

PROBLEM STATEMENT AND SOLUTION

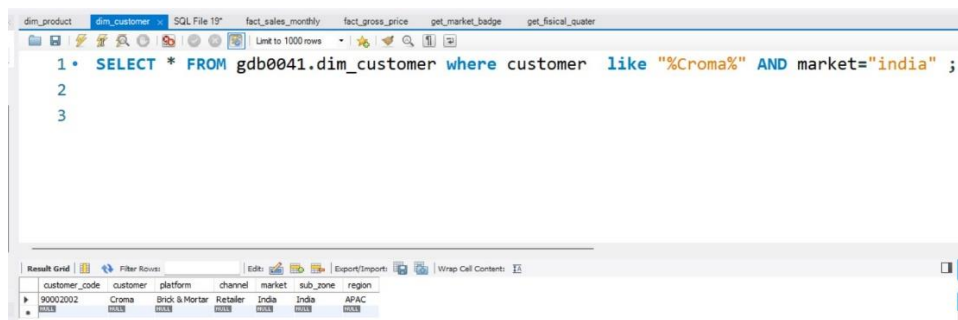
1/ Generate a report of individual product sales (aggregated on a monthly basis) for CROMA India customer for FY= 2021

The report should have the following fields,

1. Month
2. Product Name
3. Variant
4. Sold Quantity
5. Gross Price per item
6. Gross price total

-- a/ first grab customer codes for Croma india

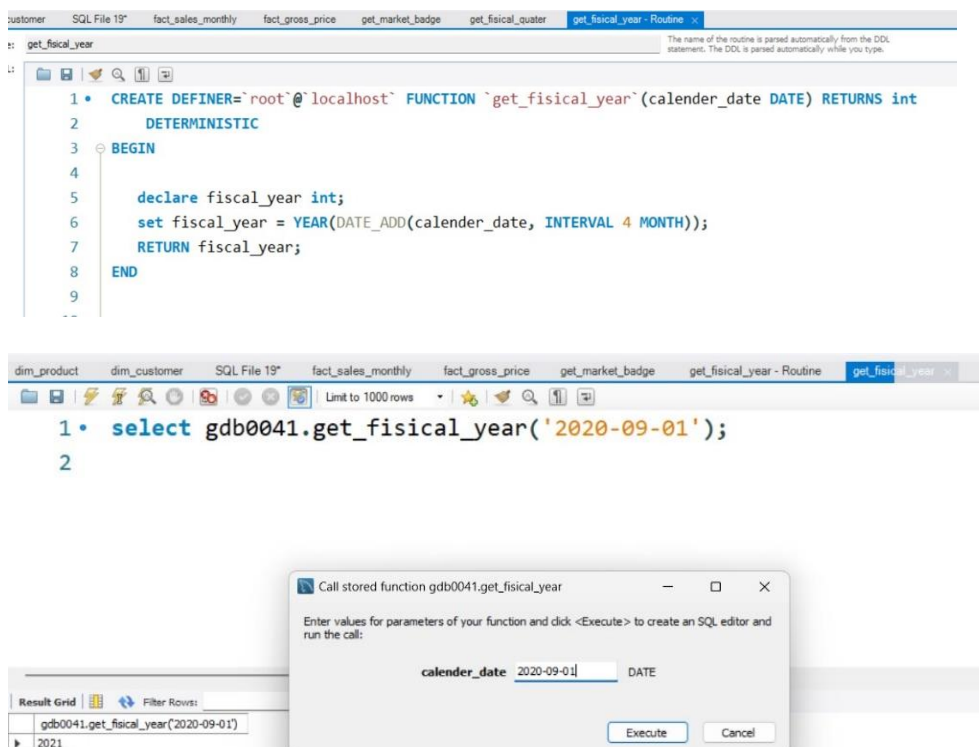
```
SELECT * FROM gdb0041.dim_customer where customer like "%Croma%" AND market="India" ;
```



The screenshot shows a SQL IDE window with a query editor and a result grid. The query is: `SELECT * FROM gdb0041.dim_customer where customer like "%Croma%" AND market="India" ;`. The result grid shows one row of data.

