

Part 3(last) of my project :

Supply Chain ANALYTICS for AtliQ Hardwares using MySQL

task:1

Forecast accuracy report using cte (It exists at the scope of statements)

query and result

```
1 • set sql_mode='';
2 • with forecast_err_table as (      select
3       s.customer_code as customer_code,
4       c.customer as customer,
5       c.market as market,
6       (s.forecast_quantity) as forecast_qty,
7       (s.sold_quantity) as sold_qty,
8       (s.forecast_quantity-s.sold_quantity) ,
9       sum(s.sold_quantity) as total_sold_qty,
10      sum(s.forecast_quantity) as total_forecast_qty,
11      sum(s.forecast_quantity-s.sold_quantity) as net_error,
12      round(sum(s.forecast_quantity-s.sold_quantity)*100/sum(s.forecast_quantity),1) as net_error_pct,
13      sum(abs(s.forecast_quantity-s.sold_quantity)) as abs_error,
14      round(sum(abs(s.forecast_quantity-s.sold_quantity))*100/sum(s.forecast_quantity),2) as abs_error_pct
15  from fact_act_est s
16  join dim_customer c
17  on s.customer_code = c.customer_code
18  where s.fiscal_year=2021
19  group by customer_code)
20
21 select
22 *,
23 if (abs_error_pct > 100, 0, 100.0 - abs_error_pct) as forecast_accuracy
24 from forecast_err_table
25 order by forecast_accuracy desc;
```

customer_code	customer	market	forecast_qty	sold_qty	(s.forecast_quantity-s.sold_quantity)	total_sold_qty	total_forecast_qty	net_error	net_error_pct	abs_error	abs_error_pct	forecast_accuracy
90013120	Coolblue	Italy	26	23	3	109547	133532	23985	18	70467	52.77	47.23
70010048	Atliq e Store	Bangladesh	38	34	4	119439	142010	22571	15.9	75711	53.31	46.69
90023027	Costco	Canada	26	51	-25	236189	279962	43773	15.6	149303	53.33	46.67
90023026	Relief	Canada	47	9	38	228988	273492	44504	16.3	146948	53.73	46.27

TASK:2

Write a stored proc for the same

```
1 CREATE DEFINER='root'@'localhost' PROCEDURE `get_forecast_accuracy`(  
2     in_fiscal_year INT  
3 )  
4 BEGIN  
5     with forecast_err_table as (  
6         select  
7             s.customer_code as customer_code,  
8             c.customer as customer_name,  
9             c.market as market,  
10            sum(s.sold_quantity) as total_sold_qty,  
11            sum(s.forecast_quantity) as total_forecast_qty,  
12            sum(s.forecast_quantity-s.sold_quantity) as net_error,  
13            round(sum(s.forecast_quantity-s.sold_quantity)*100/sum(s.forecast_quantity),1) as net_error_pct,  
14            sum(abs(s.forecast_quantity-s.sold_quantity)) as abs_error,  
15            round(sum(abs(s.forecast_quantity-s.sold_quantity))*100/sum(s.forecast_quantity),2) as abs_error_pct  
16        from fact_act_est s  
17        join dim_customer c  
18        on s.customer_code = c.customer_code  
19        where s.fiscal_year=in_fiscal_year  
20        group by customer_code  
21    )  
22    select  
23        *,  
24        if (abs_error_pct > 100, 0, 100.0 - abs_error_pct) as forecast_accuracy  
25    from forecast_err_table  
26    order by forecast_accuracy desc;  
27 END
```

