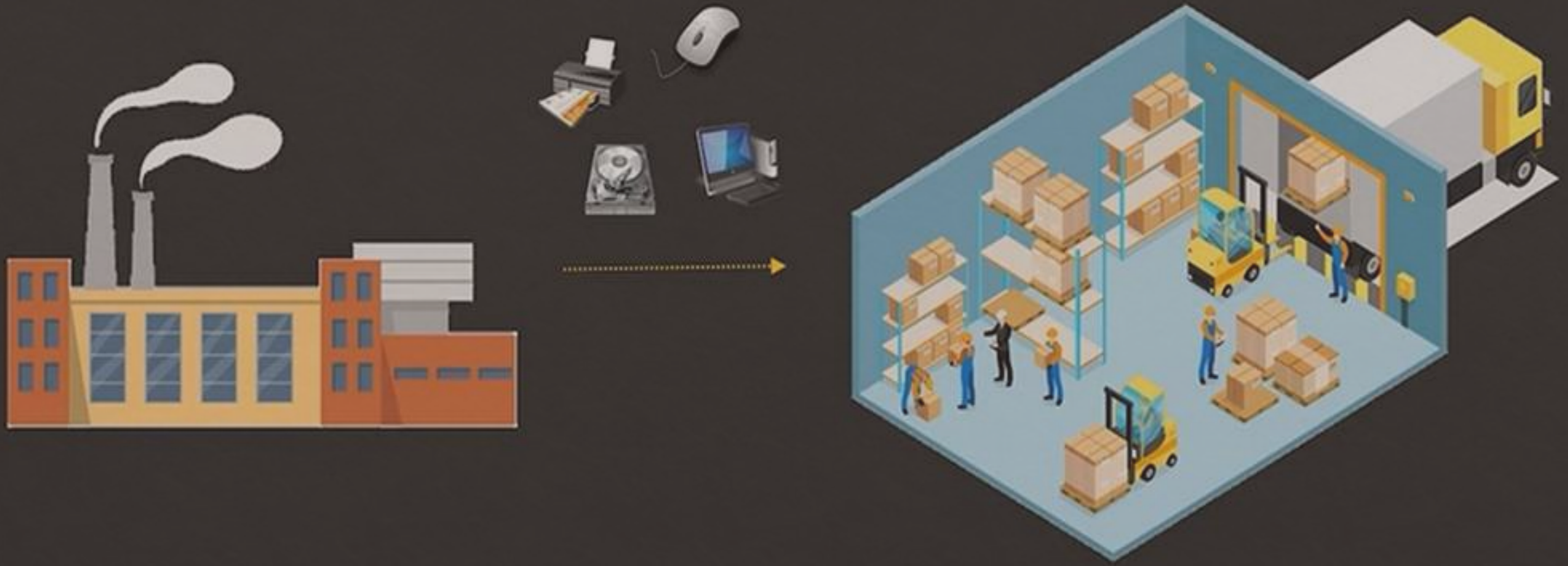
AD-HOC ANALYSIS

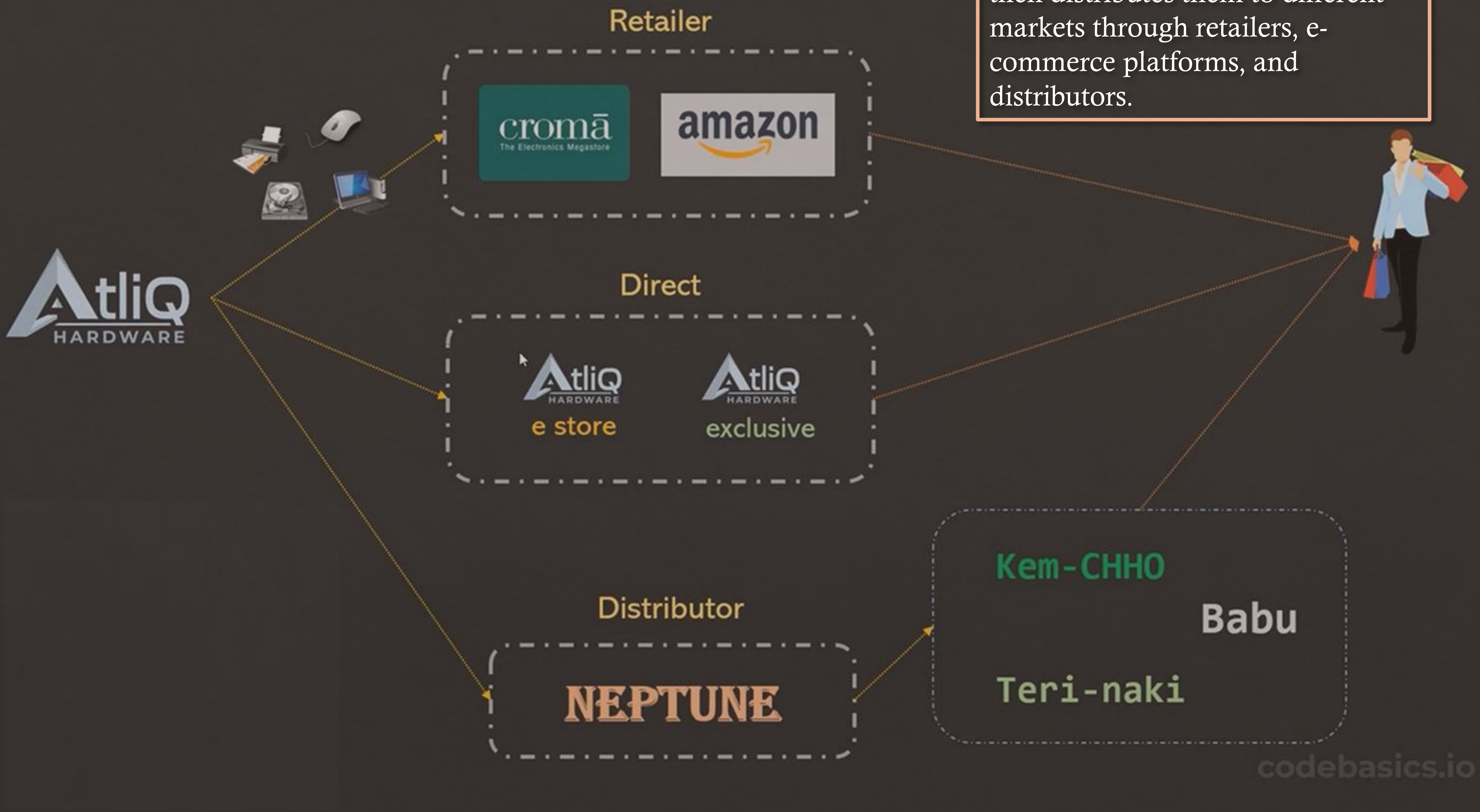
Solving Real Business Problems Using SQL