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

--b/ create a function 'get_fiscal_year' to get fiscal year by passing the date



The screenshot shows a SQL IDE window with a query editor and a result grid. The query is: `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`. The result grid shows one row of data.

```
1 • CREATE DEFINER='root'@'localhost' FUNCTION `get_fiscal_year` (calender_date DATE) RETURNS int
2   DETERMINISTIC
3   BEGIN
4
5   declare fiscal_year int;
6   set fiscal_year = YEAR(DATE_ADD(calender_date, INTERVAL 4 MONTH));
7   RETURN fiscal_year;
8   END
9
```

The result grid shows one row of data:

gdb0041.get_fiscal_year('2020-09-01')
2021

A dialog box titled "Call stored function gdb0041.get_fiscal_year" is shown, prompting the user to enter values for parameters of the function and click <Execute> to create an SQL editor and run the call. The dialog box shows the parameter "calender_date" with the value "2020-09-01" and the type "DATE".

--c/ create a function 'get_fiscal_quater' to get fiscal year by passing the date

```
dim_customer  SQL File 19*  fact_sales_monthly  fact_gross_price  get_market_badge  get_fiscal_year - Routine  get_fiscal_year  get_fiscal_quater - Routine
Name: get_fiscal_quater
DDL:
1 • CREATE DEFINER='root'@'localhost' FUNCTION `get_fiscal_quater` (calender_date date) RETURNS char(2) CHARSET utf8mb4
2 DETERMINISTIC
3 BEGIN
4 declare m tinyint;
5 declare qtr char(2);
6 set m = month(calender_date);
7 case
8 when m in (9,10,11) then
9 set qtr = "Q1";
10 when m in (12,1,2) then
11 set qtr = "Q2";
12 when m in (3,4,5) then
13 set qtr = "Q3";
14 ELSE
15 set qtr = "Q4";
16 end case;
17 RETURN qtr;
18 END
```

dim_customer SQL File 19* fact_sales_monthly fact_gross_price get_market_badge get_fiscal_year - Routine get_fiscal_year get_fiscal_quater - Routine

Limit to 1000 rows

```
1 • select gdb0041.get_fiscal_quater('2020-09-01');
2
```

Call stored function gdb0041.get_fiscal_quater

Enter values for parameters of your function and click <Execute> to create an SQL editor and run the call:

calender_date 2020-09-01 date

Execute Cancel

Result Grid | Filter Rows:

gdb0041.get_fiscal_quater('2020-09-01')
Q1

--d / sold quantity

```
26 -- sold quantity
27 • SELECT s.date, p.product_code, p.product,p.variant,s.sold_quantity
28 FROM gdb0041.fact_sales_monthly s join dim_product p
29 on p.product_code = s.product_code
30 where customer_code = 90002002 and
31 get_fiscal_year(date) = 2021
32 order by date desc;
```

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

date	product_code	product	variant	sold_quantity
2021-08-01	A0118150101	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Standard	88
2021-08-01	A0118150102	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Plus	196
2021-08-01	A0118150103	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Premium	50
2021-08-01	A0118150104	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Premium Plus	29
2021-08-01	A0219150201	AQ WereWolf NAS Internal Hard Drive HDD – 8....	Standard	95
2021-08-01	A0219150202	AQ WereWolf NAS Internal Hard Drive HDD – 8....	Plus	117
2021-08-01	A0220150203	AQ WereWolf NAS Internal Hard Drive HDD – 8....	Premium	54
2021-08-01	A0320150301	AQ Zion Saga	Standard	243
2021-08-01	A0321150302	AQ Zion Saga	Plus	75
2021-08-01	A0321150303	AQ Zion Saga	Premium	151
2021-08-01	A0418150103	AQ Mforce Gen X	Standard 3	140
2021-08-01	A0418150104	AQ Mforce Gen X	Plus 1	166
2021-08-01	A0418150105	AQ Mforce Gen X	Plus 2	163
2021-08-01	A0418150106	AQ Mforce Gen X	Plus 3	65
2021-08-01	A0519150201	AQ Mforce Gen Y	Standard 1	187
2021-08-01	A0519150202	AQ Mforce Gen Y	Standard 2	167
2021-08-01	A0519150203	AQ Mforce Gen Y	Standard 3	93
2021-08-01	A0519150204	AQ Mforce Gen Y	Plus 1	135
2021-08-01	A0519150205	AQ Mforce Gen Y	Plus 2	82
2021-08-01	A0519150206	AQ Mforce Gen Y	Plus 3	196
2021-08-01	A0519150207	AQ Mforce Gen Y	Premium 1	105
2021-08-01	A0519150208	AQ Mforce Gen Y	Premium 2	64
2021-08-01	A0619150301	AQ Mforce Gen Z	Standard 1	52

