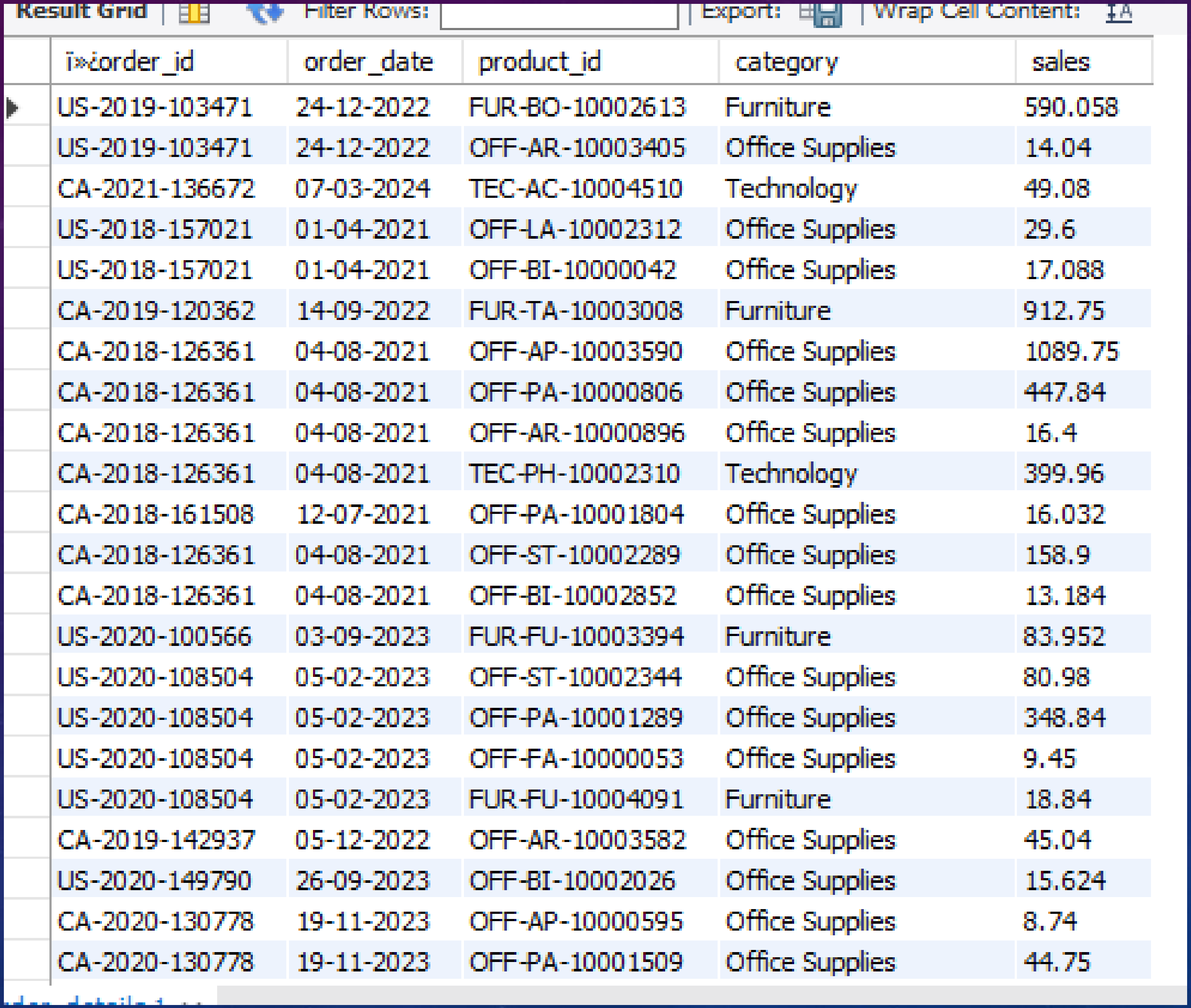
SQL TIPS AND TRICKS

PART 37

SQL for Business Intelligence

MAYURI DANDEKAR

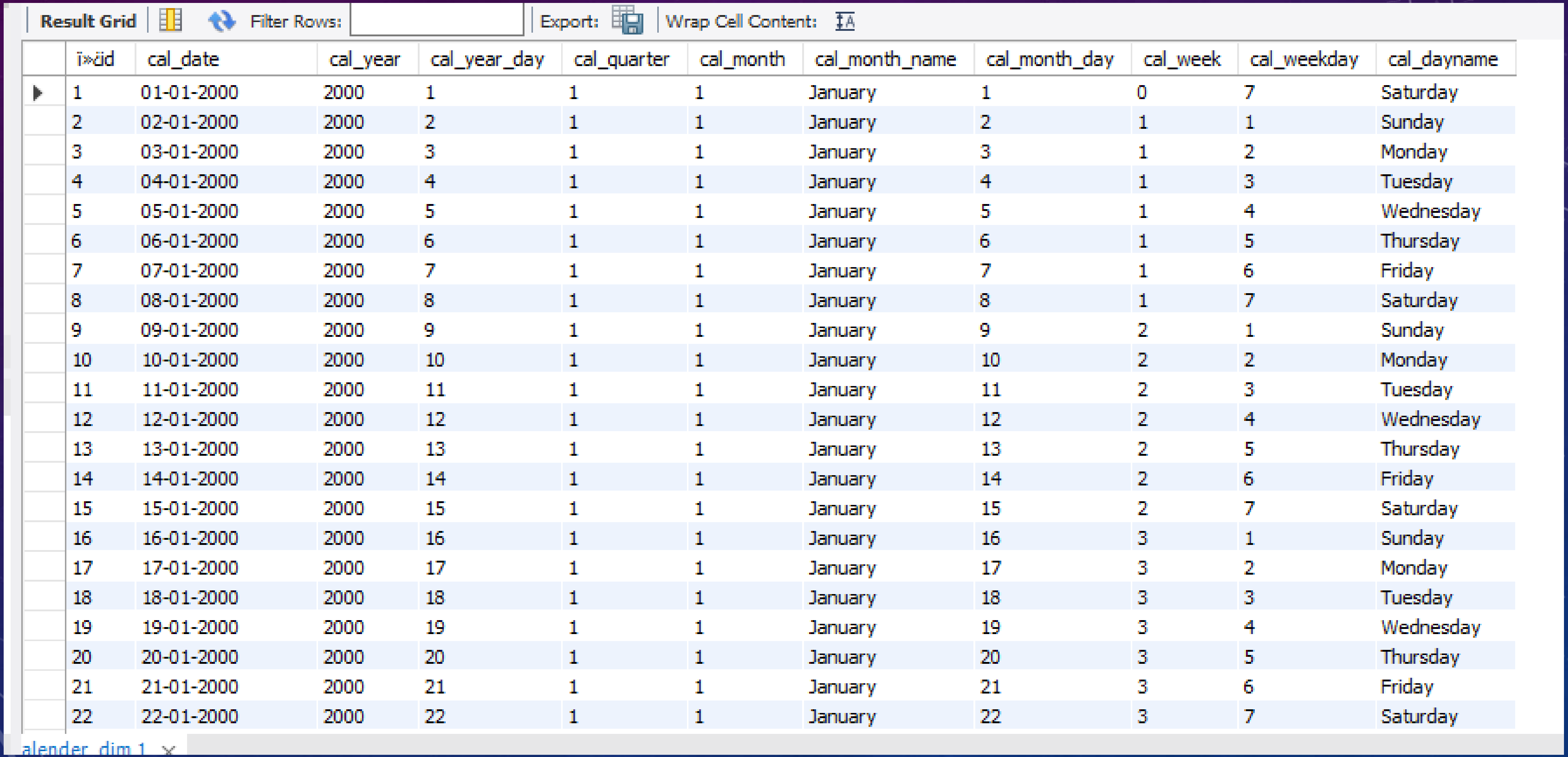
Sample "**order_details**" table of 702 records



ï»¿order_id	order_date	product_id	category	sales
US-2019-103471	24-12-2022	FUR-BO-10002613	Furniture	590.058
US-2019-103471	24-12-2022	OFF-AR-10003405	Office Supplies	14.04
CA-2021-136672	07-03-2024	TEC-AC-10004510	Technology	49.08
US-2018-157021	01-04-2021	OFF-LA-10002312	Office Supplies	29.6
US-2018-157021	01-04-2021	OFF-BI-10000042	Office Supplies	17.088
CA-2019-120362	14-09-2022	FUR-TA-10003008	Furniture	912.75
CA-2018-126361	04-08-2021	OFF-AP-10003590	Office Supplies	1089.75