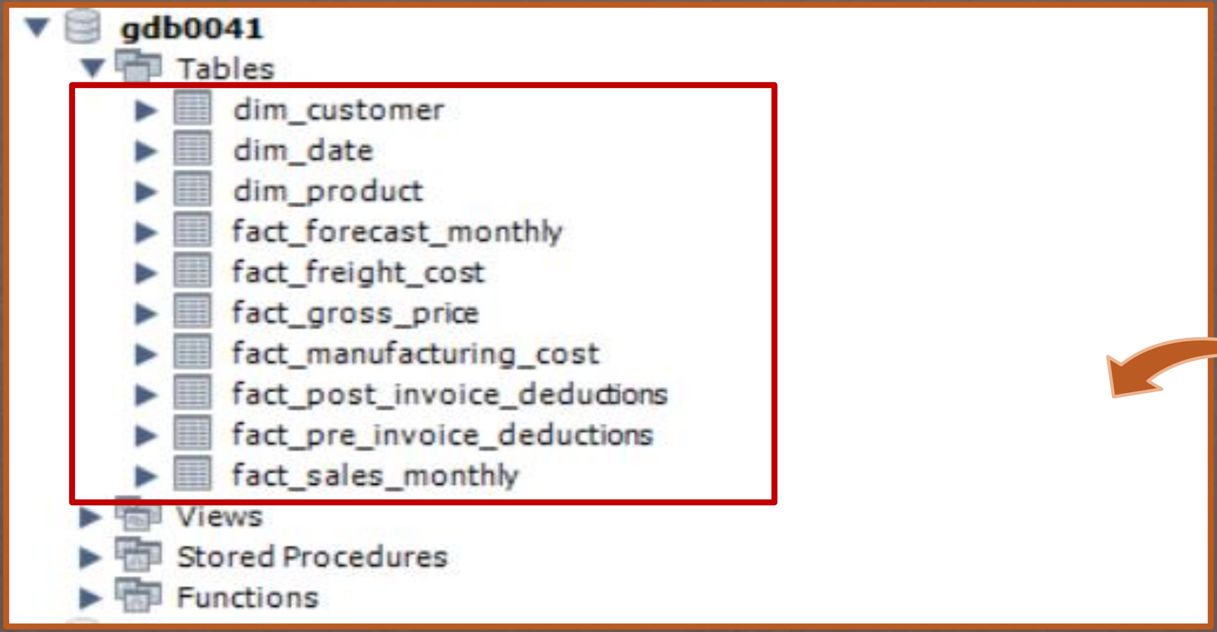
SQL

INSIGHTS FOR ATLIQ HARDWARE



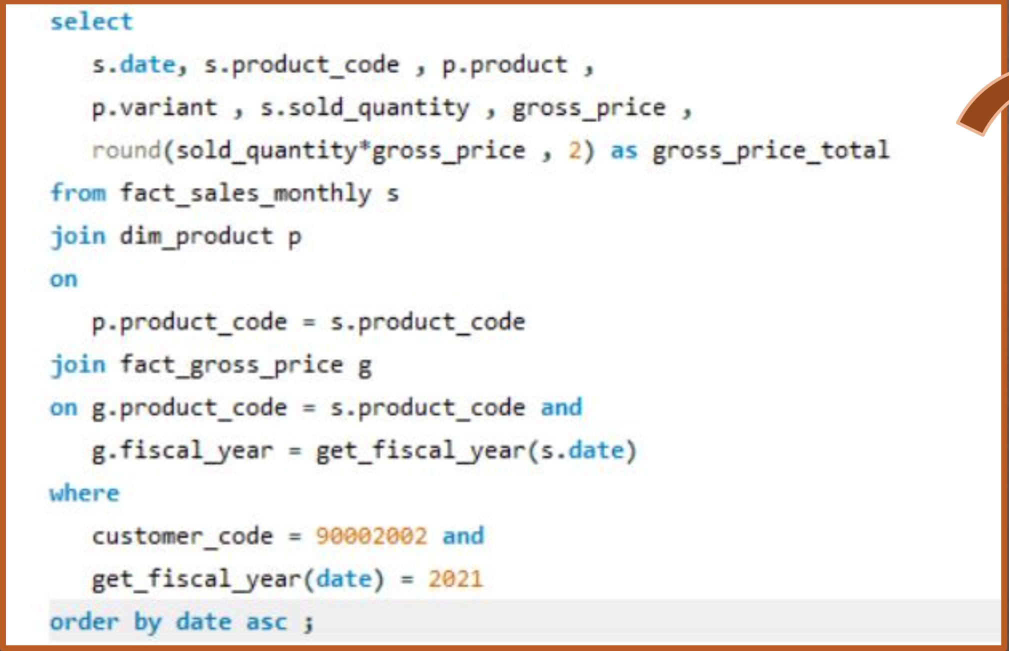
AtliQ manufactures products, stores them in a central warehouse then distributes them to different markets through retailers, e-commerce platforms, and distributors.



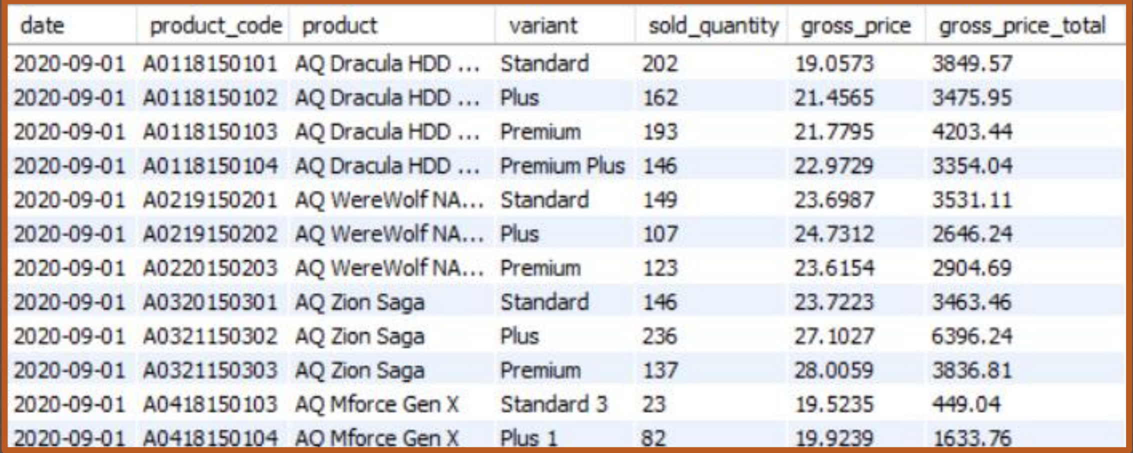

gdb0041
Tables
dim_customer
dim_date
dim_product
fact_forecast_monthly
fact_freight_cost
fact_gross_price
fact_manufacturing_cost
fact_post_invoice_deductions
fact_pre_invoice_deductions
fact_sales_monthly
Views
Stored Procedures
Functions

The dataset includes dim_customer, dim_product, dim_date, and multiple fact tables capturing sales, prices, costs, and deductions.

Problem Statement: Generate a monthly product-wise sales report for Croma India for FY 2021.
The report must include:
Month
Product Name
Variant
Sold Quantity
Gross Price Per Item
Gross Price Total