Result 9

--e/ Gross Price per item

```
dim_product dim_customer SQL File 19* fact_sales_monthly fact_gross_price get_market_badge get_fiscal_year - Routine get_fiscal_year get_fiscal_q
Limit to 1000 rows
33 order by date desc;
34
35
36 -- Gross price per item
37 • SELECT s.date, p.product_code, p.product,p.variant,s.sold_quantity , g.gross_price
38 FROM gdb0041.fact_sales_monthly s
39 join dim_product p
40 on p.product_code = s.product_code
41 join fact_gross_price g
42 on
43 g.product_code = s.product_code and
44 g.fiscal_year = get_fiscal_year(s.date)
45 where
46 customer_code = 90002002 and
47 get_fiscal_year(date) = 2021
48 order by date desc;
```

date	product_code	product	variant	sold_quantity	gross_price
2021-08-01	A3521150703	AQ Trigger	Plus 1	380	31.7665
2021-08-01	A2620150606	AQ Qwerty Ms	Premium 2	992	16.7850
2021-08-01	A7321160302	AQ Wi Power Dx3	Plus	193	43.9446
2021-08-01	A4620110603	AQ Gen Y	Standard Red	31	366.0459
2021-08-01	A0118150101	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Standard	88	19.0573
2021-08-01	A5519110303	AQ Gamer 3	Standard Black	11	604.2721
2021-08-01	A7321160301	AQ Wi Power Dx3	Standard	459	40.7954
2021-08-01	A2021150503	AQ MB Lito 2	Plus 2	35	45.4377
2021-08-01	A0118150102	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Plus	196	21.4565
2021-08-01	A7220160203	AQ Wi Power Dx2	Premium	586	37.4784
2021-08-01	A5519110302	AQ Gamer 3	Standard Cool...	10	605.8642
2021-08-01	A3420150601	AQ Qwerty	Standard 1	414	24.5690
2021-08-01	A7220160202	AQ Wi Power Dx2	Plus	264	35.2053
2021-08-01	A0118150103	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Premium	50	21.7795
2021-08-01	A7219160201	AQ Wi Power Dx2	Standard	531	32.9575
2021-08-01	A7119160103	AQ Wi Power Dx1	Premium	137	28.7736
2021-08-01	A0118150104	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Premium Plus	29	22.9729
2021-08-01	A5519110301	AQ Gamer 3	Standard Fire...	5	589.7411
2021-08-01	A3920150305	AQ LION x3	Premium	53	27.4546
2021-08-01	A7119160102	AQ Wi Power Dx1	Plus	454	29.9264
2021-08-01	A4419110406	AQ Elite	Plus Red	23	287.5666
2021-08-01	A0219150201	AQ WereWolf NAS Internal Hard Drive HDD – 8...	Standard	95	23.6987
2021-08-01	A2620150605	AQ Qwerty Ms	Premium 1	603	16.5882
2021-08-01	A7118160101	AQ Wi Power Dx1	Standard	251	29.6712
2021-08-01	A6819160203	AO Pen Drive DRC	Premium	436	5.0984