CA-2018-126361	04-08-2021	OFF-PA-10000806	Office Supplies	447.84
CA-2018-126361	04-08-2021	OFF-AR-10000896	Office Supplies	16.4
CA-2018-126361	04-08-2021	TEC-PH-10002310	Technology	399.96
CA-2018-161508	12-07-2021	OFF-PA-10001804	Office Supplies	16.032
CA-2018-126361	04-08-2021	OFF-ST-10002289	Office Supplies	158.9
CA-2018-126361	04-08-2021	OFF-BI-10002852	Office Supplies	13.184
US-2020-100566	03-09-2023	FUR-FU-10003394	Furniture	83.952
US-2020-108504	05-02-2023	OFF-ST-10002344	Office Supplies	80.98
US-2020-108504	05-02-2023	OFF-PA-10001289	Office Supplies	348.84
US-2020-108504	05-02-2023	OFF-FA-10000053	Office Supplies	9.45
US-2020-108504	05-02-2023	FUR-FU-10004091	Furniture	18.84
CA-2019-142937	05-12-2022	OFF-AR-10003582	Office Supplies	45.04
US-2020-149790	26-09-2023	OFF-BI-10002026	Office Supplies	15.624
CA-2020-130778	19-11-2023	OFF-AP-10000595	Office Supplies	8.74
CA-2020-130778	19-11-2023	OFF-PA-10001509	Office Supplies	44.75

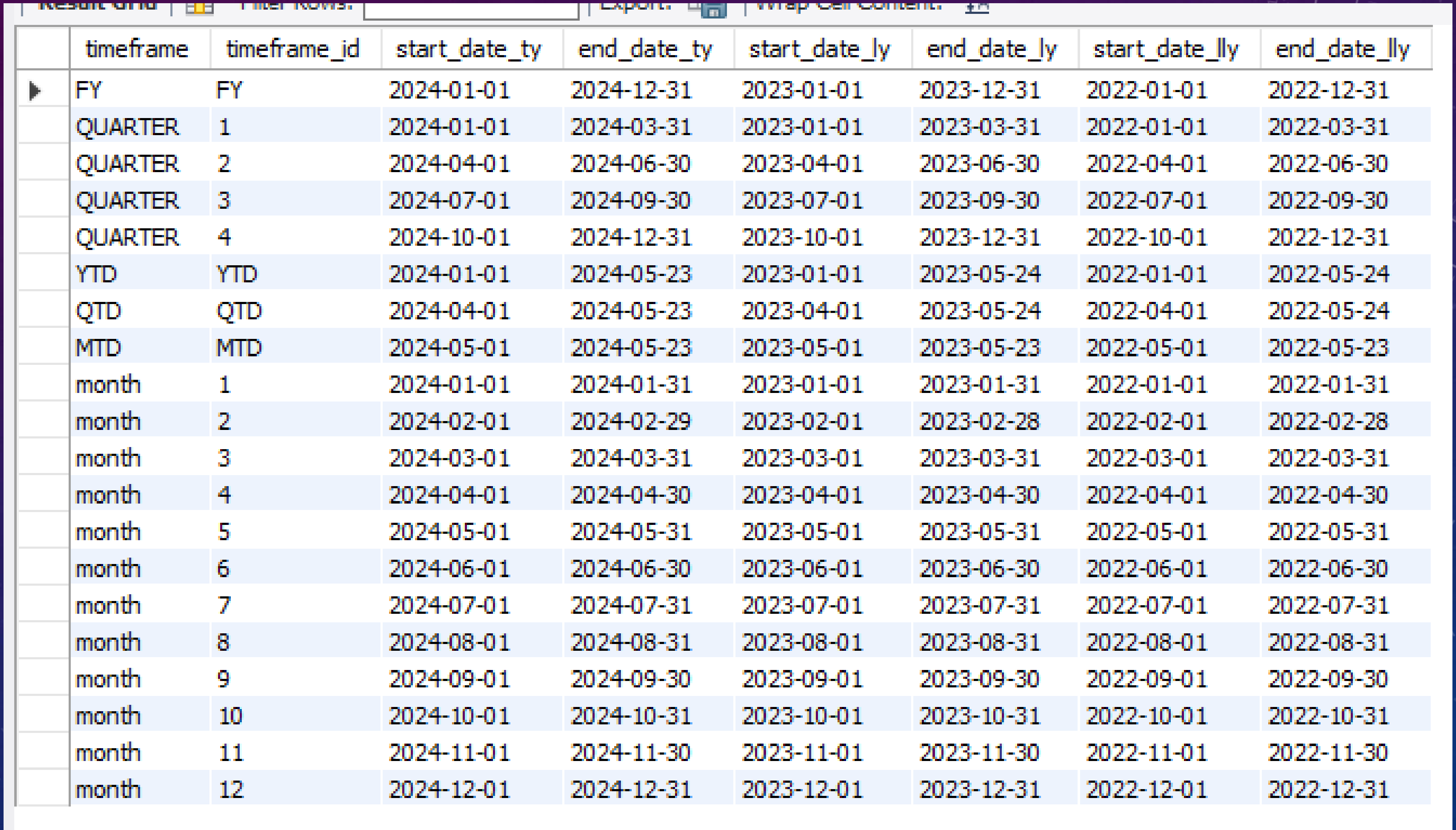
STEP 1– create calender_dim table

(Check **PART 15** to create calender table)

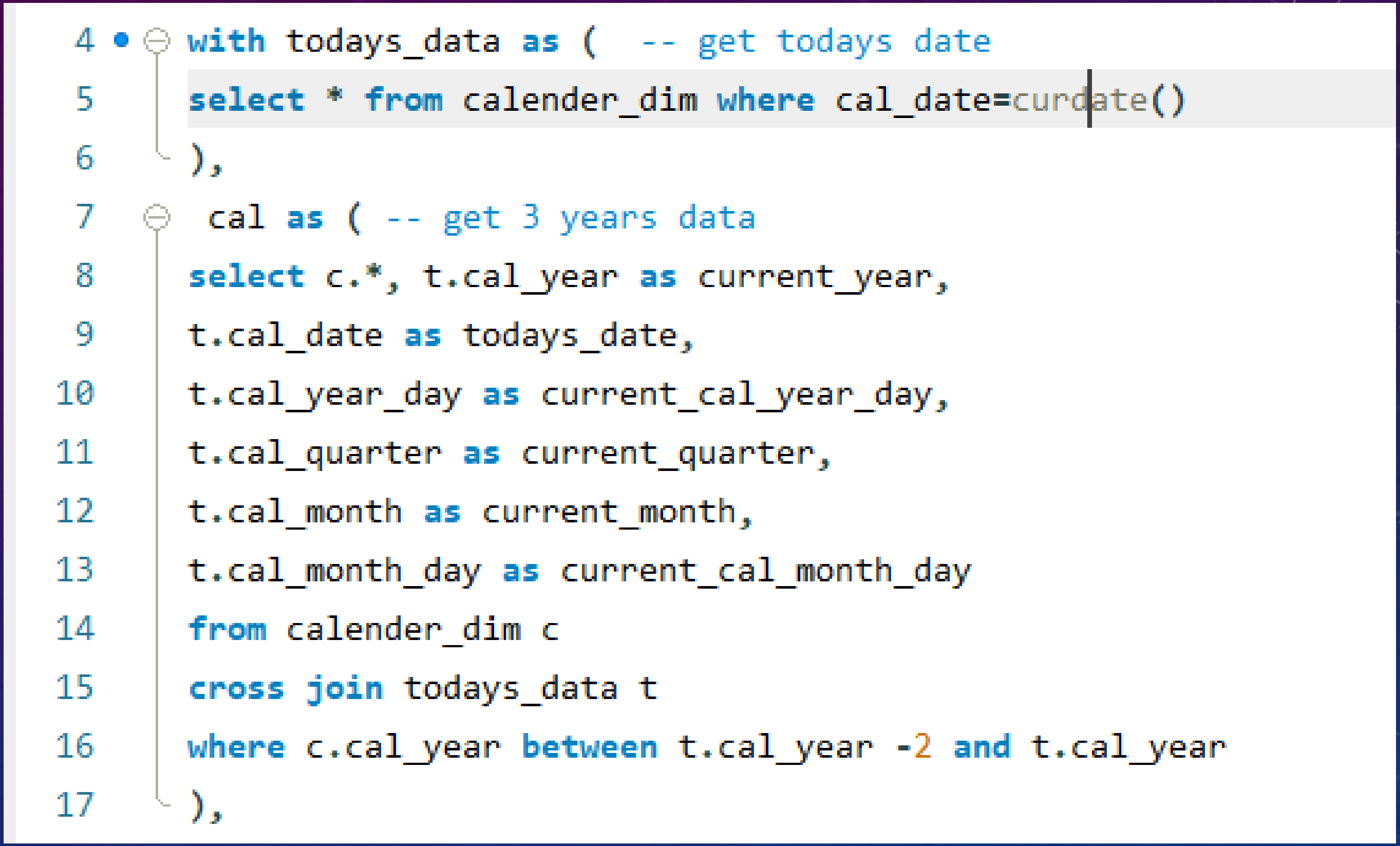


ï»¿id	cal_date	cal_year	cal_year_day	cal_quarter	cal_month	cal_month_name	cal_month_day	cal_week	cal_weekday	cal_dayname
1	01-01-2000	2000	1	1	1	January	1	0	7	Saturday
2	02-01-2000	2000	2	1	1	January	2	1	1	Sunday
3	03-01-2000	2000	3	1	1	January	3	1	2	Monday
