**Stored procedures**

**1) get\_market\_badge:**

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

IN in\_market varchar(45),

IN in\_fiscal\_year INT,

OUT out\_badge varchar(45)

)

BEGIN

declare qty int default 0;

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;

if qty > 5000000 then

SET out\_badge = "Gold";

else

SET out\_badge = "Silver";

end if;

END

**2) get\_monthly\_gross\_sales\_for\_customer:**

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

c\_code TEXT

)

BEGIN

select

s.date,

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

from fact\_sales\_monthly s

join fact\_gross\_price g

on

g.product\_code = s.product\_code

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

where find\_in\_set(s.customer\_code, c\_code) > 0

group by s.date

Order by s.date asc;

END

**3) get\_product\_by\_net\_sales**

CREATE DEFINER=`root`@`localhost` PROCEDURE `get\_product\_by\_net\_sales`(

IN in\_fiscal\_year INT,

IN in\_top\_N\_product INT

)

BEGIN

Select

product,

Round(sum(net\_sales)/1000000,2) as net\_sales\_millions

from net\_sales

where fiscal\_year = in\_fiscal\_year

group by product

order by net\_sales\_millions desc

limit in\_top\_N\_product;

END

**4) get\_top\_n\_customer\_by\_net\_sales**

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

IN in\_fiscal\_year INT,

IN in\_top\_n\_customers INT

)

BEGIN

Select

customer,

Round(sum(net\_sales)/1000000,2) as net\_sales\_millions

from net\_sales s

join dim\_customer c

on

s.customer\_code = c.customer\_code

where fiscal\_year = in\_fiscal\_year

group by customer

order by net\_sales\_millions desc

limit in\_top\_n\_customers;

END

**5) get\_top\_n\_customer\_by\_net\_sales\_by\_market**

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

IN in\_market varchar(45),

IN in\_fiscal\_year INT,

IN in\_top\_n\_customers INT

)

BEGIN

Select

customer,

Round(sum(net\_sales)/1000000,2) as net\_sales\_millions

from net\_sales s

join dim\_customer c

on

s.customer\_code = c.customer\_code

where fiscal\_year = in\_fiscal\_year and s.market = in\_market

group by customer

order by net\_sales\_millions desc

limit in\_top\_n\_customers;

END

**6) get\_top\_n\_markets\_by\_net\_sales**

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

IN in\_fiscal\_year int,

IN in\_top\_N\_countries int

)

BEGIN

Select

market,

Round(sum(net\_sales)/1000000,2) as net\_sales\_millions

from net\_sales

where fiscal\_year = in\_fiscal\_year

group by market

order by net\_sales\_millions desc

limit in\_top\_N\_countries;

END

**7) get\_forecast\_accuracy**

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

in\_fiscal\_year INT

)

BEGIN

WITH cteAccuracy AS (

SELECT

DISTINCT customer\_code,

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

SUM(s.forecast\_quantity) as total\_forecast\_quantity,

SUM(forecast\_quantity - sold\_quantity) AS net\_err,

SUM(forecast\_quantity - sold\_quantity) \* 100 / SUM(forecast\_quantity) AS net\_err\_pct,

SUM(ABS(forecast\_quantity - sold\_quantity)) AS abs\_err,

SUM(ABS(forecast\_quantity - sold\_quantity)) \* 100 / SUM(forecast\_quantity) AS abs\_err\_pct

FROM

fact\_act\_est s

WHERE

s.fiscal\_year = in\_fiscal\_year

GROUP BY

customer\_code

)

SELECT

ca.\*,

c.customer,

c.market,

IF(abs\_err\_pct > 100, 0, 100 - abs\_err\_pct) AS forecast\_accuracy

FROM

cteAccuracy ca

JOIN

dim\_customer c USING (customer\_code)

ORDER BY

forecast\_accuracy desc;

END

**Views**

**1) gross\_sales**

CREATE

ALGORITHM = UNDEFINED

DEFINER = `root`@`localhost`

SQL SECURITY DEFINER

VIEW `gross\_sales` AS

SELECT

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

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

`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\_per\_item`,

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

2) AS `gross\_price\_total`

FROM

(((`fact\_sales\_monthly` `s`

JOIN `dim\_product` `p` ON ((`s`.`product\_code` = `p`.`product\_code`)))

JOIN `dim\_customer` `c` ON ((`s`.`customer\_code` = `c`.`customer\_code`)))

JOIN `fact\_gross\_price` `g` ON (((`g`.`fiscal\_year` = `s`.`fiscal\_year`)

AND (`g`.`product\_code` = `s`.`product\_code`))))

**2) net\_sales**

CREATE

ALGORITHM = UNDEFINED

DEFINER = `root`@`localhost`

SQL SECURITY DEFINER

VIEW `net\_sales` AS

SELECT

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

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

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

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

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

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

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

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

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

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

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

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

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

FROM

`sales\_postinv\_discount`

**3) sales\_postinv\_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`,

((1 - `s`.`pre\_invoice\_discount\_pct`) \* `s`.`gross\_price\_total`) AS `net\_invoice\_sales`,

(`po`.`discounts\_pct` + `po`.`other\_deductions\_pct`) AS `post\_invoice\_discount\_pct`

FROM

(`sales\_preinv\_discount` `s`

JOIN `fact\_post\_invoice\_deductions` `po` ON (((`po`.`customer\_code` = `s`.`customer\_code`)

AND (`po`.`product\_code` = `s`.`product\_code`)

AND (`po`.`date` = `s`.`date`))))

**4) sales\_preinv\_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`,

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

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

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

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

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

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

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

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

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

FROM

((((`fact\_sales\_monthly` `s`

JOIN `dim\_product` `p` ON ((`p`.`product\_code` = `s`.`product\_code`)))

JOIN `dim\_customer` `c` ON ((`c`.`customer\_code` = `s`.`customer\_code`)))

JOIN `fact\_gross\_price` `g` ON (((`g`.`product\_code` = `s`.`product\_code`)

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

JOIN `fact\_pre\_invoice\_deductions` `pre` ON (((`pre`.`customer\_code` = `s`.`customer\_code`)

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

ORDER BY `s`.`date`

**Functions**

**1) get\_fiscal\_quarter**

CREATE DEFINER=`root`@`localhost` 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 = "Q!";

when m in (12,1,2) then

SET QTR = "Q2";

when m in (3,4,5) then

SET QTR = "Q3";

ELSE

SET QTR = "Q4";

END CASE;

RETURN QTR;

END

**2) get\_fiscal\_year**

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

calender\_year DATE

) RETURNS int

DETERMINISTIC

BEGIN

DECLARE fiscal\_year INT;

SET fiscal\_year = YEAR(DATE\_ADD(calender\_year, INTERVAL 4 MONTH));

RETURN fiscal\_year;

END