--f/ Gross price total

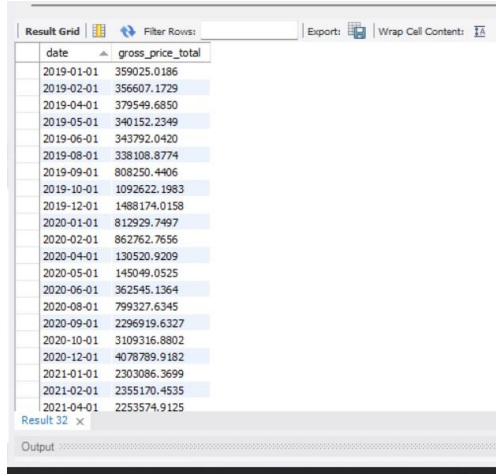
```
dim_product dim_customer SQL File 19* fact_sales_monthly fact_gross_price get_market_badge get_fiscal_year - Routine get_fiscal_year get_fiscal_quater - Routine
Limit to 1000 rows
51 -- gross price total
52 • SELECT s.date, p.product_code, p.product,p.variant,s.sold_quantity , g.gross_price, round(g.gross_price*s.sold_quantity)
53 as gross_price_total
54 FROM gdb0041.fact_sales_monthly s
55 join dim_product p
56 on p.product_code = s.product_code
57 join fact_gross_price g
58 on
59 g.product_code = s.product_code and
60 g.fiscal_year = get_fiscal_year(s.date)
61 where
62 customer_code = 90002002 and
63 get_fiscal_year(date) = 2021
64 order by date asc;
```

date	product_code	product	variant	sold_quantity	gross_price	gross_price_total
2020-09-01	A0118150101	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Standard	202	19.0573	3850
2020-09-01	A4419110403	AQ Elite	Standard Red	16	288.0503	4609
2020-09-01	A2720150701	AQ Trigger Ms	Standard 1	822	17.0917	14049
2020-09-01	A4218110204	AQ Digit	Plus Grey	27	232.1038	6267
2020-09-01	A5419110205	AQ Gamer 2	Plus Cool Blue	7	570.7578	3995
2020-09-01	A5419110206	AQ Gamer 2	Plus Black	4	601.6398	2407
2020-09-01	A3220150401	AQ Lite	Standard 1	197	18.4943	3643
2020-09-01	A5419110204	AQ Gamer 2	Plus Firey Red	5	602.9200	3015
2020-09-01	A2620150606	AQ Qwerty Ms	Premium 2	688	16.7850	11548
2020-09-01	A0118150102	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Plus	162	21.4565	3476
2020-09-01	A4319110304	AQ Velocity	Plus Grey	40	267.0636	10683
2020-09-01	A5419110207	AQ Gamer 2	Premium Black	5	599.2302	2996
2020-09-01	A2721150702	AQ Trigger Ms	Standard 2	171	17.2368	2947
2020-09-01	A2021150503	AQ MB Lito 2	Plus 2	17	45.4377	772
2020-09-01	A5419110208	AQ Gamer 2	Premium Mist...	4	608.4070	2434
2020-09-01	A3718150103	AQ LION x1	Plus 2	28	17.5697	492
2020-09-01	A3920150305	AQ LION x3	Premium	66	27.4546	1812
2020-09-01	A5419110203	AQ Gamer 2	Standard Black	14	595.5711	8338
2020-09-01	A0118150103	AQ Dracula HDD – 3.5 Inch SATA 6 Gb/s 5400 R...	Premium	193	21.7795	4203
2020-09-01	A6818160202	AQ Pen Drive DRC	Plus	855	3.8531	3294
2020-09-01	A4118110105	AQ Aspiren	Plus Blue	22	182.8932	4024

2/ Gross Sales Report: Total Sales Amount

--a/ Aggregate Monthly Gross Sales Report for CROMA India for all the years: Prepare a report on aggregate monthly gross sales to track sales generated by CROMA India.

```
73
74 -- monthly sales report
75 • SELECT s.date, SUM(g.gross_price * s.sold_quantity) as gross_price_total
76 FROM gdb0041.fact_sales_monthly s join fact_gross_price g
77 on s.product_code = g.product_code and
78 g.fiscal_year = get_fisical_year(s.date)
79 where customer_code = 90002002
80 group by s.date
81 order by s.date asc;
82
```