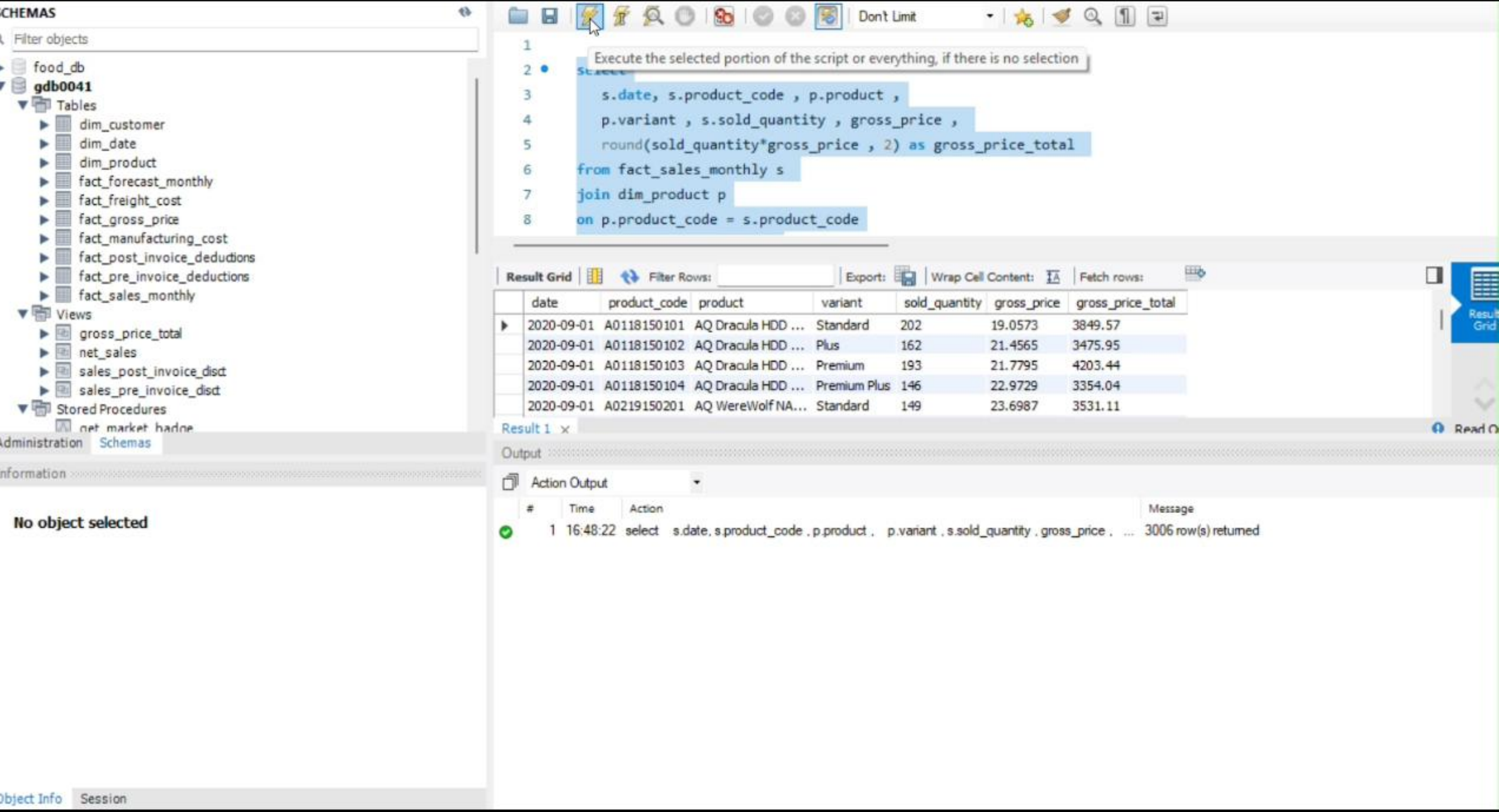


```
select
   s.date, s.product_code , p.product ,
   p.variant , s.sold_quantity , gross_price ,
   round(sold_quantity*gross_price , 2) as gross_price_total
from fact_sales_monthly s
join dim_product p
on
   p.product_code = s.product_code
join fact_gross_price g
on g.product_code = s.product_code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



date	product_code	product	variant	sold_quantity	gross_price	gross_price_total
2020-09-01	A0118150101	AQ Dracula HDD ...	Standard	202	19.0573	3849.57
2020-09-01	A0118150102	AQ Dracula HDD ...	Plus	162	21.4565	3475.95
2020-09-01	A0118150103	AQ Dracula HDD ...	Premium	193	21.7795	4203.44
2020-09-01	A0118150104	AQ Dracula HDD ...	Premium Plus	146	22.9729	3354.04
2020-09-01	A0219150201	AQ WereWolf NA...	Standard	149	23.6987	3531.11
2020-09-01	A0219150202	AQ WereWolf NA...	Plus	107	24.7312	2646.24
2020-09-01	A0220150203	AQ WereWolf NA...	Premium	123	23.6154	2904.69
2020-09-01	A0320150301	AQ Zion Saga	Standard	146	23.7223	3463.46
2020-09-01	A0321150302	AQ Zion Saga	Plus	236	27.1027	6396.24
2020-09-01	A0321150303	AQ Zion Saga	Premium	137	28.0059	3836.81
2020-09-01	A0418150103	AQ Mforce Gen X	Standard 3	23	19.5235	449.04
2020-09-01	A0418150104	AQ Mforce Gen X	Plus 1	82	19.9239	1633.76

Sample of result



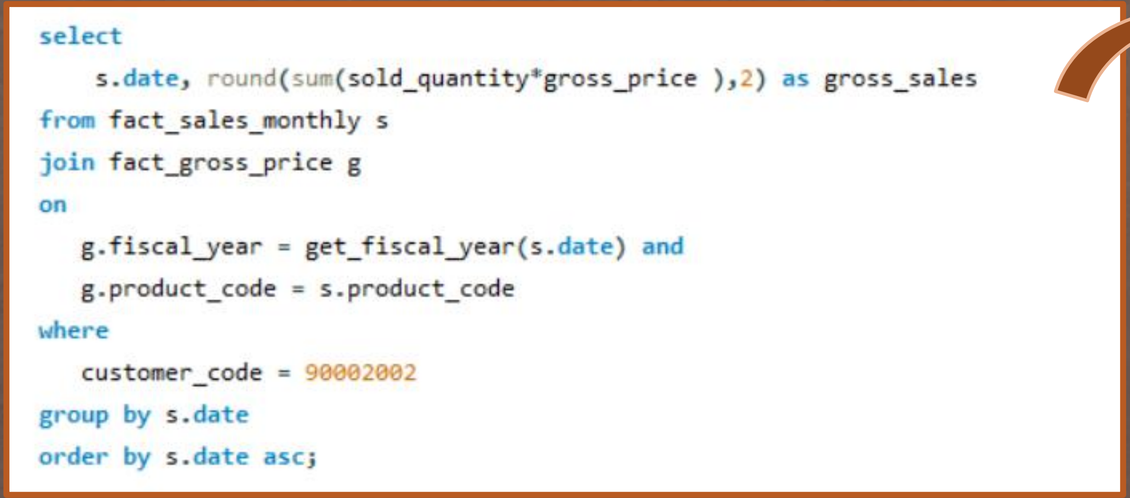
Execute the selected portion of the script or everything, if there is no selection

