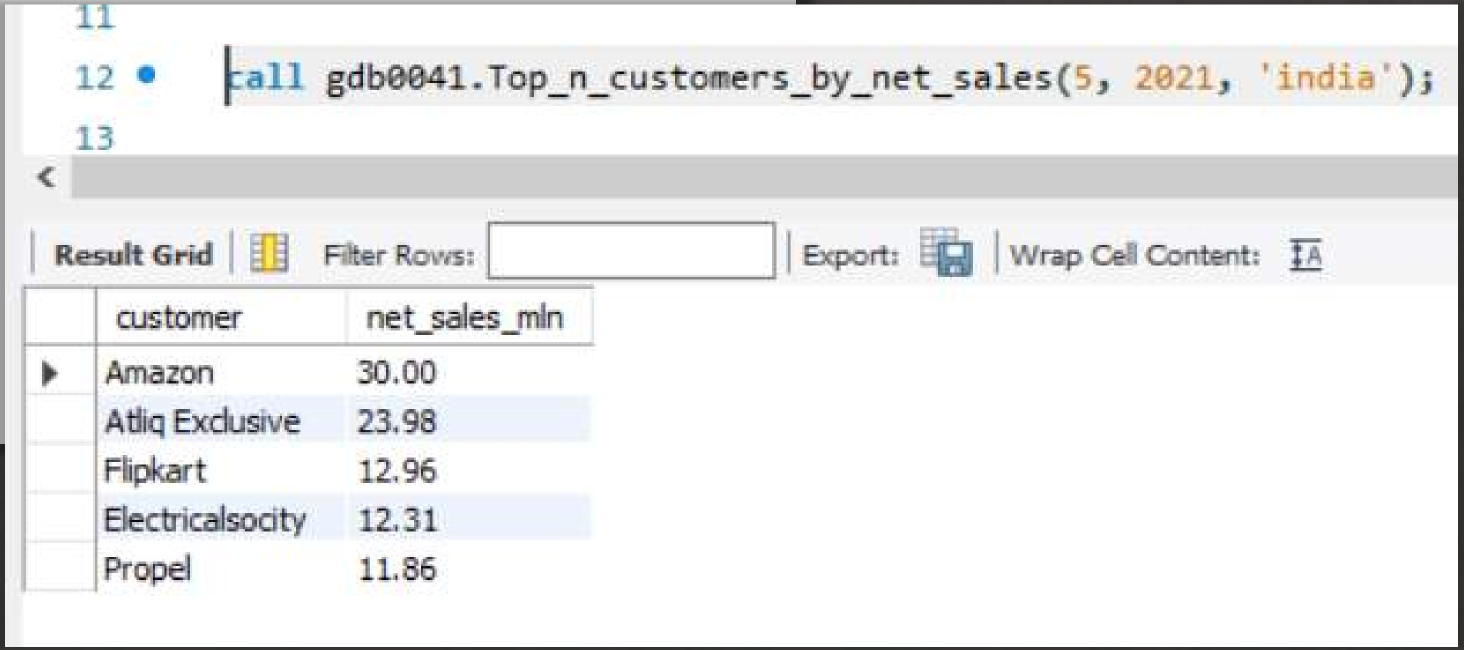
```
call gdb0041.Top_n_markets_by_net_sales(5, 2021);
```

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

Top customers by net sales



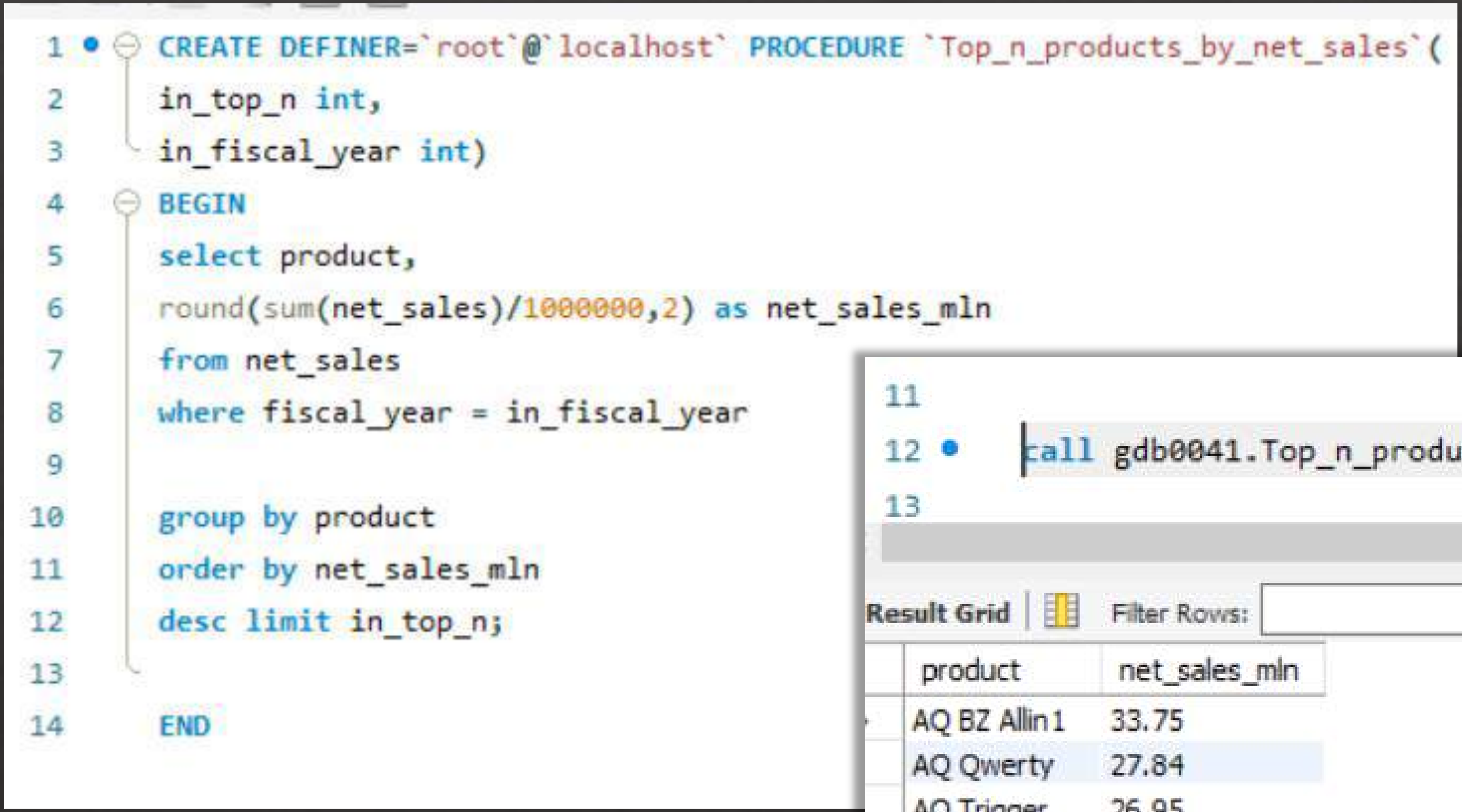
```sql
CREATE DEFINER=`root`@`localhost` PROCEDURE `Top_n_customers_by_net_sales`(
in_top_n int,
in_fiscal_year int,
in_market varchar(45))
BEGIN
select customer,
round(sum(net_sales)/1000000,2) as net_sales_mln
from net_sales n
join dim_customer c using (customer_code)
where fiscal_year = in_fiscal_year
and n.market = in_market
group by customer
order by net_sales_mln
desc limit in_top_n;

END
```



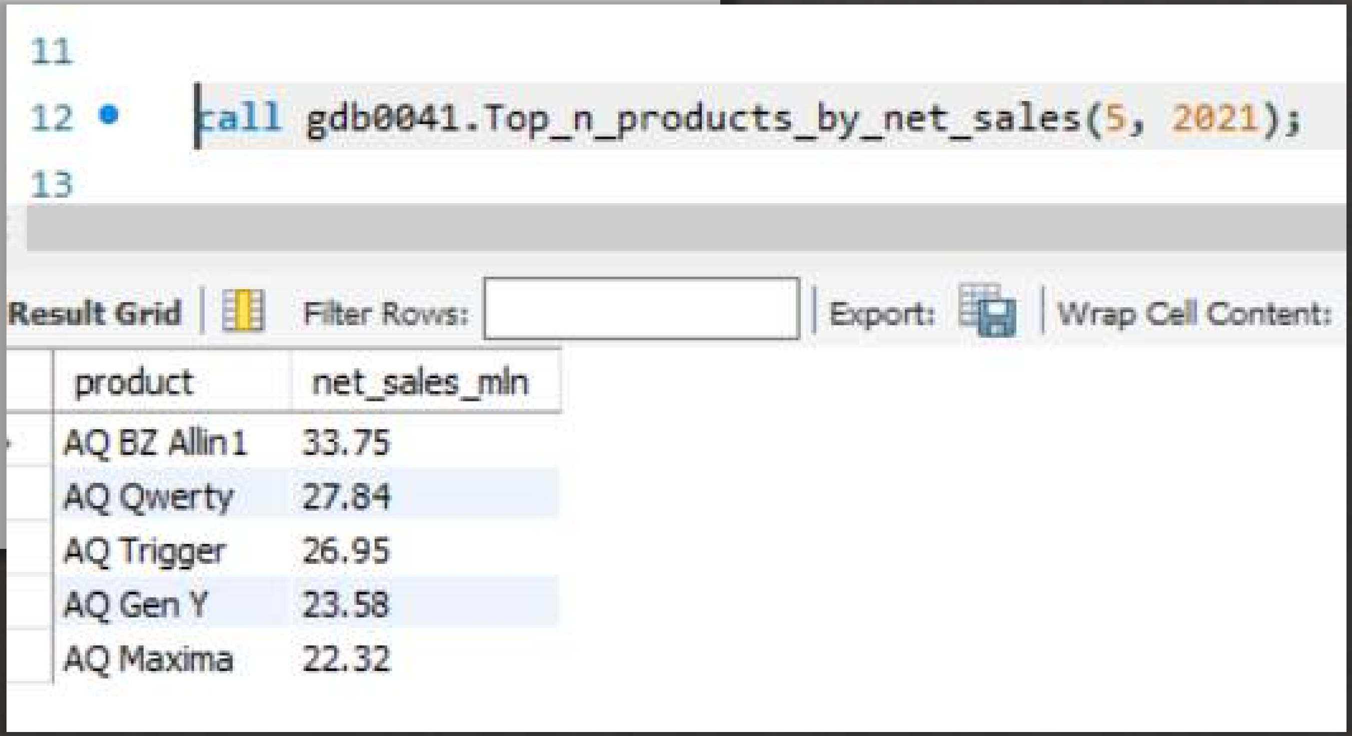
```sql
call gdb0041.Top_n_customers_by_net_sales(5, 2021, 'india');
```

customer	net_sales_mln
Amazon	30.00
Atliq Exclusive	23.98
Flipkart	12.96
Electricalsocity	12.31
Propel	11.86