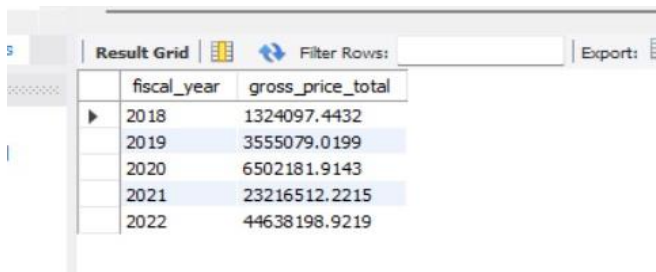


The screenshot shows a database query result grid with two columns: 'date' and 'gross_price_total'. The data is sorted by date in ascending order, showing monthly sales from January 2019 to April 2021. The total sales amount for the period shown is 2253574.9125.

date	gross_price_total
2019-01-01	359025.0186
2019-02-01	356607.1729
2019-04-01	379549.6850
2019-05-01	340152.2349
2019-06-01	343792.0420
2019-08-01	338108.8774
2019-09-01	808250.4406
2019-10-01	1092622.1983
2019-12-01	1488174.0158
2020-01-01	812929.7497
2020-02-01	862762.7656
2020-04-01	130520.9209
2020-05-01	145049.0525
2020-06-01	362545.1364
2020-08-01	799327.6345
2020-09-01	2296919.6327
2020-10-01	3109316.8802
2020-12-01	4078789.9182
2021-01-01	2303086.3699
2021-02-01	2355170.4535
2021-04-01	2253574.9125

--b / Croma yearly gross sale report

```
83 -- croma yerly gross sale report
84 • SELECT get_fisical_year(s.date) as fiscal_year, SUM(g.gross_price * s.sold_quantity)
85 as gross_price_total FROM gdb0041.fact_sales_monthly s join fact_gross_price g
86 on s.product_code = g.product_code and
87 g.fiscal_year = get_fisical_year(s.date)
88 where customer_code = 90002002
89 group by get_fisical_year(s.date)
90 order by fiscal_year;
91
```



The screenshot shows a database query result grid with two columns: 'fiscal_year' and 'gross_price_total'. The data is sorted by fiscal year in ascending order, showing the total sales for each year from 2018 to 2022.

fiscal_year	gross_price_total
2018	1324097.4432
2019	3555079.0199
2020	6502181.9143
2021	23216512.2215
2022	44638198.9219

-- Generate monthly gross sales report for any customer using stored procedure

```
SQL File 19*  fact_sales_monthly  fact_gross_price  get_market_badge  get_fiscal_year - Routine  get_fiscal_year  get_monthly_gross_sale_for_cu... x  get_monthly_gross_sale_for_cu...

Name: get_monthly_gross_sale_for_customer The name of the routine is parsed automatically from the DDL statement. The DDL is parsed automatically while you type.

DDL:
1 • CREATE DEFINER=`root`@`localhost` PROCEDURE `get_monthly_gross_sale_for_customer`(in_customer_code TEXT)
2 BEGIN
3     SELECT s.date, SUM(g.gross_price * s.sold_quantity) as gross_price_total FROM gdb0041.fact_sales_monthly s join fact_gross_price g
4     on s.product_code = g.product_code and
5     g.fiscal_year = get_fiscal_year(s.date)
6     where
7     find_in_set(s.customer_code, in_customer_code)>0
8     group by s.date;
9 END
```

dim_customer SQL File 19* fact_sales_monthly fact_gross_price get_market_badge get_fiscal_year - Routine get_fiscal_year get_monthly_gross_sale_for_customer

Limit to 1000 rows

```
1 • call gdb0041.get_monthly_gross_sale_for_customer('90002002');
2
```

Result Grid Filter Rows: Export: Wrap Cell Contents:

date	gross_price_total
2021-12-01	19537146.5599
2021-10-01	13908229.2869
2021-09-01	11192823.0751
2021-08-01	2349478.8219
2021-06-01	2288587.4483
2021-05-01	2181587.7843
2021-04-01	2253574.9125
2021-02-01	2355170.4535
2021-01-01	2303086.3699
2020-12-01	4078789.9182
2020-10-01	3109316.8802
2020-09-01	2296919.6327
2020-08-01	799327.6345
2020-06-01	362545.1364
2020-05-01	145049.0525
2020-04-01	130520.9209
2020-02-01	862762.7656
2020-01-01	812929.7497

Call stored procedure gdb0041.get_monthly_gross_sale_f...