```
select
    s.date, s.product_code , p.product ,
    p.variant , s.sold_quantity , gross_price ,
    round(sold_quantity*gross_price , 2) as gross_price_total
from fact_sales_monthly s
join dim_product p
on p.product_code = s.product_code
```

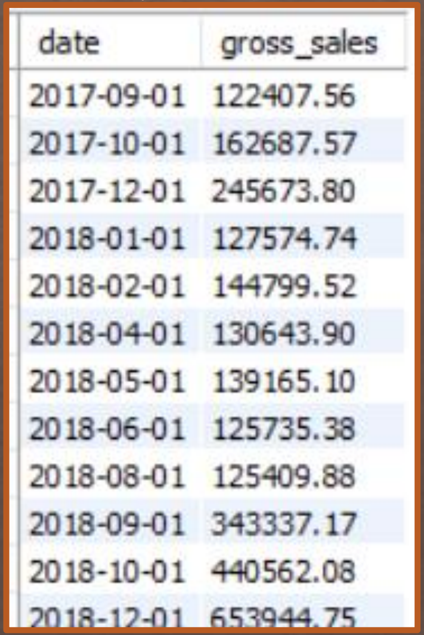
date	product_code	product	variant	sold_quantity	gross_price	gross_price_total
2020-09-01	A0118150101	AQ Dracula HDD ...	Standard	202	19.0573	3849.57
2020-09-01	A0118150102	AQ Dracula HDD ...	Plus	162	21.4565	3475.95
2020-09-01	A0118150103	AQ Dracula HDD ...	Premium	193	21.7795	4203.44
2020-09-01	A0118150104	AQ Dracula HDD ...	Premium Plus	146	22.9729	3354.04
2020-09-01	A0219150201	AQ WereWolf NA...	Standard	149	23.6987	3531.11

#	Time	Action	Message
1	16:48:22	select s.date, s.product_code , p.product , p.variant , s.sold_quantity , gross_price , ...	3006 row(s) returned

Generate an aggregate monthly gross sales report for the Croma India customer.
The report should include:
Month
Total gross sales amount to Croma India in that month



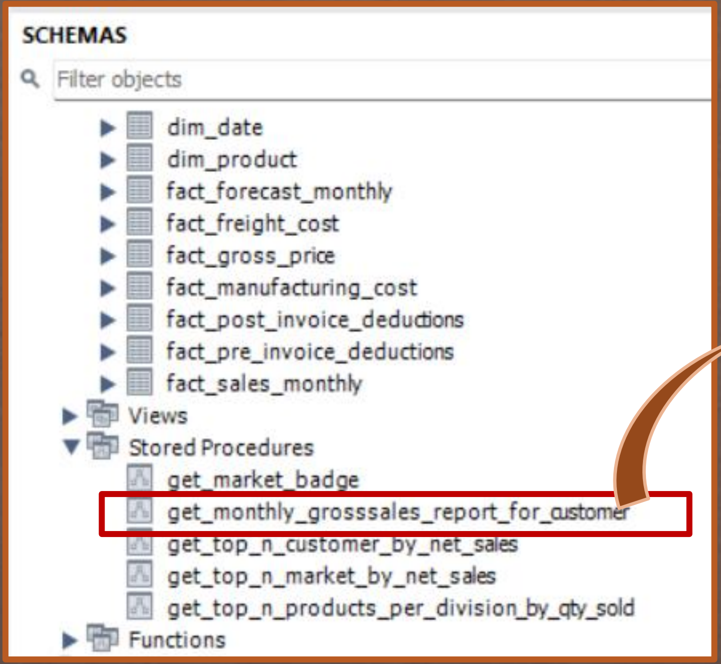
```
select
    s.date, round(sum(sold_quantity*gross_price ),2) as gross_sales
from fact_sales_monthly s
join fact_gross_price g
on
   g.fiscal_year = get_fiscal_year(s.date) and
   g.product_code = s.product_code
where
   customer_code = 90002002
group by s.date
order by s.date asc;
```



date	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
2018-10-01	440562.08
2018-12-01	653944.75

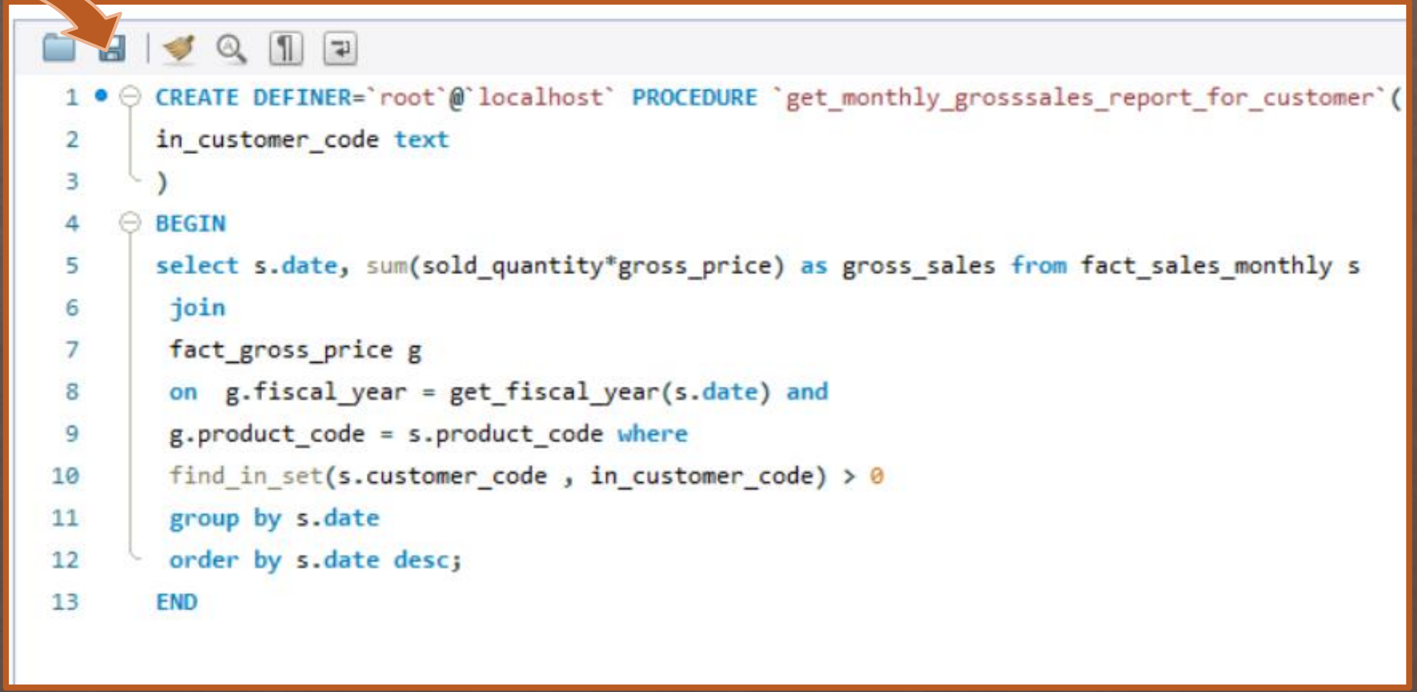
Sample of result

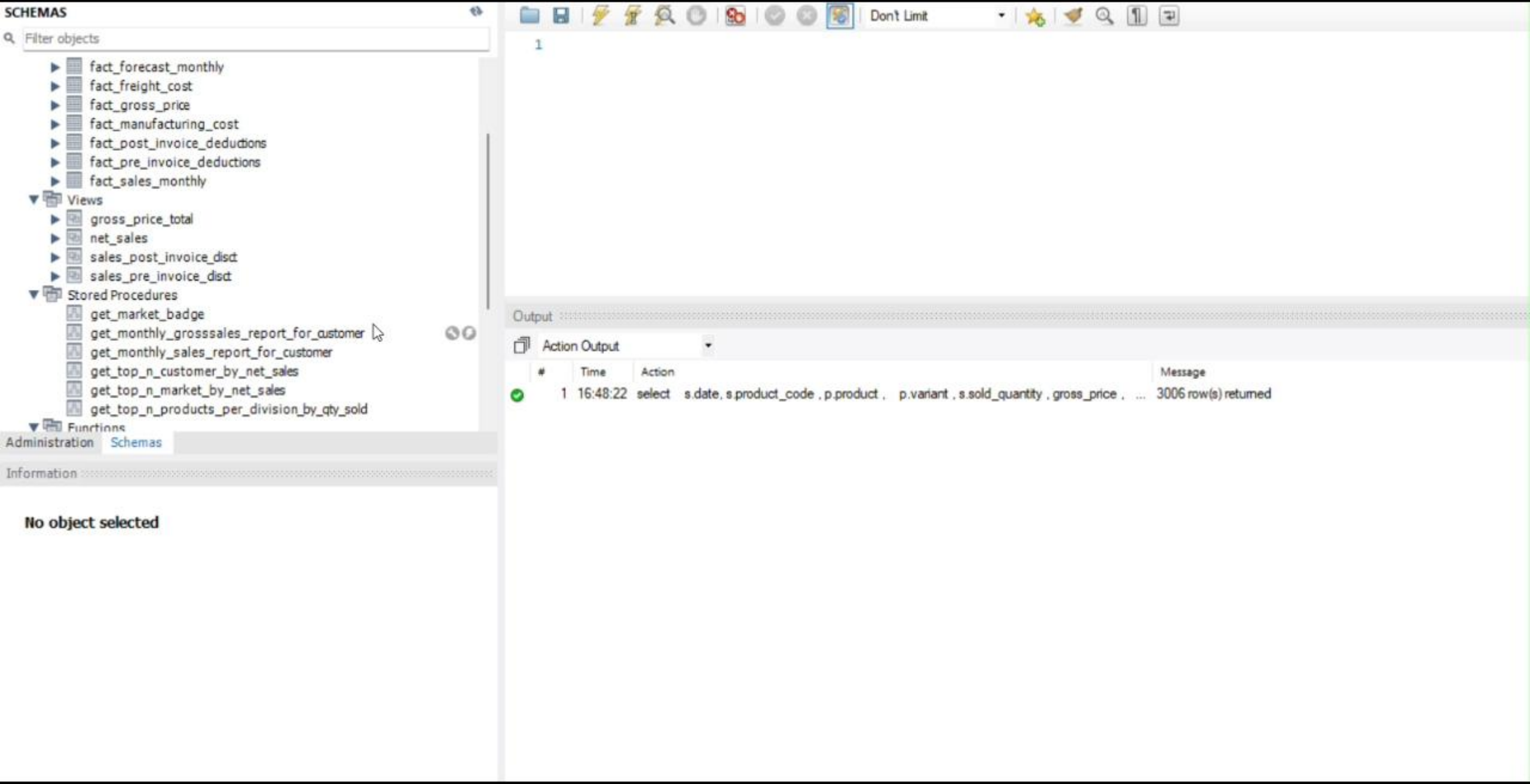
Built a reusable stored procedure to fetch monthly aggregated gross sales for any selected customer


SCHEMAS

Filter objects

- dim_date
- dim_product
- fact_forecast_monthly
- fact_freight_cost
- fact_gross_price
- fact_manufacturing_cost
- fact_post_invoice_deductions
- fact_pre_invoice_deductions
- fact_sales_monthly
- Views
- Stored Procedures
  - get_market_badge
  - get_monthly_grosssales_report_for_customer
  - get_top_n_customer_by_net_sales
  - get_top_n_market_by_net_sales
  - get_top_n_products_per_division_by_qty_sold
- Functions