Top products by net sales



```sql
CREATE DEFINER=`root`@`localhost` PROCEDURE `Top_n_products_by_net_sales`(
in_top_n int,
in_fiscal_year 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
```



```sql
call gdb0041.Top_n_products_by_net_sales(5, 2021);
```

product	net_sales_mln
AQ BZ Allin1	33.75
AQ Qwerty	27.84
AQ Trigger	26.95
AQ Gen Y	23.58
AQ Maxima	22.32

5. Gross sales view

With following columns

-- date

-- fiscal_year

-- customer_code

-- customer

-- market

-- product_code

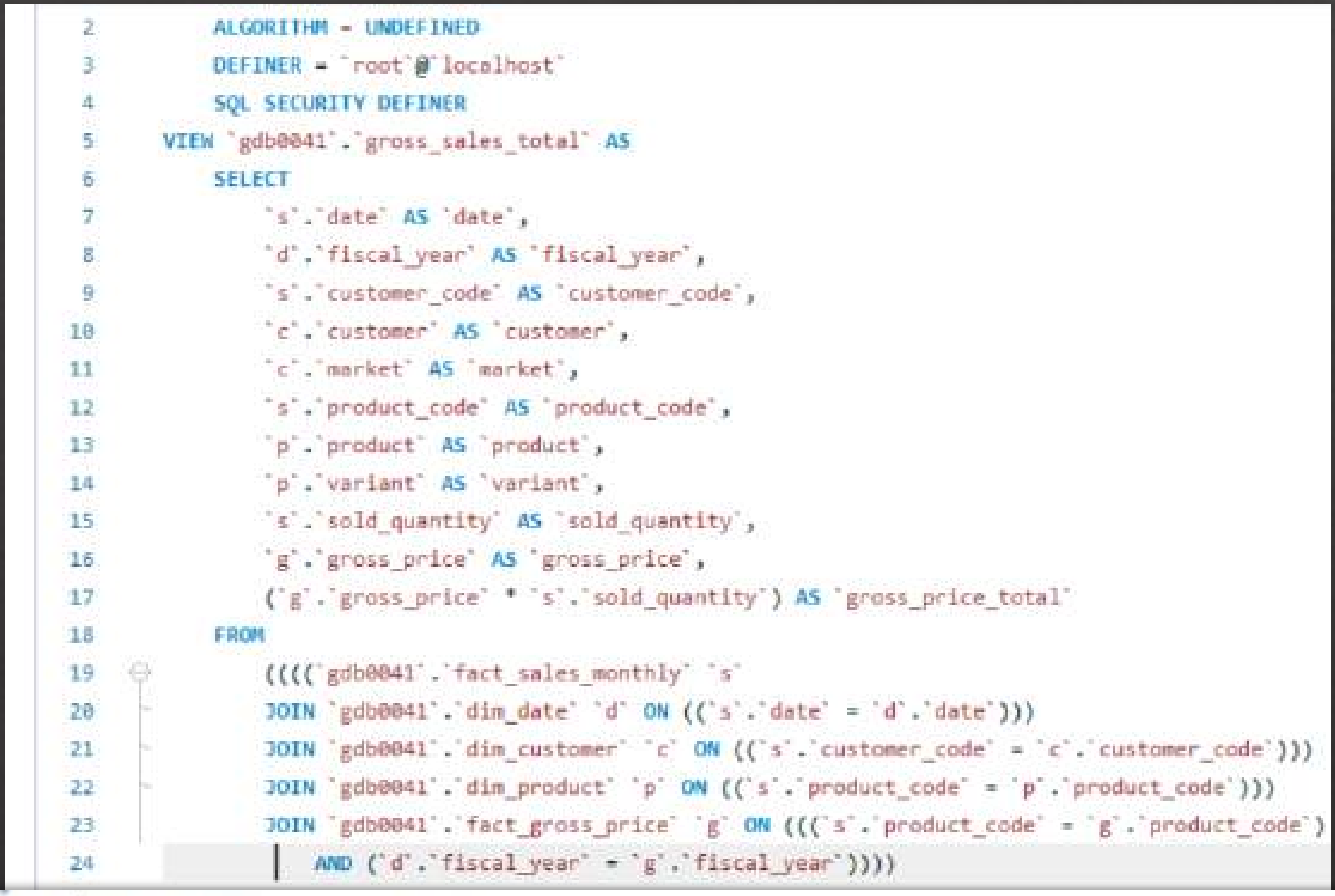
-- product

-- variant

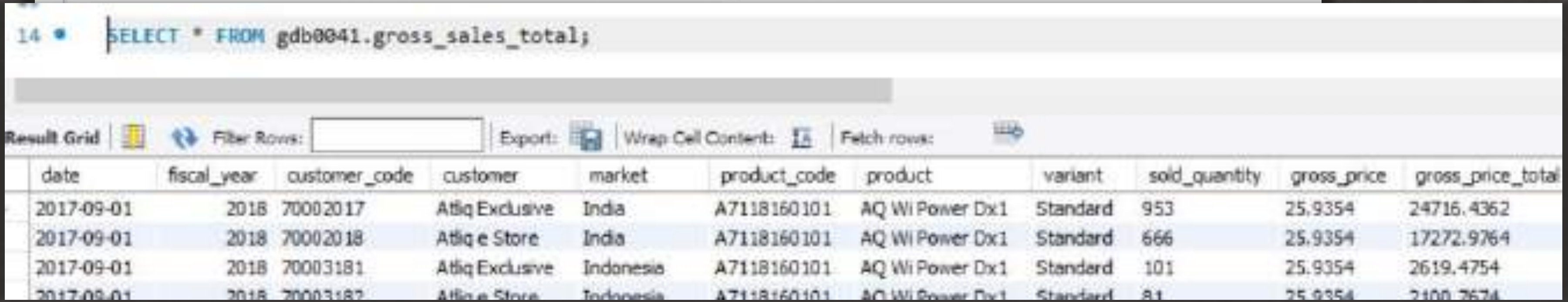
-- sold_quantity

-- gross_price_per_item

-- gross_price_total



```sql
        ALGORITHM = UNDEFINED
        DEFINER = `root`@`localhost`
        SQL SECURITY DEFINER
    VIEW `gdb0041`.`gross_sales_total` AS
        SELECT
            `s`.`date` AS `date`,
            `d`.`fiscal_year` AS `fiscal_year`,
            `s`.`customer_code` AS `customer_code`,
            `c`.`customer` AS `customer`,
            `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`,
            (`g`.`gross_price` * `s`.`sold_quantity`) AS `gross_price_total`
        FROM
            ((((`gdb0041`.`fact_sales_monthly` `s`
            JOIN `gdb0041`.`dim_date` `d` ON ((`s`.`date` = `d`.`date`)))
            JOIN `gdb0041`.`dim_customer` `c` ON ((`s`.`customer_code` = `c`.`customer_code`)))
            JOIN `gdb0041`.`dim_product` `p` ON ((`s`.`product_code` = `p`.`product_code`)))
            JOIN `gdb0041`.`fact_gross_price` `g` ON (((`s`.`product_code` = `g`.`product_code`)
                AND (`d`.`fiscal_year` = `g`.`fiscal_year`))))
```

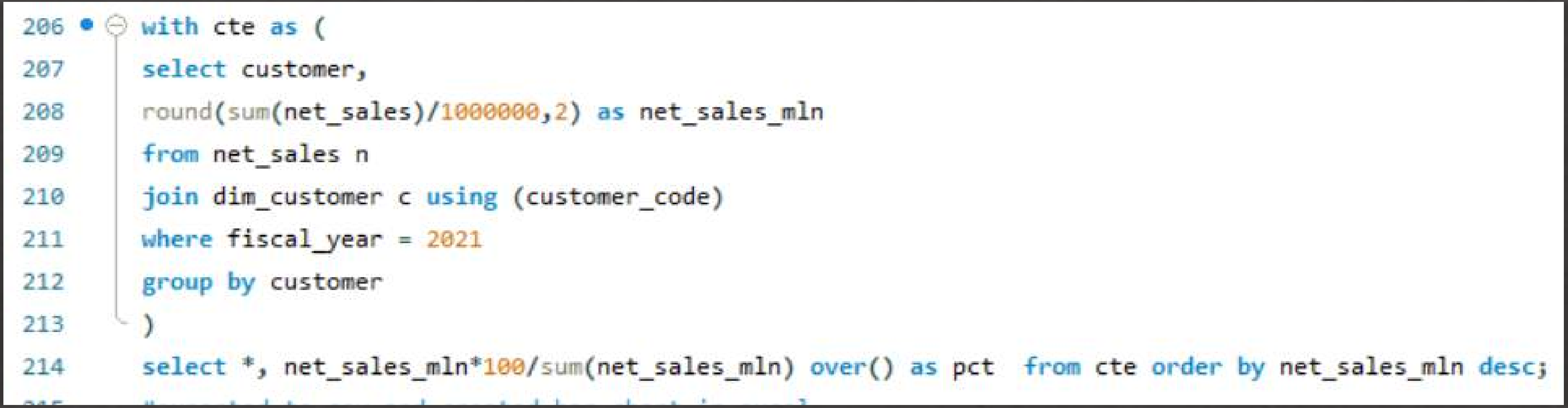


```sql
SELECT * FROM gdb0041.gross_sales_total;
```