Call stored procedure gdb0041.get_forecast_accuracy

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

in_fiscal_year [IN] INT

Execute Cancel

RESULT

Limit to 1000 rows

```

1 • call gdb0041.get_forecast_accuracy(2021);
2

```

customer_code	customer_name	market	total_sold_qty	total_forecast_qty	net_error	net_error_pct	abs_error	abs_error_pct	forecast_accuracy
90013120	Coolblue	Italy	109547	133532	23985	18	70467	52.77	47.23
70010048	Atliq e Store	Bangladesh	119439	142010	22571	15.9	75711	53.31	46.69
90023027	Costco	Canada	236189	279962	43773	15.6	149303	53.33	46.67
90023026	Relief	Canada	228988	273492	44504	16.3	146948	53.73	46.27
90017051	Forward S...	Portugal	86823	118067	31244	26.5	63568	53.84	46.16
90017058	Mbit	Portugal	86860	110195	23335	21.2	59473	53.97	46.03
90023028	walmart	Canada	239081	283323	44242	15.6	153058	54.02	45.98
90023024	Sage	Canada	246397	287233	40836	14.2	155610	54.18	45.82
90015146	Mbit	Norway	147152	210507	63355	30.1	114189	54.24	45.76
90013124	Amazon	Italy	110898	136116	25218	18.5	73826	54.24	45.76
90017054	Flawless S...	Portugal	84371	114698	30327	26.4	62483	54.48	45.52
70027208	Atliq e Store	Brazil	33713	47321	13608	28.8	25784	54.49	45.51
90015147	Chiptec	Norway	154897	223867	68970	30.8	122100	54.54	45.46
80001019	Neptune	China	1113979	1275248	161269	12.6	695779	54.56	45.44
90015144	Sound	Norway	160074	225637	65563	29.1	123257	54.63	45.37
90009130	Logic Stores	Newzealand	103290	110175	6885	6.2	60225	54.66	45.34
90015149	UniEuro	Norway	142086	212500	70414	33.1	116172	54.67	45.33
90021088	Electricals...	United Kin...	224350	323689	99339	30.7	176975	54.67	45.33
90017050	Electricals...	Portugal	85272	114688	29416	25.6	62760	54.72	45.28

TASK:3

Forecast accuracy report using temporary table (It exists for the entire session)

```

1 • drop table if exists forecast_err_table;
2 • create temporary table forecast_err_table
3     select
4         s.customer_code as customer_code,
5         c.customer as customer_name,
6         c.market as market,
7         sum(s.sold_quantity) as total_sold_qty,
8         sum(s.forecast_quantity) as total_forecast_qty,
9         sum(s.forecast_quantity-s.sold_quantity) as net_error,
10        round(sum(s.forecast_quantity-s.sold_quantity)*100/sum(s.forecast_quantity),1) as net_error_pct,
11        sum(abs(s.forecast_quantity-s.sold_quantity)) as abs_error,
12        round(sum(abs(s.forecast_quantity-s.sold_quantity))*100/sum(s.forecast_quantity),2) as abs_error_pct
13    from fact_act_est s
14    join dim_customer c
15    on s.customer_code = c.customer_code
16    where s.fiscal_year=2021
17    group by customer_code;
18
19 • select
20     *,
21     if (abs_error_pct > 100, 0, 100.0 - abs_error_pct) as forecast_accuracy
22 from forecast_err_table
23 order by forecast_accuracy desc;
24

```

result Grid | Filter Rows: | Export: | Wrap Cell Content: |

customer_code	customer_name	market	total_sold_qty	total_forecast_qty	net_error	net_error_pct	abs_error	abs_error_pct	forecast_accuracy
90013120	Coolblue	Italy	109547	133532	23985	18	70467	52.77	47.23
70010048	Atliq e Store	Bangladesh	119439	142010	22571	15.9	75711	53.31	46.69
90023027	Costco	Canada	236189	279962	43773	15.6	149303	53.33	46.67
90023026	Relief	Canada	228988	273492	44504	16.3	146948	53.73	46.27
90017051	Forward Stores	Portugal	86823	118067	31244	26.5	63568	53.84	46.16

TASK: 4

Learning about triggers

Database Triggers

create the trigger to automatically insert record in fact_act_est table whenever insertion happens in fact_sales_monthly

A trigger in MySQL is a set of SQL statements that reside in a system catalog. It is a special type of stored procedure that is invoked automatically in response to an event. Each trigger is associated with a table, which is activated on any DML statement such as INSERT, UPDATE, or DELETE.

Why we need/use triggers in MySQL?