```
CREATE DEFINER=`root`@`localhost` PROCEDURE `get_monthly_grosssales_report_for_customer`(
in_customer_code text
)
BEGIN
select s.date, sum(sold_quantity*gross_price) as gross_sales from fact_sales_monthly s
 join
 fact_gross_price g
 on  g.fiscal_year = get_fiscal_year(s.date) and
 g.product_code = s.product_code where
 find_in_set(s.customer_code , in_customer_code) > 0
 group by s.date
 order by s.date desc;
END
```



SCHEMAS

Filter objects

- fact_forecast_monthly
- fact_freight_cost
- fact_gross_price
- fact_manufacturing_cost
- fact_post_invoice_deductions
- fact_pre_invoice_deductions
- fact_sales_monthly
- Views
  - gross_price_total
  - net_sales
  - sales_post_invoice_disct
  - sales_pre_invoice_disct
- Stored Procedures
  - get_market_badge
  - get_monthly_grosssales_report_for_customer
  - get_monthly_sales_report_for_customer
  - get_top_n_customer_by_net_sales
  - get_top_n_market_by_net_sales
  - get_top_n_products_per_division_by_qty_sold
- Functions

Administration Schemas

Information

No object selected

Don't Limit

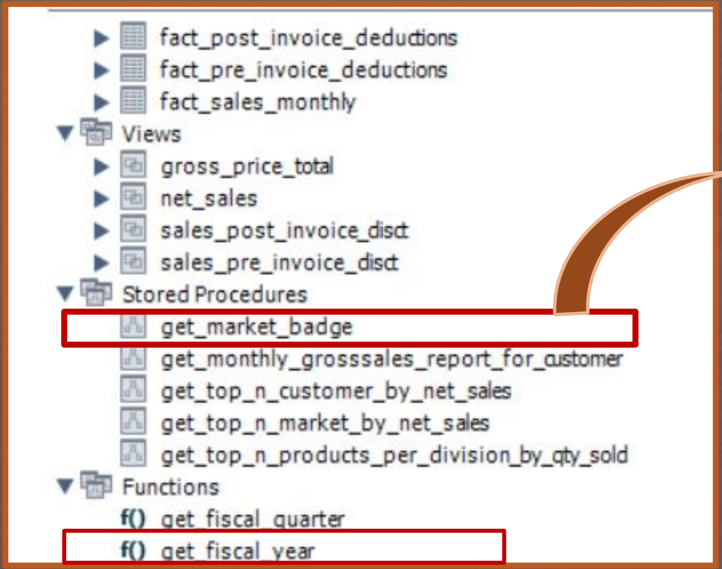
Output

Action Output

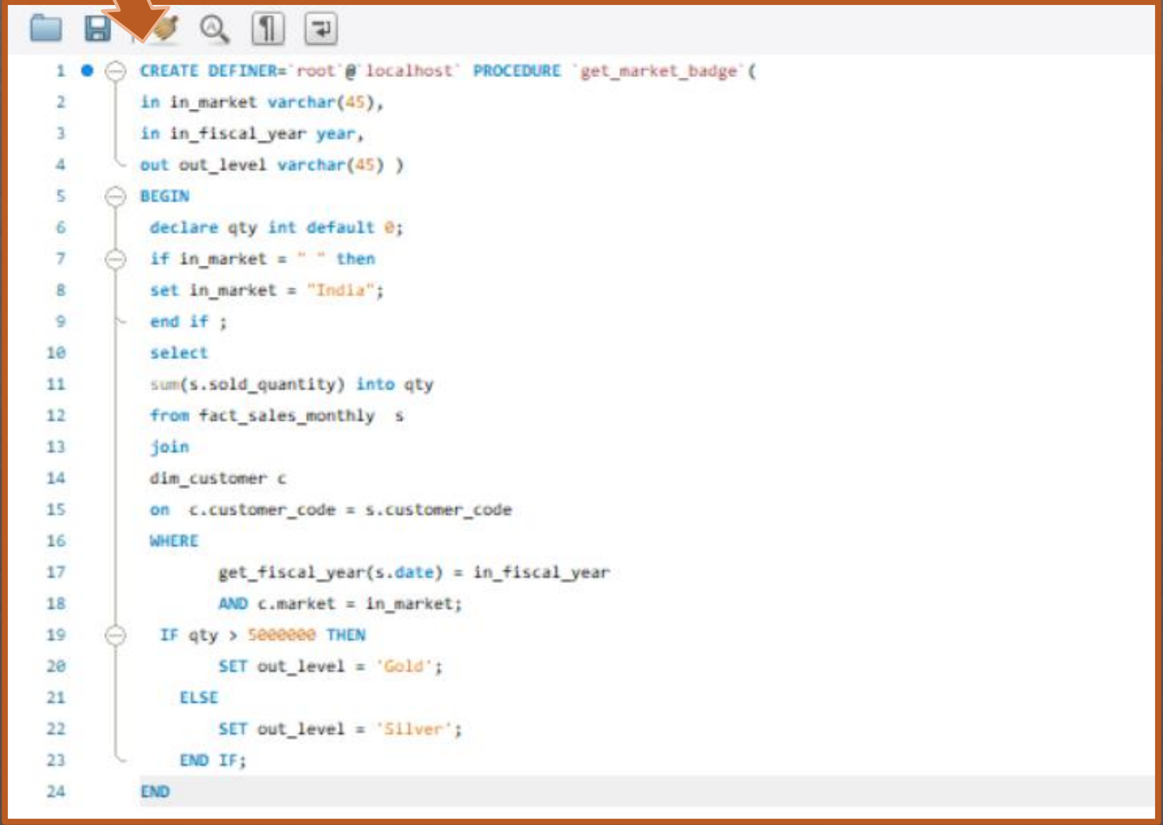
```
#  Time      Action                                                                              Message
1  16:48:22  select  s.date, s.product_code , p.product ,  p.variant , s.sold_quantity , gross_price ,  ...  3006 row(s) returned
```

Create a stored procedure that determines the market badge (Gold or Silver) based on total sold quantity for a given market and fiscal year.

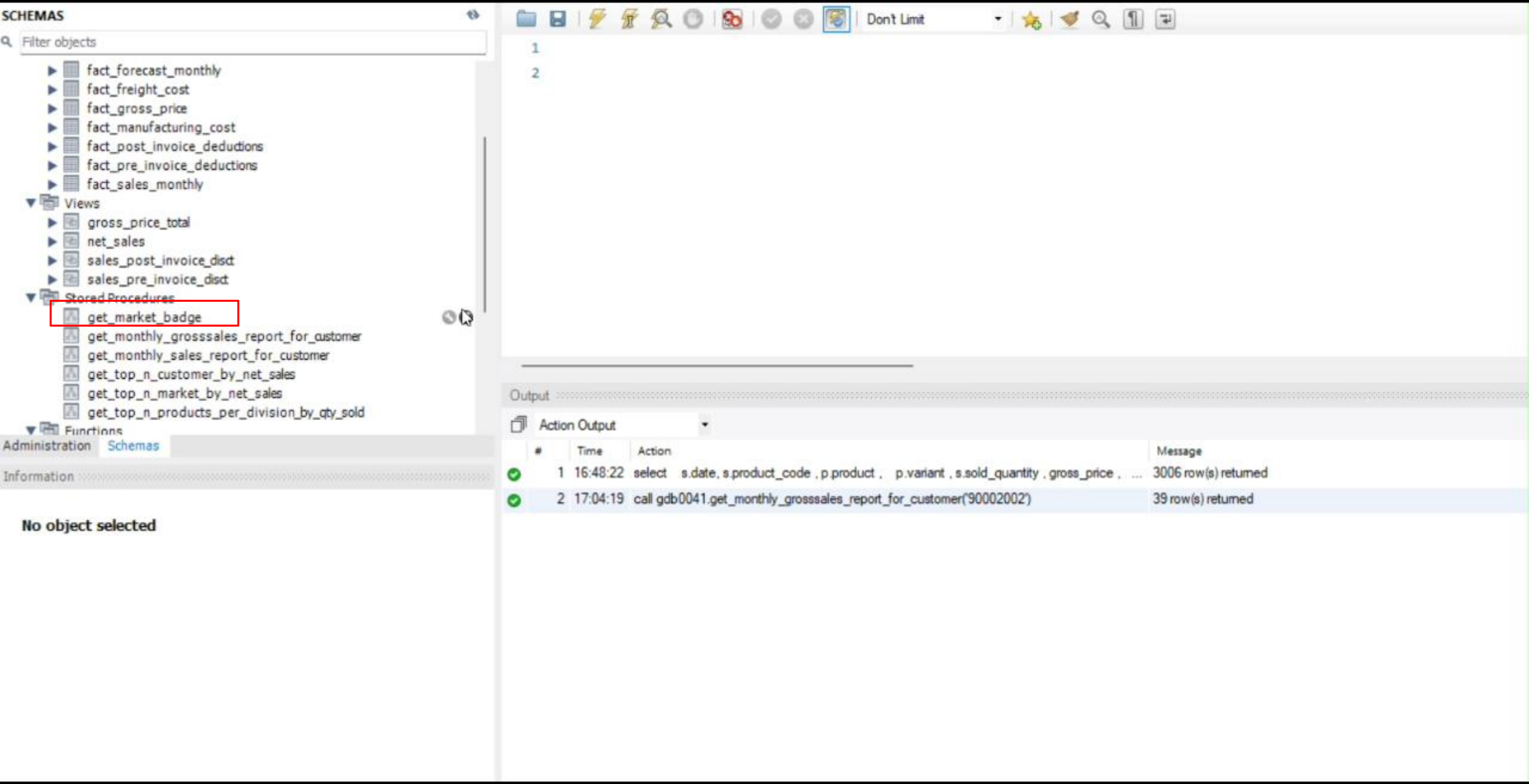
Logic Given
If total sold quantity > 5 million → Market Badge = Gold
Else → Market Badge = Silver