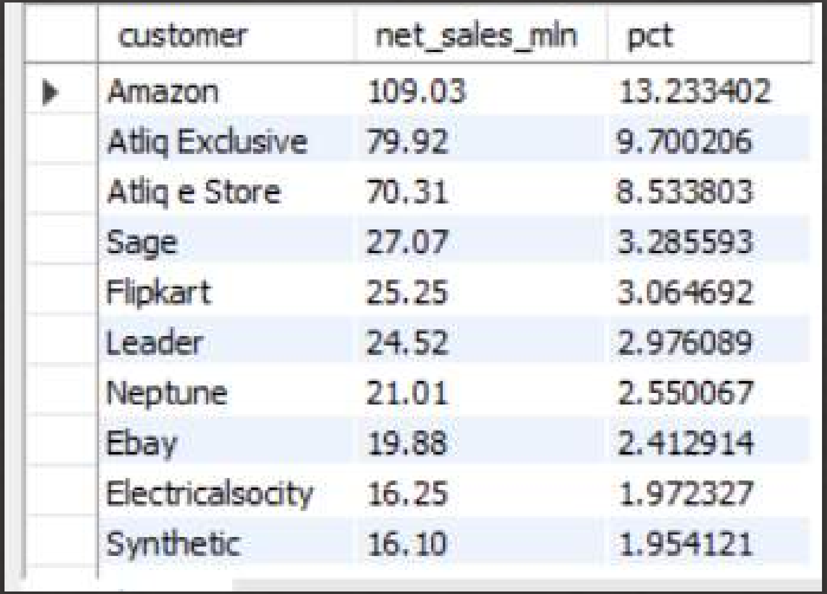
Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

date	fiscal_year	customer_code	customer	market	product_code	product	variant	sold_quantity	gross_price	gross_price_total
2017-09-01	2018	70002017	Atliq Exclusive	India	A7118160101	AQ Wi Power Dx1	Standard	953	25.9354	24716.4362
2017-09-01	2018	70002018	Atliq e Store	India	A7118160101	AQ Wi Power Dx1	Standard	666	25.9354	17272.9764
2017-09-01	2018	70003181	Atliq Exclusive	Indonesia	A7118160101	AQ Wi Power Dx1	Standard	101	25.9354	2619.4754
2017-09-01	2018	70003182	Atliq e Store	Indonesia	A7118160101	AQ Wi Power Dx1	Standard	81	25.9354	2100.7674

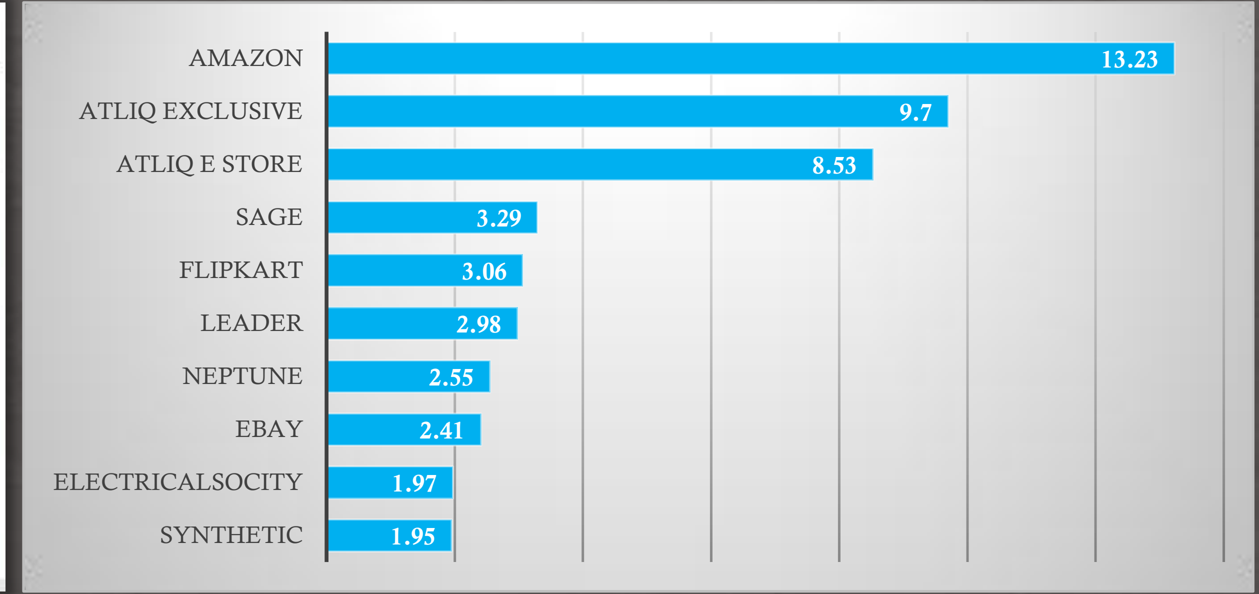
6. Percentage contribution on global net sales by given customer



```sql
with cte as (
select customer,
round(sum(net_sales)/1000000,2) as net_sales_mln
from net_sales n
join dim_customer c using (customer_code)
where fiscal_year = 2021
group by customer
)
select *, net_sales_mln*100/sum(net_sales_mln) over() as pct  from cte order by net_sales_mln desc;
```

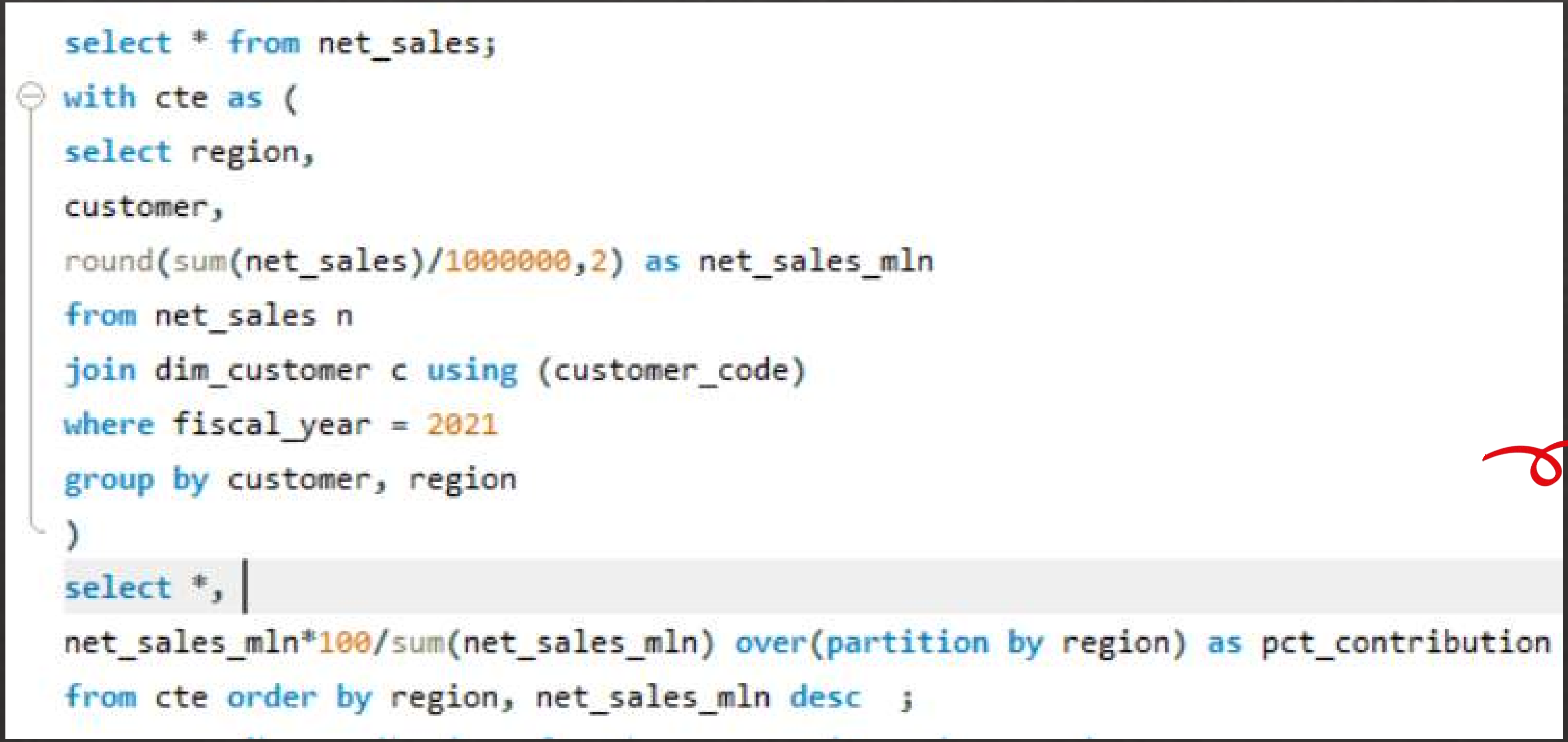


customer	net_sales_mln	pct
Amazon	109.03	13.233402
Atliq Exclusive	79.92	9.700206
Atliq e Store	70.31	8.533803
Sage	27.07	3.285593
Flipkart	25.25	3.064692
Leader	24.52	2.976089
Neptune	21.01	2.550067
Ebay	19.88	2.412914
Electricalsocity	16.25	1.972327
Synthetic	16.10	1.954121