We need/use triggers in MySQL due to the following features:

- Triggers help us to enforce business rules.
- Triggers help us to validate data even before they are inserted or updated.
- Triggers help us to keep a log of records like maintaining audit trails in tables.
- SQL triggers provide an alternative way to check the integrity of data.
- Triggers provide an alternative way to run the scheduled task.
- Triggers increases the performance of SQL queries because it does not need to compile each time the query is executed.

My Work:

```
CREATE DEFINER=CURRENT_USER TRIGGER `fact_sales_monthly_AFTER_INSERT` AFTER INSERT ON `fact_sales_monthly` FOR EACH ROW
```

```
BEGIN
```

```
    insert into fact_act_est
```

```
        (date, product_code, customer_code, sold_quantity)
```

```
values (
```

```
    NEW.date,
```

```
    NEW.product_code,
```

```
    NEW.customer_code,
```

```
NEW.sold_quantity
)

on duplicate key update

sold_quantity = values(sold_quantity);

END
```

create the trigger to automatically insert record in fact_act_est table whenever insertion happens in fact_forecast_monthly

```
CREATE DEFINER=CURRENT_USER TRIGGER `fact_forecast_monthly_AFTER_INSERT` AFTER INSERT ON `fact_forecast_monthly` FOR EACH ROW
```

```
BEGIN
```

```
insert into fact_act_est
```

```
(date, product_code, customer_code, forecast_quantity)
```

```
values (
```

```
NEW.date,
```

```
NEW.product_code,
```

```
NEW.customer_code,
```

```
NEW.forecast_quantity
```

```
)
```

```
on duplicate key update
```

```
forecast_quantity = values(forecast_quantity);
```

```
END
```

```
-- To see all the Triggers
```

```
show triggers;
```

-- Insert the records in the fact_sales_monthly and fact_forecast_monthly tables and check whether records inserted in fact_act_est table

insert into fact_sales_monthly

(date, product_code, customer_code, sold_quantity)

values

("2030-09-01", "HAHA", 99, 89);

insert into fact_forecast_monthly

(date, product_code, customer_code, forecast_quantity)

values

("2030-09-01", "HAHA", 99, 43);

select * from fact_act_est where customer_code = 99;

TASK:5

Learn about Database Events

MySQL Events are tasks that run according to a schedule. Therefore, we sometimes refer to them as *scheduled* events. When you create an event, you are creating a named database object containing one or more SQL statements to be executed at one or more regular intervals, beginning and ending at a specific date and time.

-- To show all the events

show events;

-- Show variable which have event in it

show variables like "%event%";

-- Creating the table "session_logs" in the random table and also insert the records in it

```
CREATE TABLE random_tables.session_logs (`ts` DATETIME, `session_id` INT, `user_id` INT,
`log` TEXT);
```

```
INSERT INTO `random_tables`.`session_logs`
```

```
    (`ts`, `session_id`, `user_id`, `log`)
```

```
VALUES
```

```
    ('2022-10-04 08:14:07', '898812', '523', 'CLICKED | Courses Buttom'),
```

```
    ('2022-10-14 08:18:35', '898812', '523', 'NAVIAGE BACK | Python course page ,
codebasics.io'),
```

```
    ('2022-10-16 12:07:00', '965345', '523', 'REVIEW GENERATED | Data analytics in power
bi'),
```

```
    ('2022-10-22 14:09:22', '188567', '707', 'NEW LOGIN | New login, user name:
tasty@jalebi.com'),
```

```
    ('2022-10-22 18:10:06', '188567', '707', 'COURSE PURCHASED | Data analytics in power
bi, user name: tasty@jalebi.com');
```

```
-- Delete logs that are less than 5 days old
```

```
delimiter |
```

```
CREATE EVENT e_daily_log_purge
```

```
ON SCHEDULE
```

```
EVERY 5 SECOND
```

```
COMMENT 'Purge logs that are more than 5 days old'
```

```
DO
```

```
BEGIN
```

```
delete from random_tables.session_logs
```

```
where DATE(ts) < DATE("2022-10-22") - interval 5 day;
```

```
END |
```

```
delimiter ;
```

-- drop the event

drop event if exists e_daily_log_purge;