```
▶ fact_post_invoice_deductions
▶ fact_pre_invoice_deductions
▶ fact_sales_monthly
▼ Views
  ▶ gross_price_total
  ▶ net_sales
  ▶ sales_post_invoice_disct
  ▶ sales_pre_invoice_disct
▼ Stored Procedures
    get_market_badge
    get_monthly_grosssales_report_for_customer
    get_top_n_customer_by_net_sales
    get_top_n_market_by_net_sales
    get_top_n_products_per_division_by_qty_sold
▼ Functions
    f() get_fiscal_quarter
    f() get_fiscal_year
```



```
CREATE DEFINER=`root`@`localhost` PROCEDURE `get_market_badge`(
in in_market varchar(45),
in in_fiscal_year year,
out out_level varchar(45) )
BEGIN
 declare qty int default 0;
 if in_market = " " then
 set in_market = "India";
 end if ;
 select
 sum(s.sold_quantity) into qty
 from fact_sales_monthly  s
 join
 dim_customer c
 on  c.customer_code = s.customer_code
 WHERE
        get_fiscal_year(s.date) = in_fiscal_year
        AND c.market = in_market;
  IF qty > 5000000 THEN
        SET out_level = 'Gold';
    ELSE
        SET out_level = 'Silver';
    END IF;
END
```

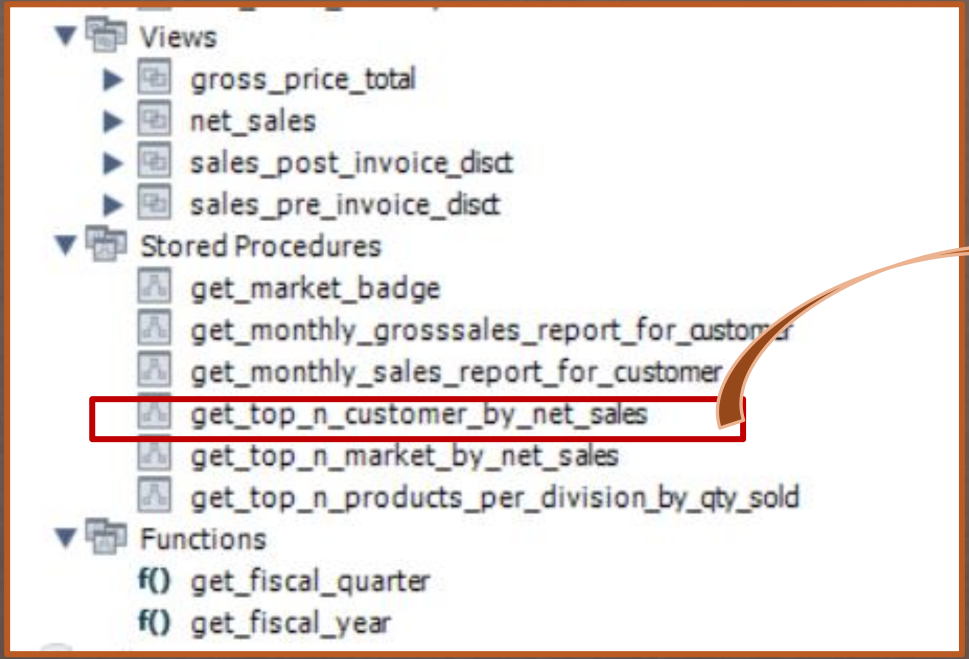


```
fact_forecast_monthly
fact_freight_cost
fact_gross_price
fact_manufacturing_cost
fact_post_invoice_deductions
fact_pre_invoice_deductions
fact_sales_monthly
Views
  gross_price_total
  net_sales
  sales_post_invoice_disct
  sales_pre_invoice_disct
Stored Procedures
  get_market_badge
  get_monthly_grosssales_report_for_customer
  get_monthly_sales_report_for_customer
  get_top_n_customer_by_net_sales
  get_top_n_market_by_net_sales
  get_top_n_products_per_division_by_qty_sold
Functions
```

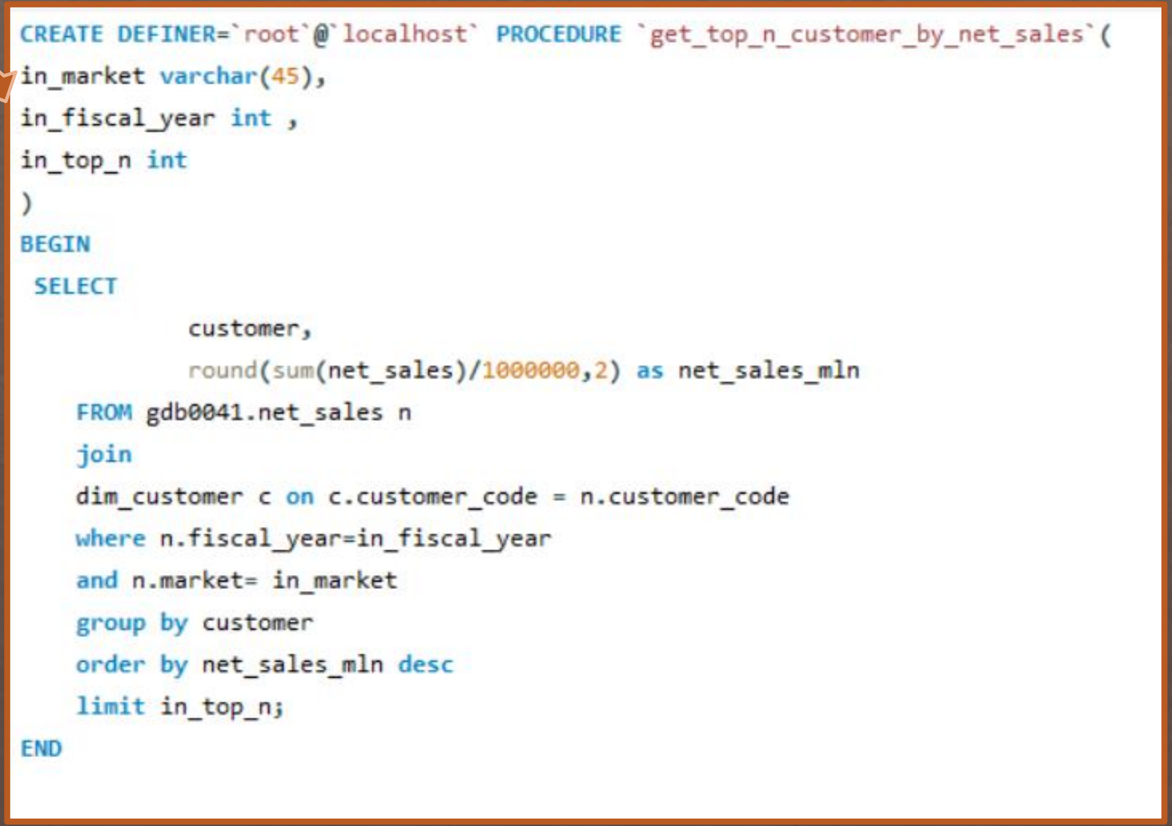
No object selected