4	04-01-2000	2000	4	1	1	January	4	1	3	Tuesday
5	05-01-2000	2000	5	1	1	January	5	1	4	Wednesday
6	06-01-2000	2000	6	1	1	January	6	1	5	Thursday
7	07-01-2000	2000	7	1	1	January	7	1	6	Friday
8	08-01-2000	2000	8	1	1	January	8	1	7	Saturday
9	09-01-2000	2000	9	1	1	January	9	2	1	Sunday
10	10-01-2000	2000	10	1	1	January	10	2	2	Monday
11	11-01-2000	2000	11	1	1	January	11	2	3	Tuesday
12	12-01-2000	2000	12	1	1	January	12	2	4	Wednesday
13	13-01-2000	2000	13	1	1	January	13	2	5	Thursday
14	14-01-2000	2000	14	1	1	January	14	2	6	Friday
15	15-01-2000	2000	15	1	1	January	15	2	7	Saturday
16	16-01-2000	2000	16	1	1	January	16	3	1	Sunday
17	17-01-2000	2000	17	1	1	January	17	3	2	Monday
18	18-01-2000	2000	18	1	1	January	18	3	3	Tuesday
19	19-01-2000	2000	19	1	1	January	19	3	4	Wednesday
20	20-01-2000	2000	20	1	1	January	20	3	5	Thursday
21	21-01-2000	2000	21	1	1	January	21	3	6	Friday
22	22-01-2000	2000	22	1	1	January	22	3	7	Saturday

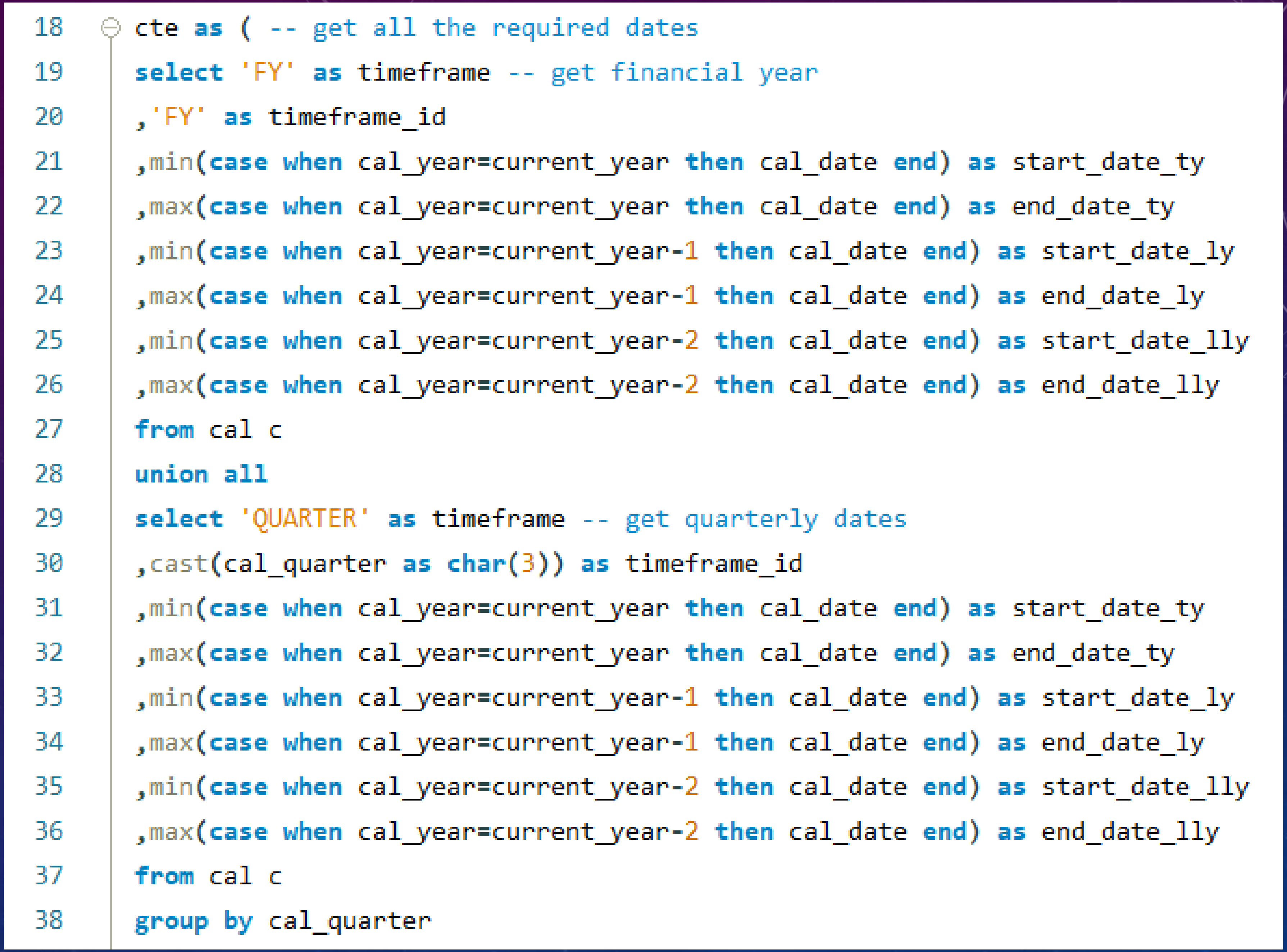
STEP 2– create "**timeframes**" table as per requirement



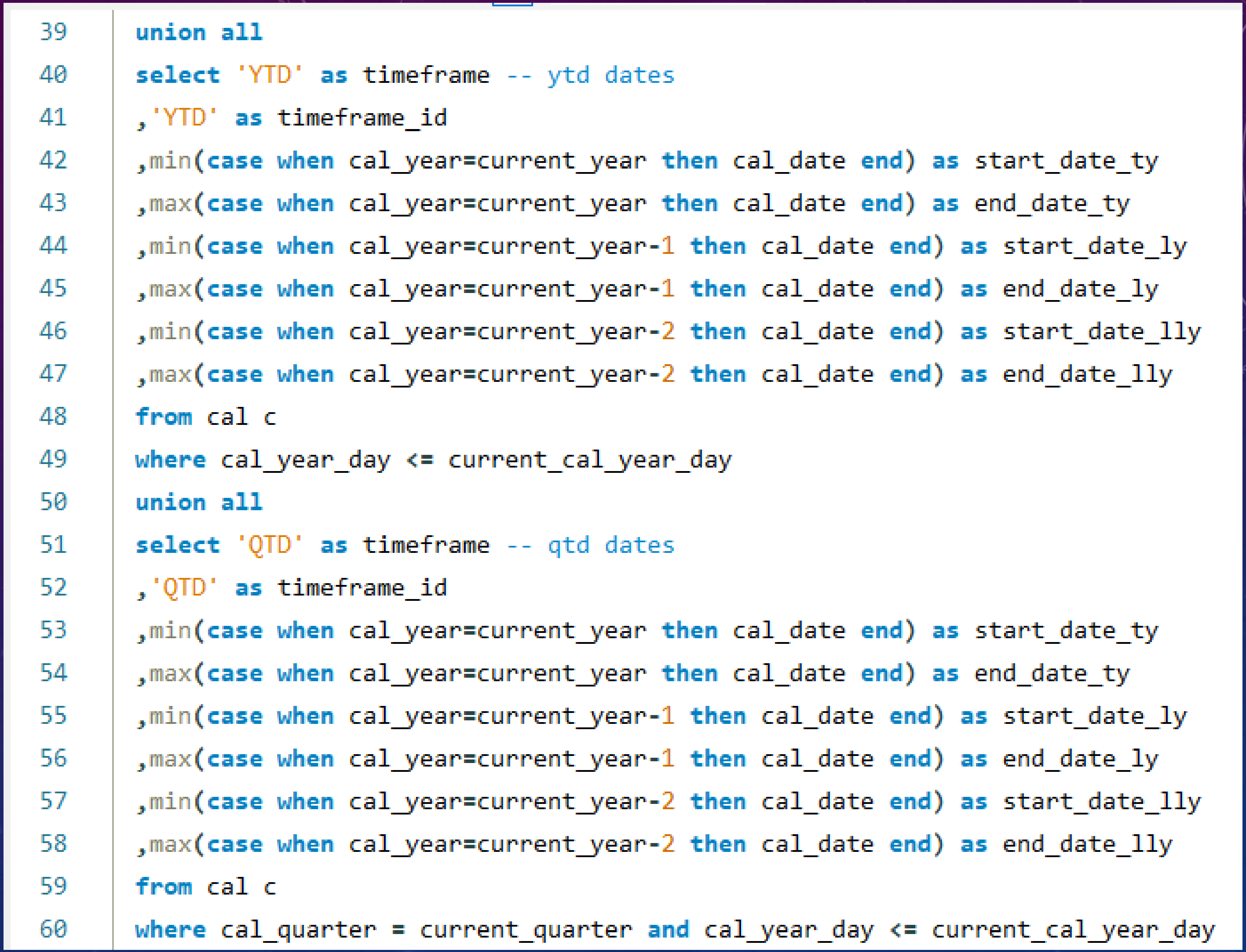
timeframe	timeframe_id	start_date_ty	end_date_ty	start_date_ly	end_date_ly	start_date_lly	end_date_lly
FY	FY	2024-01-01	2024-12-31	2023-01-01	2023-12-31	2022-01-01	2022-12-31
QUARTER	1	2024-01-01	2024-03-31	2023-01-01	2023-03-31	2022-01-01	2022-03-31
QUARTER	2	2024-04-01	2024-06-30	2023-04-01	2023-06-30	2022-04-01	2022-06-30