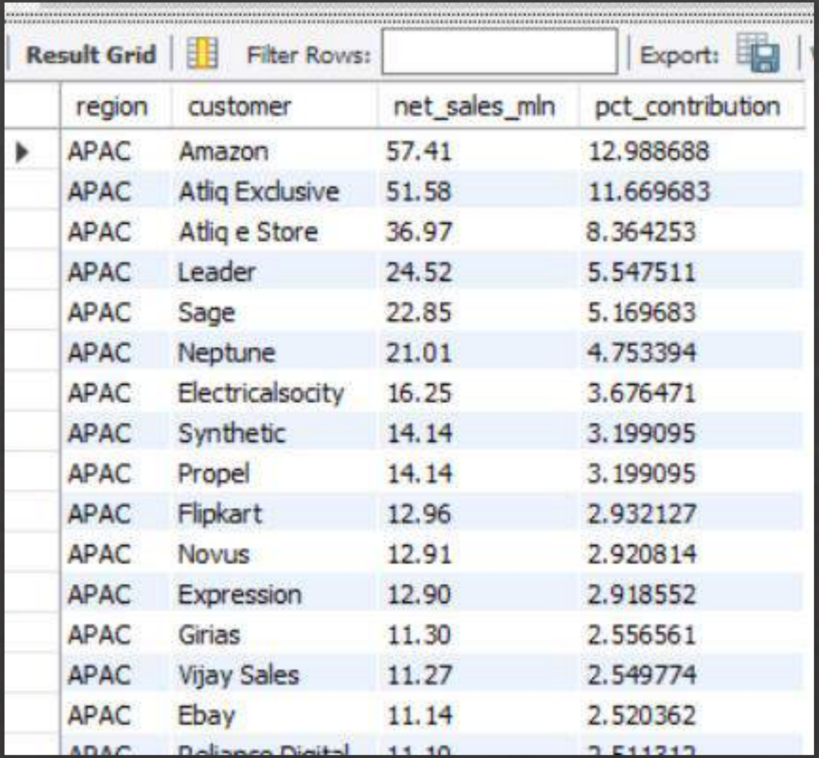


7. Percentage contribution on region wise net sales by given customer

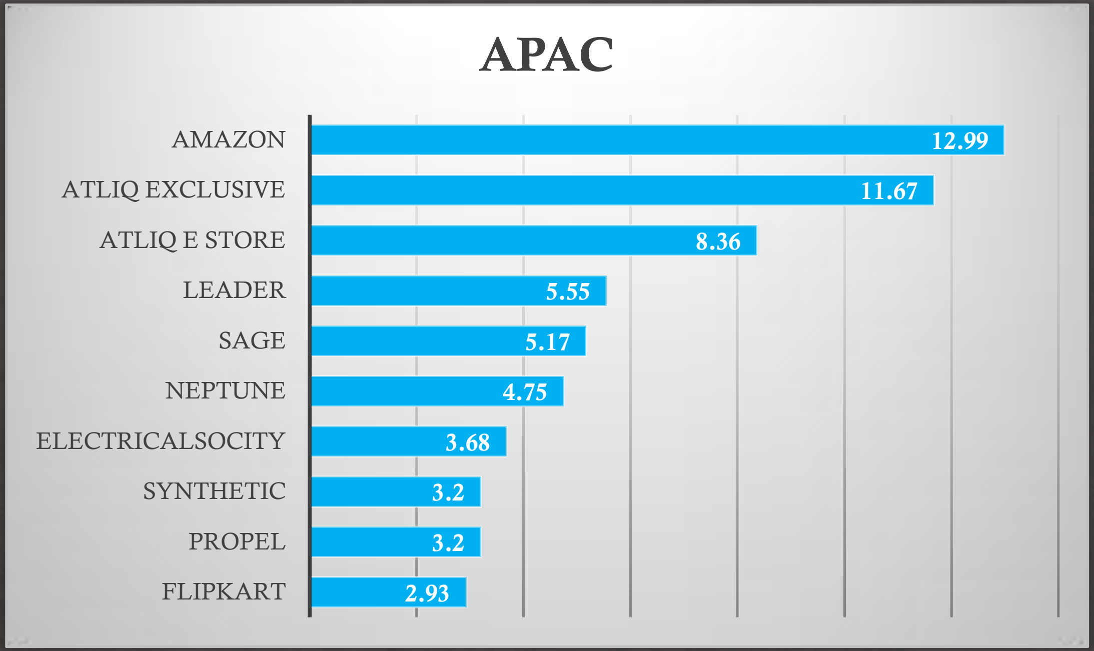
Region wise percentage net sales breakdown by customers to perform regional analysis on financial performance of the company for 2021

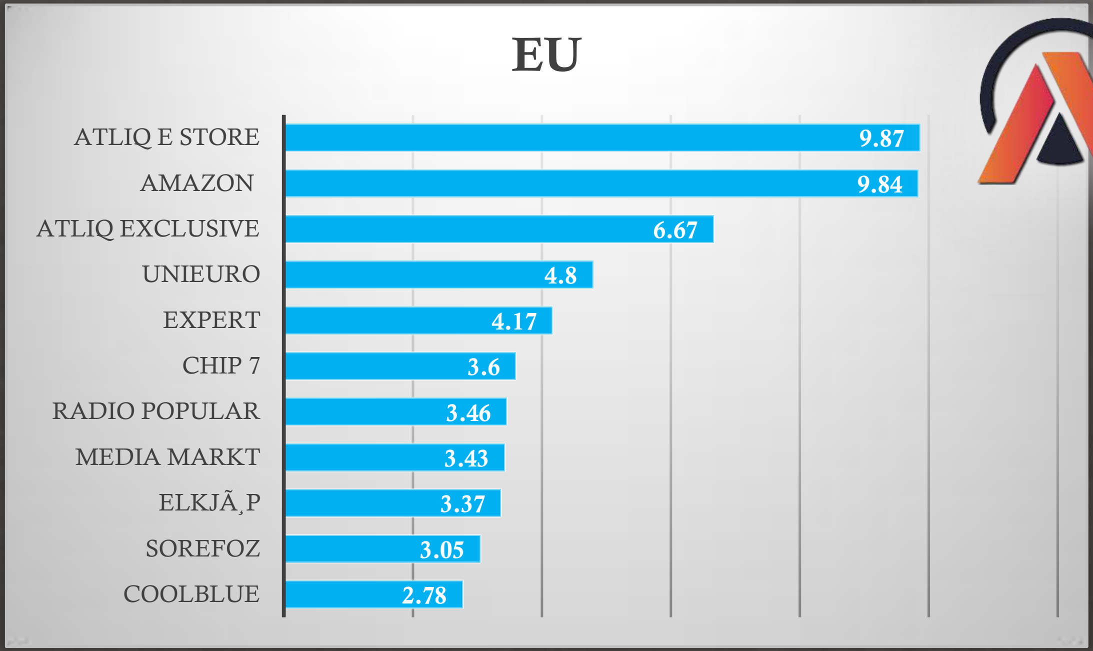


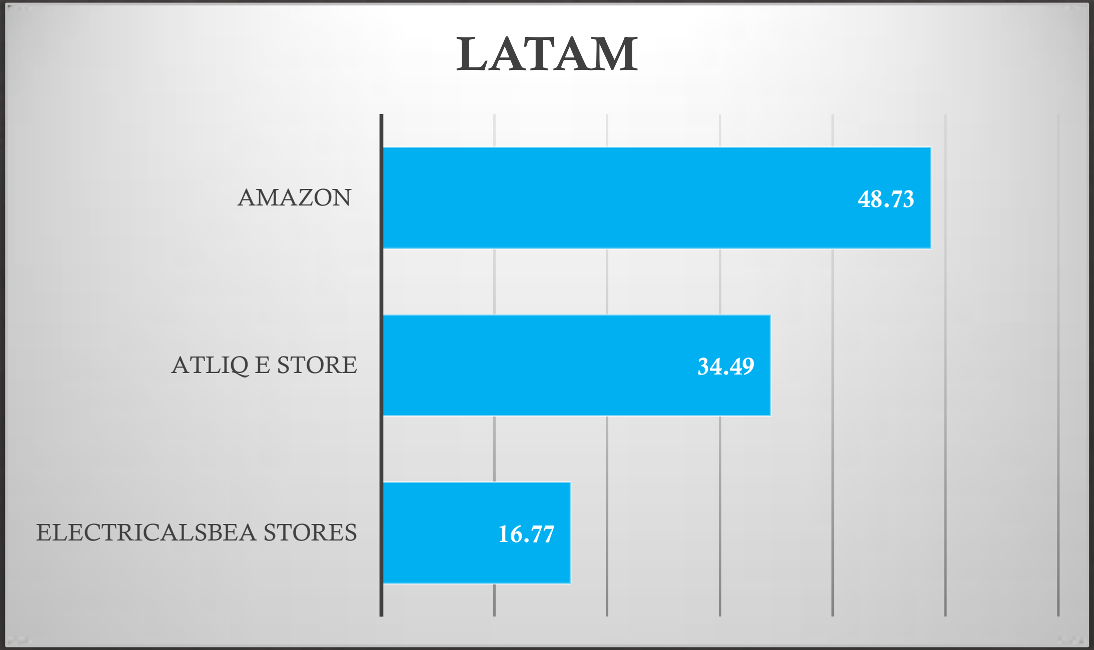
```
select * from net_sales;
with cte as (
select region,
customer,
round(sum(net_sales)/1000000,2) as net_sales_mln
from net_sales n
join dim_customer c using (customer_code)
where fiscal_year = 2021
group by customer, region
)
select *,
net_sales_mln*100/sum(net_sales_mln) over(partition by region) as pct_contribution
from cte order by region, net_sales_mln desc  ;
```