```
#  Time      Action                                                                              Message
1  16:48:22  select  s.date, s.product_code , p.product ,  p.variant , s.sold_quantity , gross_price , ...  3006 row(s) returned
2  17:04:19  call gdb0041.get_monthly_grosssales_report_for_customer('90002002')                  39 row(s) returned
```

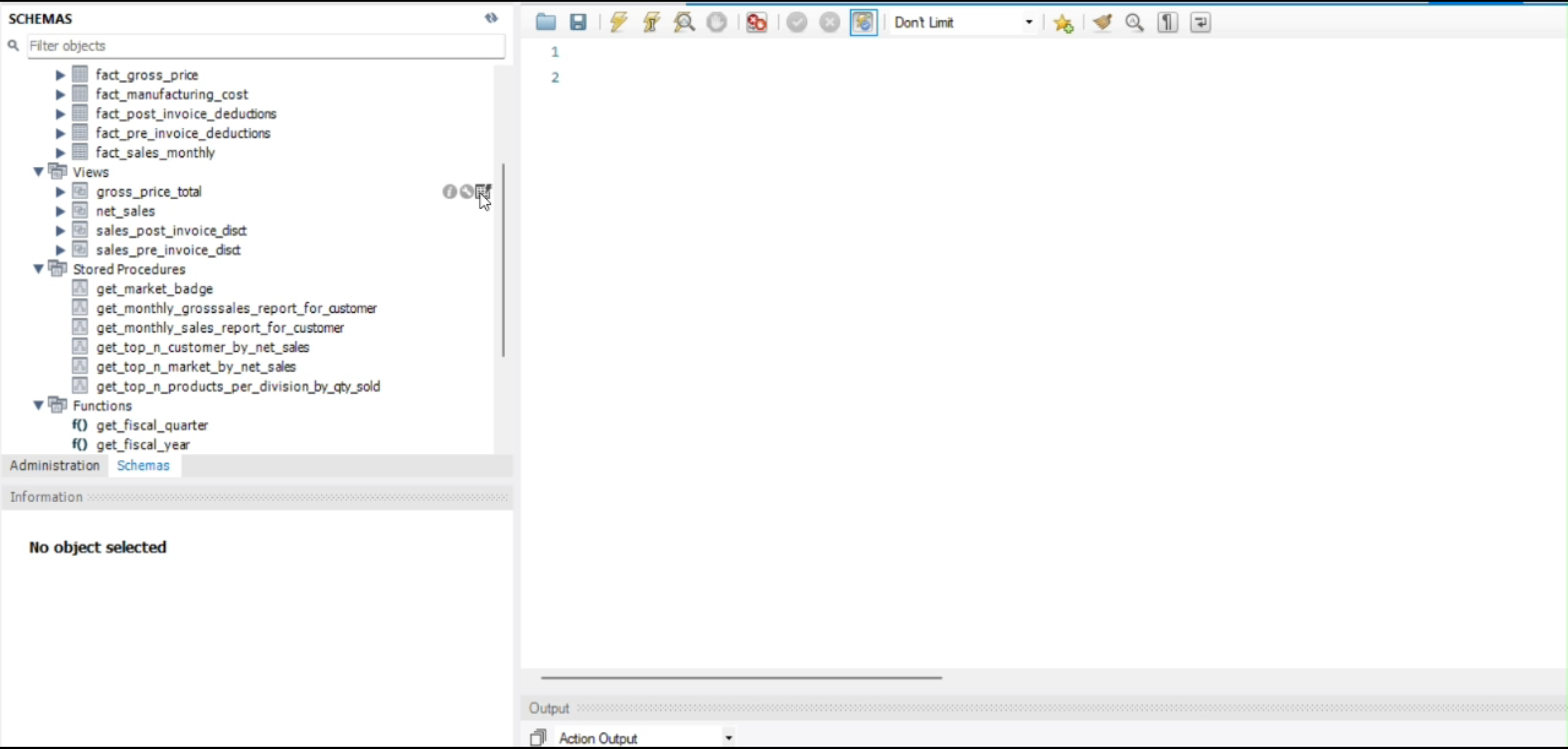
Create a stored procedure to return the Top N Customer.



Views
gross_price_total
net_sales
sales_post_invoice_disct
sales_pre_invoice_disct
Stored Procedures
get_market_badge
get_monthly_grosssales_report_for_customer
get_monthly_sales_report_for_customer
get_top_n_customer_by_net_sales
get_top_n_market_by_net_sales
get_top_n_products_per_division_by_qty_sold
Functions
get_fiscal_quarter
get_fiscal_year