Enter values for parameters of your procedure and click <Execute> to create an SQL editor and run the call:

in_customer_code 90002002 [IN] TEXT

Execute Cancel

-- Generate yearly gross sales report for any customer using stored procedure

```
dim_customer SQL File 19* fact_sales_monthly fact_gross_price get_market_badge get_fiscal_year - Routine get_yearly_gross_sale_for_cust... x

Name: get_yearly_gross_sale_for_customer The name of the routine is parsed automatically from the DDL statement. The DDL is parsed automatically while you type.

DDL:
1 • CREATE DEFINER=`root`@`localhost` PROCEDURE `get_yearly_gross_sale_for_customer`( in_customer_code TEXT)
2 BEGIN
3     SELECT get_fiscal_year(s.date) as fiscal_year, SUM(g.gross_price * s.sold_quantity) as gross_price_total
4     FROM gdb0041.fact_sales_monthly s join fact_gross_price g
5     on s.product_code = g.product_code and
6     g.fiscal_year = get_fiscal_year(s.date)
7     where
8     find_in_set(s.customer_code, in_customer_code)>0
9     group by get_fiscal_year(s.date);
10
11 END
```

dim_customer SQL File 19* fact_sales_monthly fact_gross_price get_market_badge get_fiscal_year - Routine get_yearly_gross_sale_for_cust...

Limit to 1000 rows

```
1 • call gdb0041.get_yearly_gross_sale_for_customer('90002002');
2
```

Result Grid Filter Rows: Export: Wrap Cell Contents:

fiscal_year	gross_price_total
2018	1324097.4432
2019	3555079.0199
2020	6502181.9143
2021	23216512.2215
2022	44638198.9219

Call stored procedure gdb0041.get_yearly_gross_sale_for_...

Enter values for parameters of your procedure and click <Execute> to create an SQL editor and run the call:

in_customer_code 90002002 [IN] TEXT

Execute Cancel

--Market Badge Determination: Create a stored procedure to determine the market badge (Gold or Silver) based on the total sold quantity.

```
fact_gross_price  get_market_badge  get_fisical_year - Routine  get_yearly_gross_sale_for_cust...  get_yearly_gross_sale_for_cust...  get_yearly_gross_sale_for_cust...  get_market_badge - Routine
Name: get_market_badge
DDL:
1 • CREATE DEFINER=`root`@`localhost` PROCEDURE `get_market_badge`(IN in_market varchar(45),
2   IN in_fisical_year year,
3   OUT out_badge varchar(45))
4 BEGIN
5     declare quantity int default 0;
6     if in_market = " " then
7         set in_market = "India";
8     end if;
9     select sum(sold_quantity) into quantity from fact_sales_monthly s join dim_customer c
10    on s.customer_code = c.customer_code
11    where get_fisical_year(s.date)=2021 and market = in_market
12    group by c.market;
13    if quantity > 5000000 then
14        set out_badge = "glod";
15    else set out_badge = "silver";
16    end if;
17
18 END
```

fact_gross_price get_market_badge get_fisical_year - Routine get_yearly_gross_sale_for_cust... get_yearly_gross_sale_for_cust... get_yearly_gross...

```
1 • set @out_badge = '0';
2 • call gdb0041.get_market_badge('india', 2021, @out_badge);
3 • select @out_badge;
4
```

Call stored procedure gdb0041.get_market_badge

Enter values for parameters of your procedure and click <Execute> to create an SQL editor and run the call:

in_market	india	[IN]	varchar(45)
in_fisical_year	2021	[IN]	year
out_badge		[OUT]	varchar(45)

Execute Cancel

Result Grid | Filter Rows:

@out_badge
glod