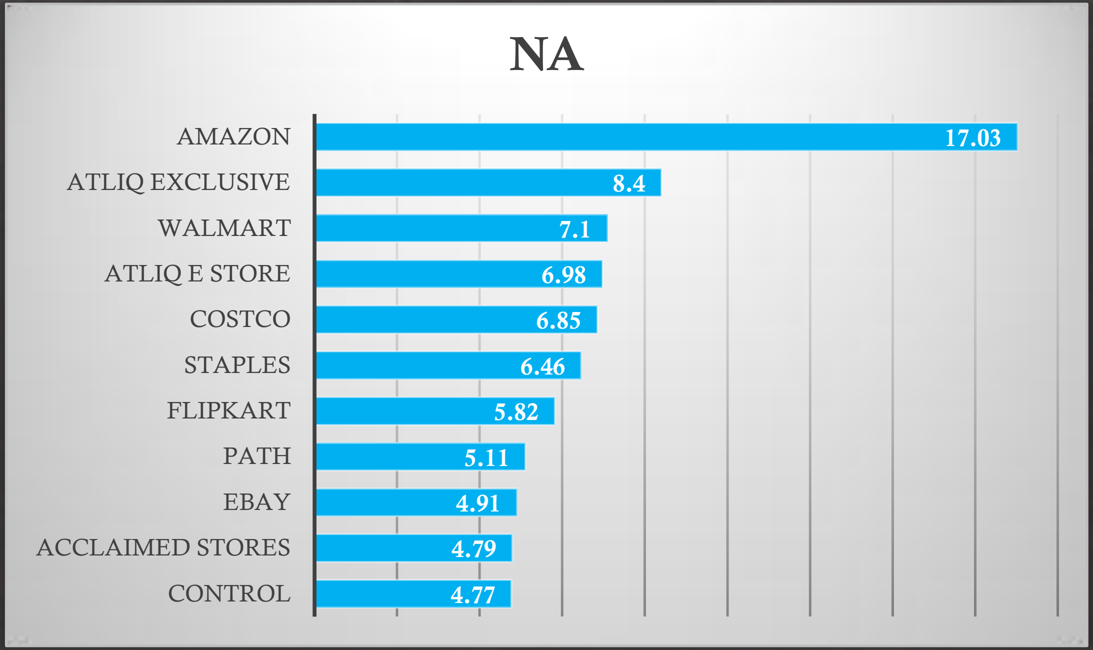


region	customer	net_sales_mln	pct_contribution
APAC	Amazon	57.41	12.988688
APAC	Atliq Exclusive	51.58	11.669683
APAC	Atliq e Store	36.97	8.364253
APAC	Leader	24.52	5.547511
APAC	Sage	22.85	5.169683
APAC	Neptune	21.01	4.753394
APAC	Electricalsocity	16.25	3.676471
APAC	Synthetic	14.14	3.199095
APAC	Propel	14.14	3.199095
APAC	Flipkart	12.96	2.932127
APAC	Novus	12.91	2.920814
APAC	Expression	12.90	2.918552
APAC	Girias	11.30	2.556561
APAC	Vijay Sales	11.27	2.549774
APAC	Ebay	11.14	2.520362


APAC
AMAZON 12.99
ATLIQ EXCLUSIVE 11.67
ATLIQ E STORE 8.36
LEADER 5.55
SAGE 5.17
NEPTUNE 4.75
ELECTRICALSOCITY 3.68
SYNTHETIC 3.2
PROPEL 3.2
FLIPKART 2.93

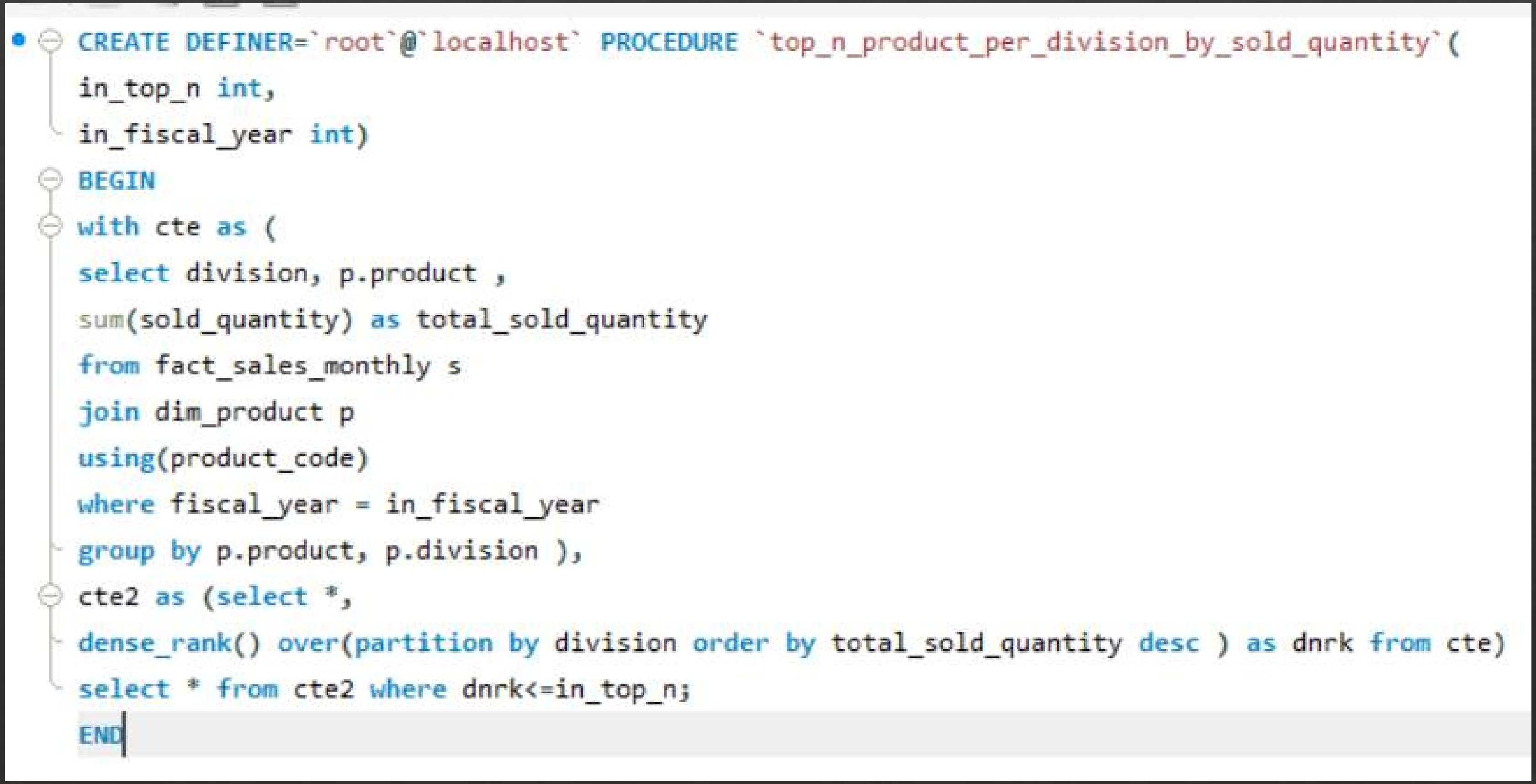

EU
ATLIQ E STORE 9.87
AMAZON 9.84
ATLIQ EXCLUSIVE 6.67
UNIEURO 4.8
EXPERT 4.17
CHIP 7 3.6
RADIO POPULAR 3.46
MEDIA MARKT 3.43
ELKJÃ¸P 3.37
SOREFOZ 3.05
COOLBLUE 2.78


LATAM
AMAZON 48.73
ATLIQ E STORE 34.49
ELECTRICALSBEA STORES 16.77

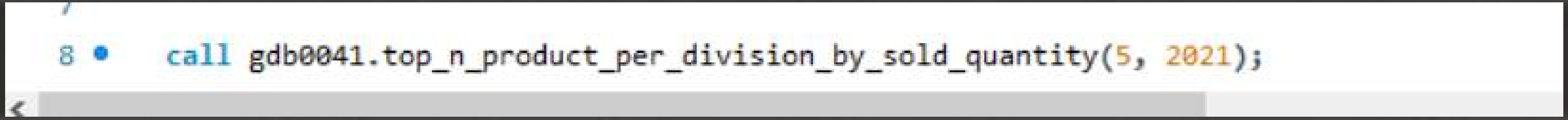

NA
AMAZON 17.03
ATLIQ EXCLUSIVE 8.4
WALMART 7.1
ATLIQ E STORE 6.98
COSTCO 6.85
STAPLES 6.46
FLIPKART 5.82
PATH 5.11
EBAY 4.91
ACCLAIMED STORES 4.79
CONTROL 4.77

8. Top n products in each division by their quantity sold

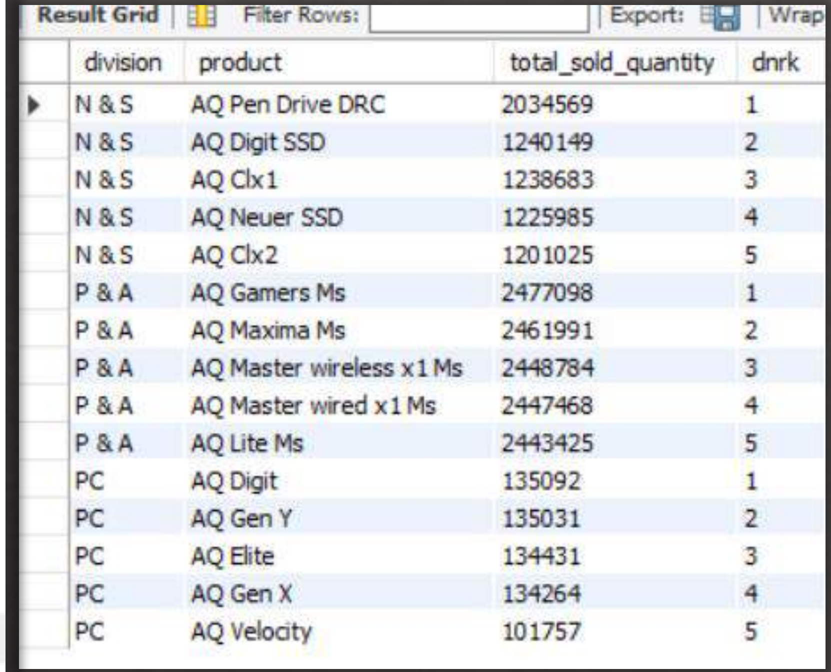
Details: create stored procedure for getting top n product in each division by their quantity sold in a given financial year



```
CREATE DEFINER=`root`@`localhost` PROCEDURE `top_n_product_per_division_by_sold_quantity`(
in_top_n int,
in_fiscal_year int)
BEGIN
with cte as (
select division, p.product ,
sum(sold_quantity) as total_sold_quantity
from fact_sales_monthly s
join dim_product p
using(product_code)
where fiscal_year = in_fiscal_year
group by p.product, p.division ),
cte2 as (select *,
dense_rank() over(partition by division order by total_sold_quantity desc ) as dnrk from cte)
select * from cte2 where dnrk<=in_top_n;
END
```