```
CREATE DEFINER=`root`@`localhost` PROCEDURE `get_top_n_customer_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 gdb0041.net_sales n
    join
    dim_customer c on c.customer_code = n.customer_code
    where n.fiscal_year=in_fiscal_year
    and n.market= in_market
    group by customer
    order by net_sales_mln desc
    limit in_top_n;
END
```


SCHEMAS

Filter objects

- fact_gross_price
- fact_manufacturing_cost
- fact_post_invoice_deductions
- fact_pre_invoice_deductions
- fact_sales_monthly
- Views
  - gross_price_total
  - net_sales
  - sales_post_invoice_disct
  - sales_pre_invoice_disct
- Stored Procedures
  - get_market_badge
  - get_monthly_grosssales_report_for_customer
  - get_monthly_sales_report_for_customer
  - get_top_n_customer_by_net_sales
  - get_top_n_market_by_net_sales
  - get_top_n_products_per_division_by_qty_sold
- Functions
  - get_fiscal_quarter
  - get_fiscal_year

Administration Schemas

Information

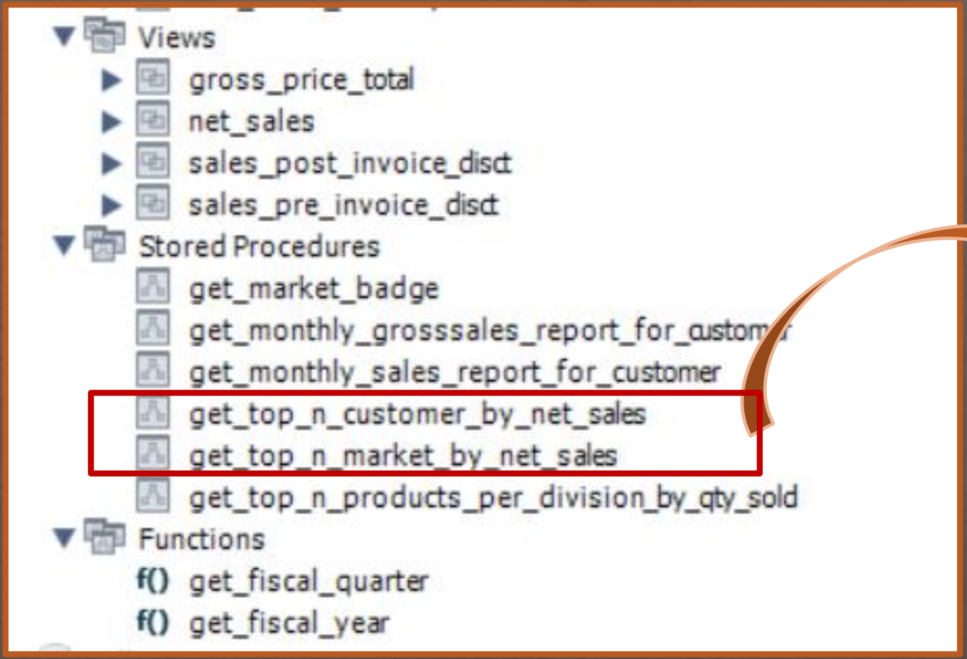
No object selected

Don't Limit

Output

Action Output

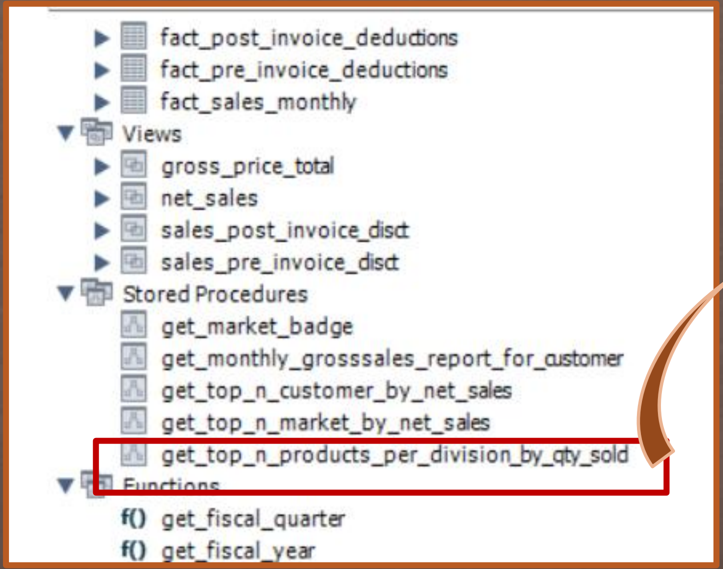
To support various analytical requirements, I developed a set of stored procedures , view and functions that streamline. These procedures ensure faster insights, consistency, and reusability across different business scenarios.



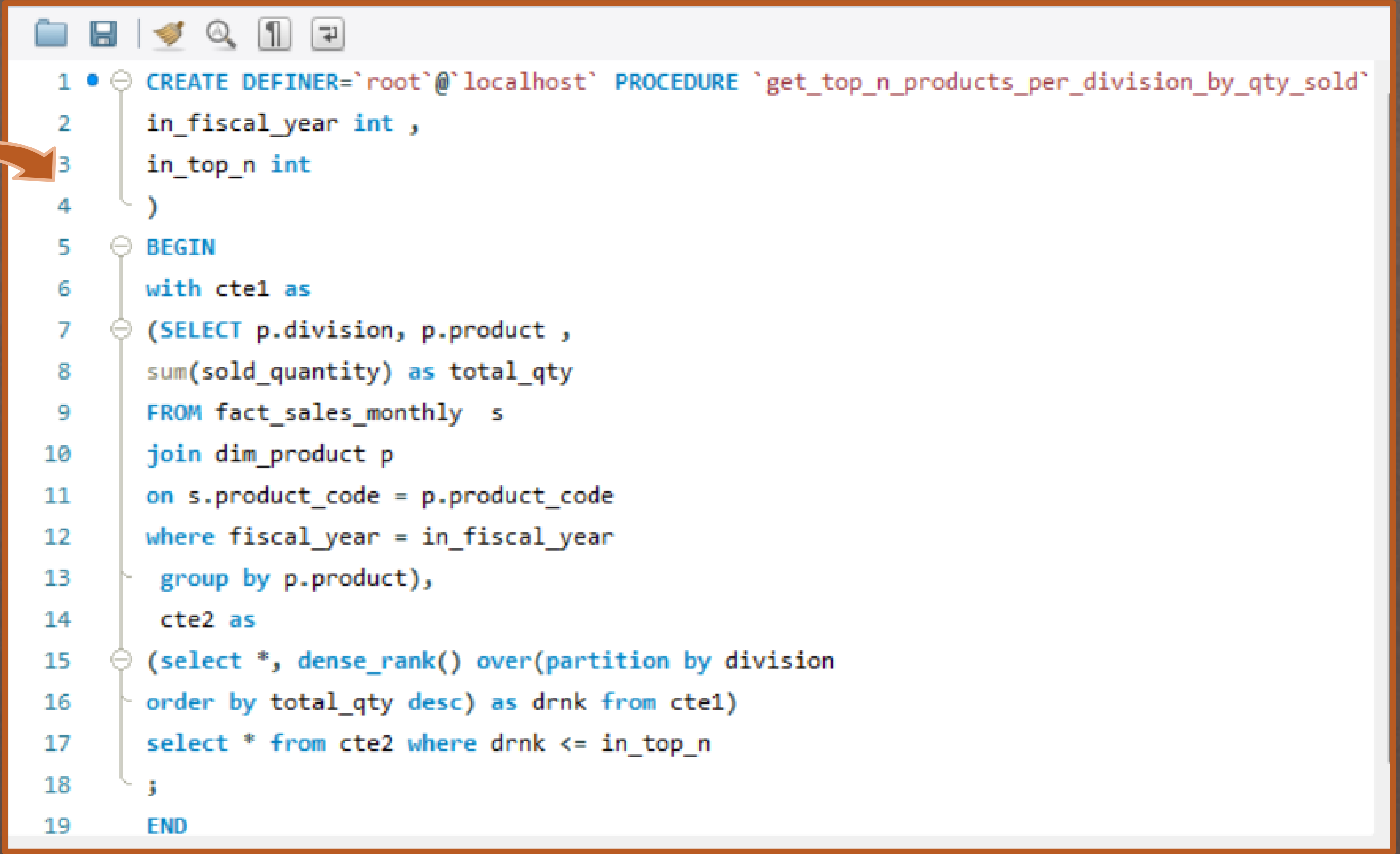
```
▼ Views
    ▶ gross_price_total
    ▶ net_sales
    ▶ sales_post_invoice_disct
    ▶ sales_pre_invoice_disct
▼ Stored Procedures
      get_market_badge
      get_monthly_grosssales_report_for_customer
      get_monthly_sales_report_for_customer
      get_top_n_customer_by_net_sales
      get_top_n_market_by_net_sales
      get_top_n_products_per_division_by_qty_sold
▼ Functions
    f() get_fiscal_quarter
    f() get_fiscal_year
```

The top-performing customers, markets for any fiscal year

Write a stored procedure to return the Top N products in each division based on quantity sold for a given financial year.
Inputs
Financial Year
N (number of top products required)



fact_post_invoice_deductions
fact_pre_invoice_deductions
fact_sales_monthly
Views
gross_price_total
net_sales
sales_post_invoice_disct
sales_pre_invoice_disct
Stored Procedures
get_market_badge
get_monthly_grosssales_report_for_customer
get_top_n_customer_by_net_sales
get_top_n_market_by_net_sales
get_top_n_products_per_division_by_qty_sold
Functions
f() get_fiscal_quarter
f() get_fiscal_year



```sql
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 s.product_code = p.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
```

Thank you for your time.
Looking forward to your feedback.