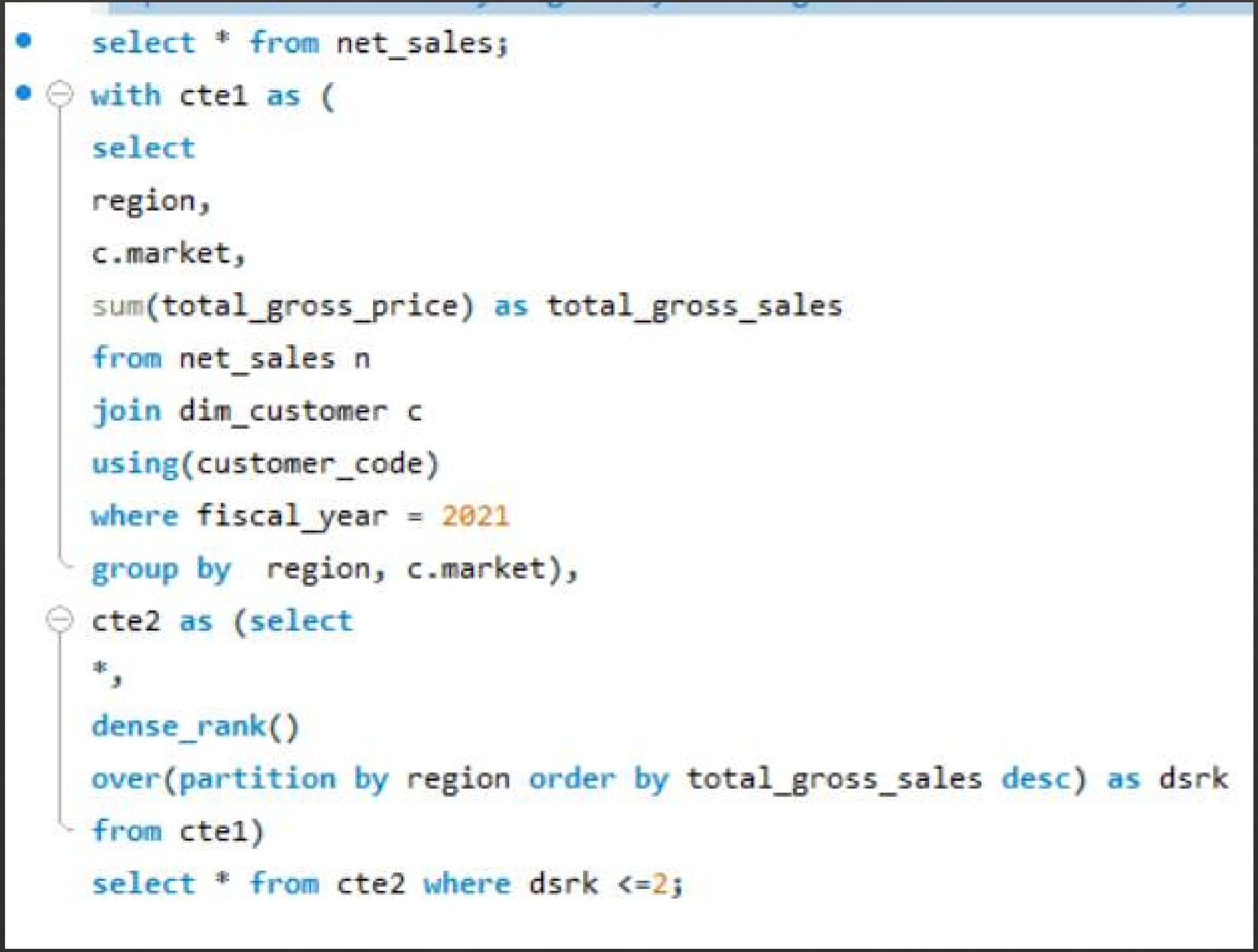
```
call gdb0041.top_n_product_per_division_by_sold_quantity(5, 2021);
```



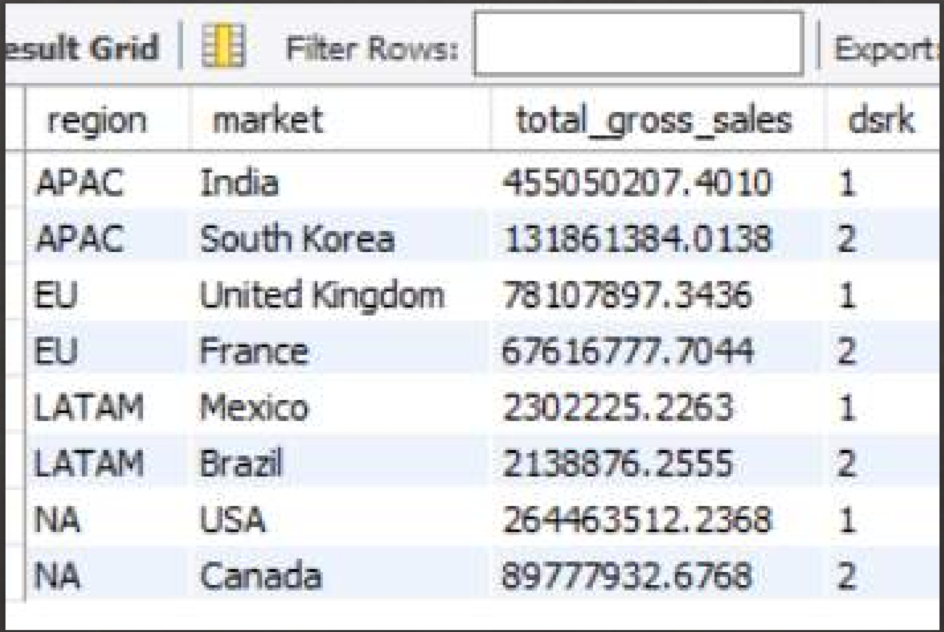
Result Grid | Filter Rows: | Export: | Wrap

division	product	total_sold_quantity	dnrk
N & S	AQ Pen Drive DRC	2034569	1
N & S	AQ Digit SSD	1240149	2
N & S	AQ Clx1	1238683	3
N & S	AQ Neuer SSD	1225985	4
N & S	AQ Clx2	1201025	5
P & A	AQ Gamers Ms	2477098	1
P & A	AQ Maxima Ms	2461991	2
P & A	AQ Master wireless x1 Ms	2448784	3
P & A	AQ Master wired x1 Ms	2447468	4
P & A	AQ Lite Ms	2443425	5
PC	AQ Digit	135092	1
PC	AQ Gen Y	135031	2
PC	AQ Elite	134431	3
PC	AQ Gen X	134264	4
PC	AQ Velocity	101757	5

9. Top two markets in every region by their gross sales in fiscal year 2021



```
select * from net_sales;
with cte1 as (
select
region,
c.market,
sum(total_gross_price) as total_gross_sales
from net_sales n
join dim_customer c
using(customer_code)
where fiscal_year = 2021
group by  region, c.market),
cte2 as (select
*,
dense_rank()
over(partition by region order by total_gross_sales desc) as dsrk
from cte1)
select * from cte2 where dsrk <=2;
```



esult Grid | Filter Rows: | Export:

region	market	total_gross_sales	dsrk
APAC	India	455050207.4010	1
APAC	South Korea	131861384.0138	2
EU	United Kingdom	78107897.3436	1
EU	France	67616777.7044	2
LATAM	Mexico	2302225.2263	1
LATAM	Brazil	2138876.2555	2
NA	USA	264463512.2368	1
NA	Canada	89777932.6768	2

10. Forecast accuracy for all customers for a given fiscal year

Details: aggregate forecast accuracy report for all the customers for a given fiscal year to track accuracy of forecast. Report should have following columns

-- customer code

-- customer name

-- market

-- total sold quantity

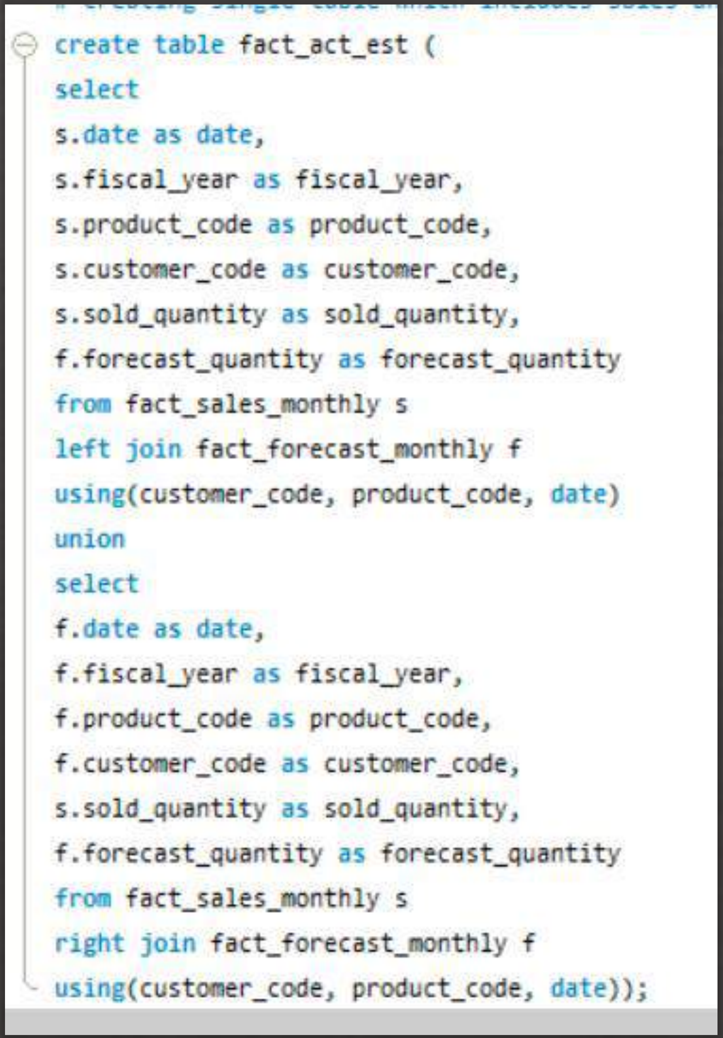
-- total forecast quantity

-- net error

-- absolute error

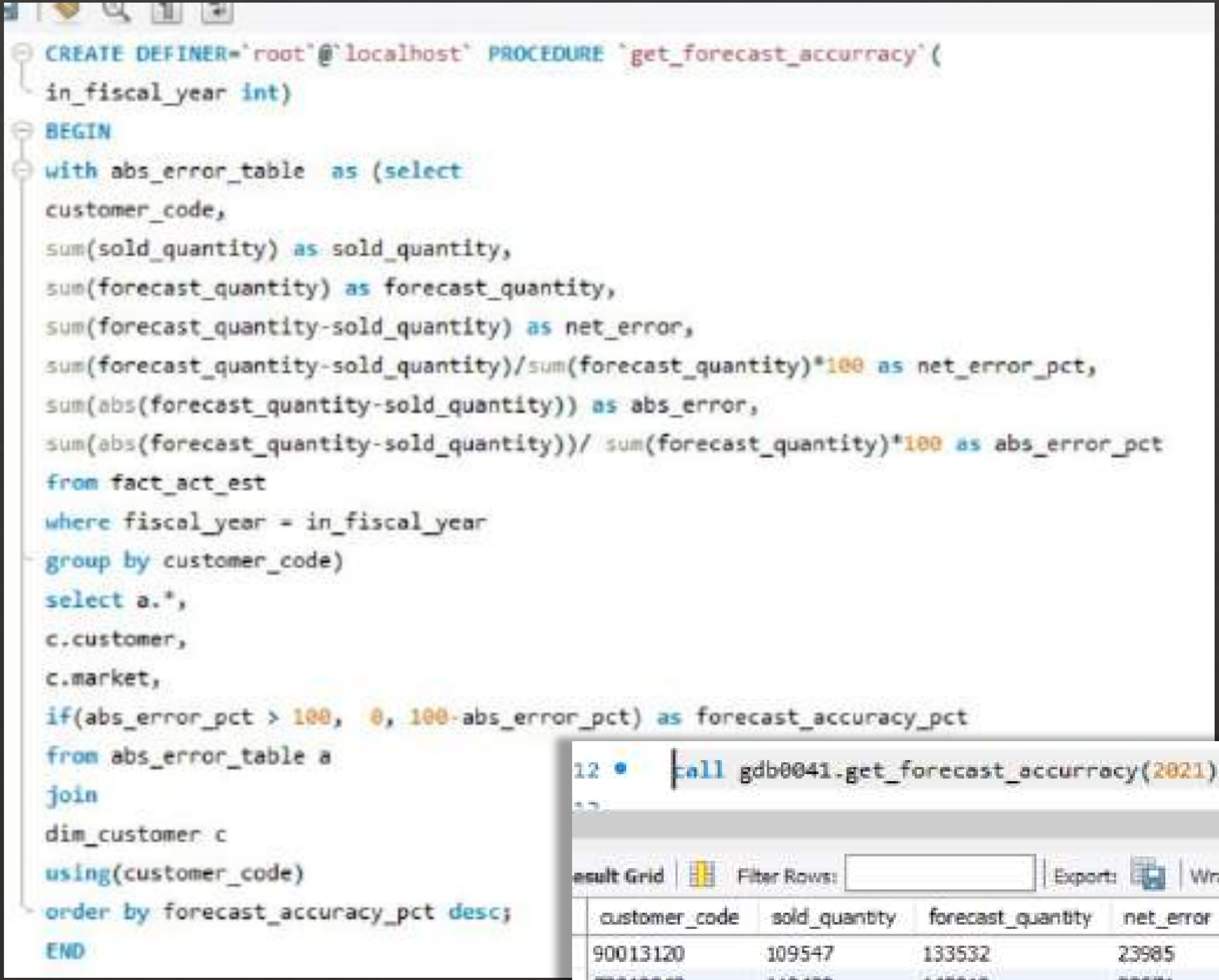
-- forecast accuracy

Created single table which includes sales and forecast quantity to make query simple and created triggers to automatically update new records from fact_sales_monthly and fact_forecat_monthly to fact_act_est

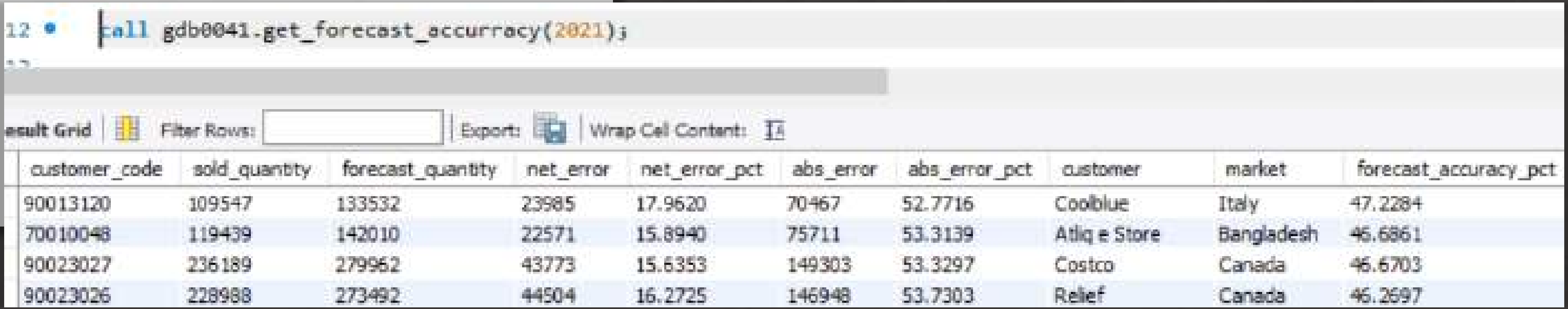


```
create table fact_act_est (
select
s.date as date,
s.fiscal_year as fiscal_year,
s.product_code as product_code,
s.customer_code as customer_code,
s.sold_quantity as sold_quantity,
f.forecast_quantity as forecast_quantity
from fact_sales_monthly s
left join fact_forecast_monthly f
using(customer_code, product_code, date)
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_sales_monthly s
right join fact_forecast_monthly f
using(customer_code, product_code, date));
```

Forecast accuracy for given year



```sql
CREATE DEFINER=`root`@`localhost` PROCEDURE `get_forecast_accurracy`(
in_fiscal_year int)
BEGIN
with abs_error_table  as (select
customer_code,
sum(sold_quantity) as sold_quantity,
sum(forecast_quantity) as forecast_quantity,
sum(forecast_quantity-sold_quantity) as net_error,
sum(forecast_quantity-sold_quantity)/sum(forecast_quantity)*100 as net_error_pct,
sum(abs(forecast_quantity-sold_quantity)) as abs_error,
sum(abs(forecast_quantity-sold_quantity))/ sum(forecast_quantity)*100 as abs_error_pct
from fact_act_est
where fiscal_year = in_fiscal_year
group by customer_code)
select a.*,
c.customer,
c.market,
if(abs_error_pct > 100,  0, 100-abs_error_pct) as forecast_accuracy_pct
from abs_error_table a
join
dim_customer c
using(customer_code)
order by forecast_accuracy_pct desc;
END
```



```sql
call gdb0041.get_forecast_accurracy(2021);
```

customer_code	sold_quantity	forecast_quantity	net_error	net_error_pct	abs_error	abs_error_pct	customer	market	forecast_accuracy_pct
90013120	109547	133532	23985	17.9620	70467	52.7716	Coolblue	Italy	47.2284
70010048	119439	142010	22571	15.8940	75711	53.3139	Atliq e Store	Bangladesh	46.6861
90023027	236189	279962	43773	15.6353	149303	53.3297	Costco	Canada	46.6703
90023026	228988	273492	44504	16.2725	146948	53.7303	Relief	Canada	46.2697

11. Which customers forecast_accuracy has dropped from 2020 to 2021

with columns

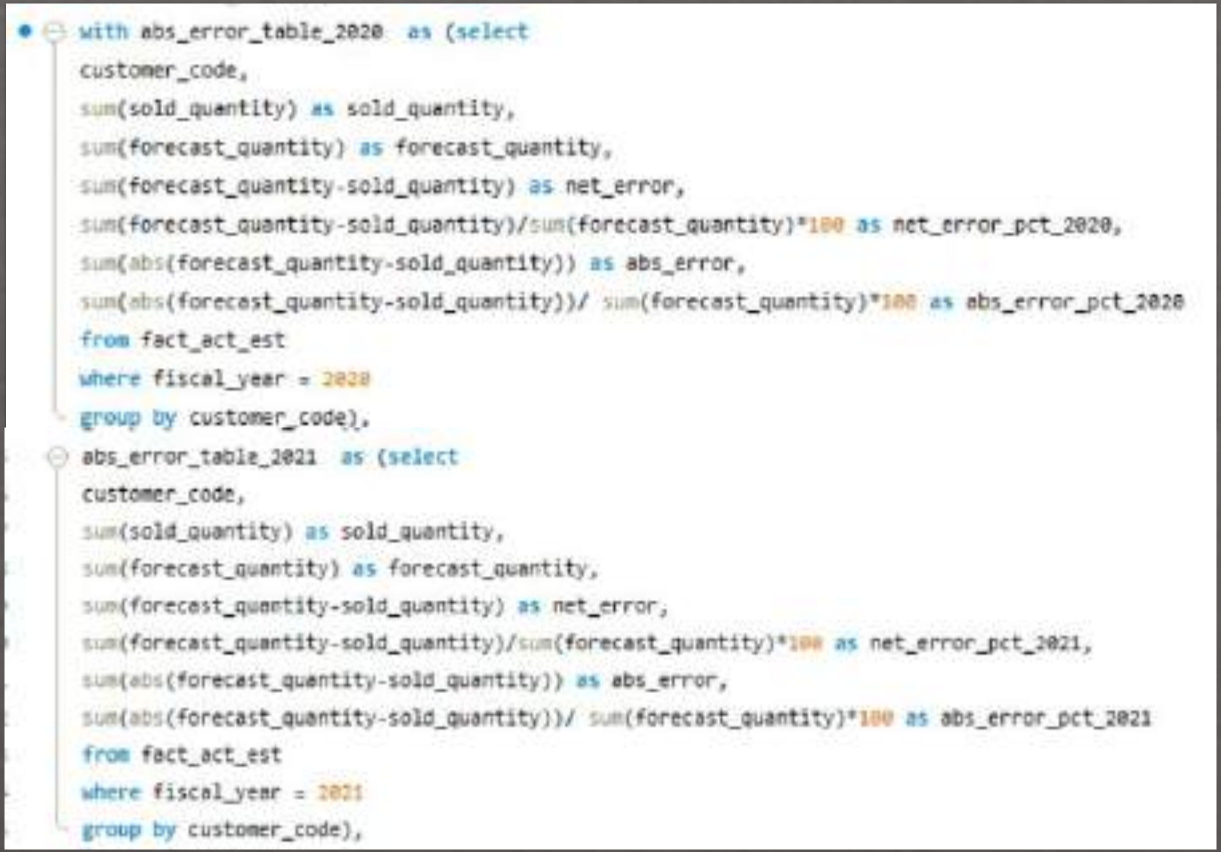
-- customer code

-- customer name

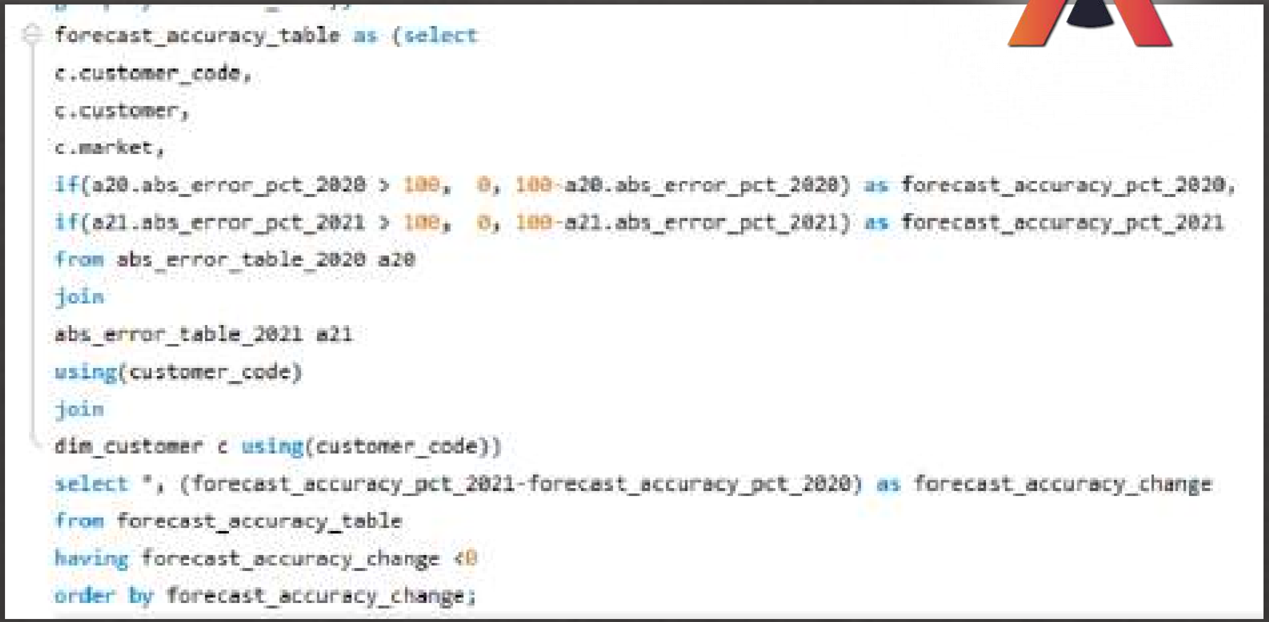
-- market

-- forecast_accuracy 2020

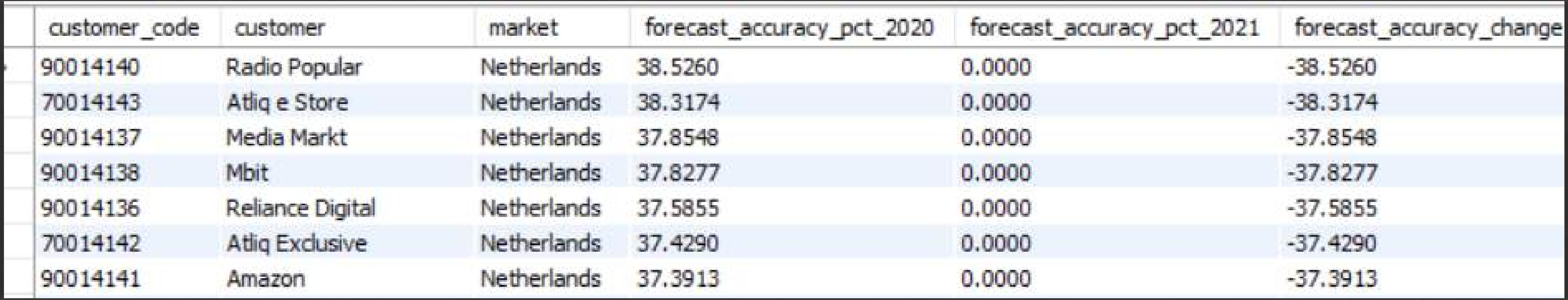
-- forecast_accuracy 2021



```
with abs_error_table_2020  as (select
    customer_code,
    sum(sold_quantity) as sold_quantity,
    sum(forecast_quantity) as forecast_quantity,
    sum(forecast_quantity-sold_quantity) as net_error,
    sum(forecast_quantity-sold_quantity)/sum(forecast_quantity)*100 as net_error_pct_2020,
    sum(abs(forecast_quantity-sold_quantity)) as abs_error,
    sum(abs(forecast_quantity-sold_quantity))/ sum(forecast_quantity)*100 as abs_error_pct_2020
    from fact_act_est
    where fiscal_year = 2020
    group by customer_code),
abs_error_table_2021  as (select
    customer_code,
    sum(sold_quantity) as sold_quantity,
    sum(forecast_quantity) as forecast_quantity,
    sum(forecast_quantity-sold_quantity) as net_error,
    sum(forecast_quantity-sold_quantity)/sum(forecast_quantity)*100 as net_error_pct_2021,
    sum(abs(forecast_quantity-sold_quantity)) as abs_error,
    sum(abs(forecast_quantity-sold_quantity))/ sum(forecast_quantity)*100 as abs_error_pct_2021
    from fact_act_est
    where fiscal_year = 2021
    group by customer_code),
```



```
forecast_accuracy_table as (select
c.customer_code,
c.customer,
c.market,
if(a20.abs_error_pct_2020 > 100,  0, 100-a20.abs_error_pct_2020) as forecast_accuracy_pct_2020,
if(a21.abs_error_pct_2021 > 100,  0, 100-a21.abs_error_pct_2021) as forecast_accuracy_pct_2021
from abs_error_table_2020 a20
join
abs_error_table_2021 a21
using(customer_code)
join
dim_customer c using(customer_code))
select *, (forecast_accuracy_pct_2021-forecast_accuracy_pct_2020) as forecast_accuracy_change
from forecast_accuracy_table
having forecast_accuracy_change <0
order by forecast_accuracy_change;
```



customer_code	customer	market	forecast_accuracy_pct_2020	forecast_accuracy_pct_2021	forecast_accuracy_change
90014140	Radio Popular	Netherlands	38.5260	0.0000	-38.5260
70014143	Atliq e Store	Netherlands	38.3174	0.0000	-38.3174
90014137	Media Markt	Netherlands	37.8548	0.0000	-37.8548
90014138	Mbit	Netherlands	37.8277	0.0000	-37.8277
90014136	Reliance Digital	Netherlands	37.5855	0.0000	-37.5855
70014142	Atliq Exclusive	Netherlands	37.4290	0.0000	-37.4290
90014141	Amazon	Netherlands	37.3913	0.0000	-37.3913

Performance optimization

- To analyse performance explain analyze, analyze were used
- To optimize performance additional tables, columns indexes were created.

Insights

- Netherlands forecast accuracy has dropped drastically in 2021
- Amazon has highest contribution percentage by net sales in all regions
- India is the biggest market with 210 million net sales
- Sales has drastically increased from 6.4 million in 2018 to 121.1 million in 2021 for Croma India

◈ Thanking You . . . :)