QUARTER	3	2024-07-01	2024-09-30	2023-07-01	2023-09-30	2022-07-01	2022-09-30
QUARTER	4	2024-10-01	2024-12-31	2023-10-01	2023-12-31	2022-10-01	2022-12-31
YTD	YTD	2024-01-01	2024-05-23	2023-01-01	2023-05-24	2022-01-01	2022-05-24
QTD	QTD	2024-04-01	2024-05-23	2023-04-01	2023-05-24	2022-04-01	2022-05-24
MTD	MTD	2024-05-01	2024-05-23	2023-05-01	2023-05-23	2022-05-01	2022-05-23
month	1	2024-01-01	2024-01-31	2023-01-01	2023-01-31	2022-01-01	2022-01-31
month	2	2024-02-01	2024-02-29	2023-02-01	2023-02-28	2022-02-01	2022-02-28
month	3	2024-03-01	2024-03-31	2023-03-01	2023-03-31	2022-03-01	2022-03-31
month	4	2024-04-01	2024-04-30	2023-04-01	2023-04-30	2022-04-01	2022-04-30
month	5	2024-05-01	2024-05-31	2023-05-01	2023-05-31	2022-05-01	2022-05-31
month	6	2024-06-01	2024-06-30	2023-06-01	2023-06-30	2022-06-01	2022-06-30
month	7	2024-07-01	2024-07-31	2023-07-01	2023-07-31	2022-07-01	2022-07-31
month	8	2024-08-01	2024-08-31	2023-08-01	2023-08-31	2022-08-01	2022-08-31
month	9	2024-09-01	2024-09-30	2023-09-01	2023-09-30	2022-09-01	2022-09-30
month	10	2024-10-01	2024-10-31	2023-10-01	2023-10-31	2022-10-01	2022-10-31
month	11	2024-11-01	2024-11-30	2023-11-01	2023-11-30	2022-11-01	2022-11-30
month	12	2024-12-01	2024-12-31	2023-12-01	2023-12-31	2022-12-01	2022-12-31



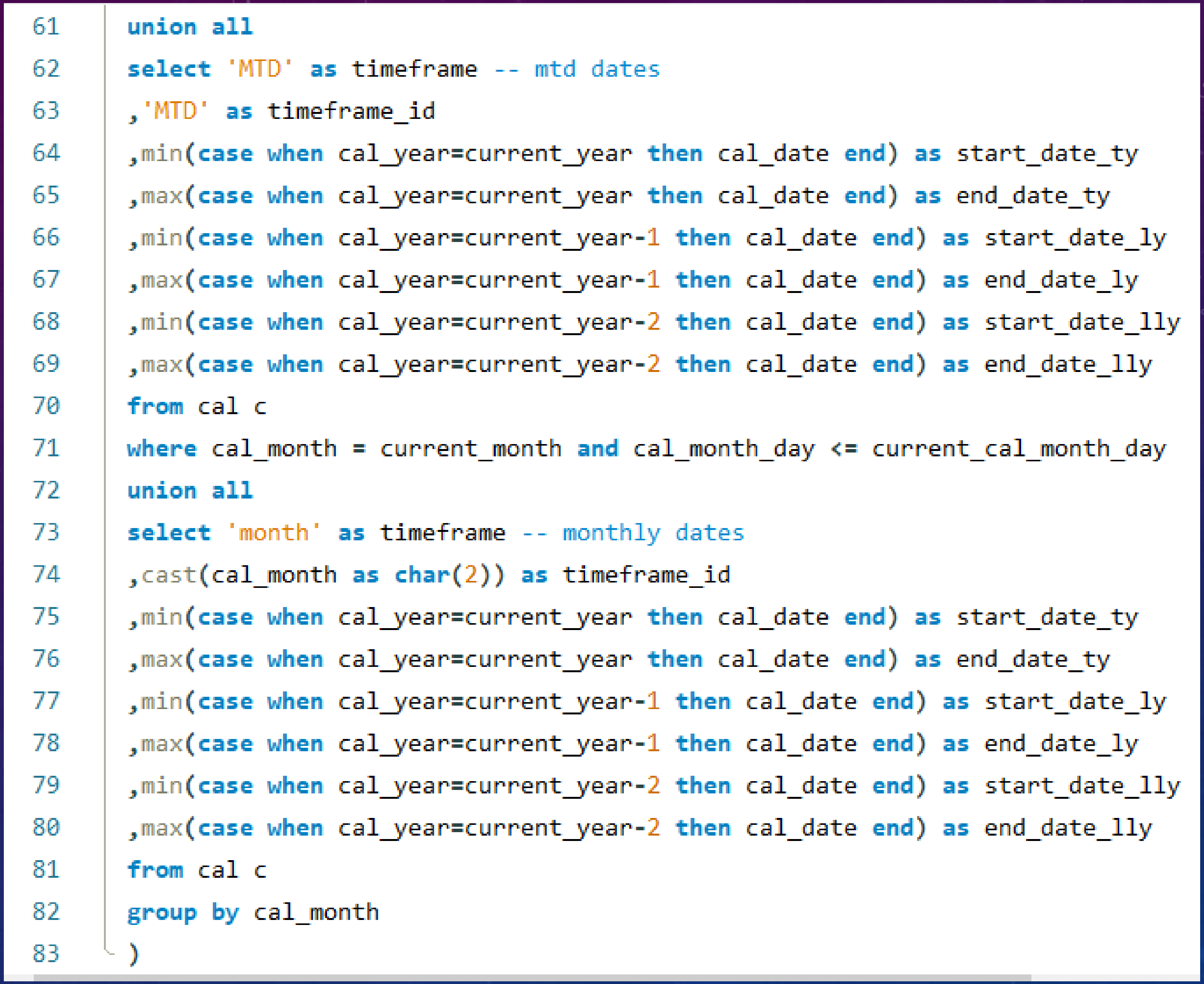
```sql
with todays_data as (  -- get todays date
select * from calender_dim where cal_date=curdate()
),
 cal as ( -- get 3 years data
select c.*, t.cal_year as current_year,
t.cal_date as todays_date,
t.cal_year_day as current_cal_year_day,
t.cal_quarter as current_quarter,
t.cal_month as current_month,
t.cal_month_day as current_cal_month_day
from calender_dim c
cross join todays_data t
where c.cal_year between t.cal_year -2 and t.cal_year
),
```



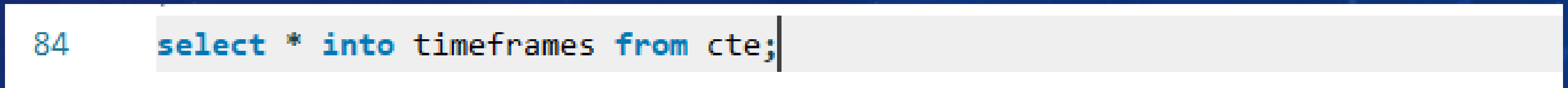
```sql
cte as ( -- get all the required dates
select 'FY' as timeframe -- get financial year
,'FY' as timeframe_id
,min(case when cal_year=current_year then cal_date end) as start_date_ty
,max(case when cal_year=current_year then cal_date end) as end_date_ty
,min(case when cal_year=current_year-1 then cal_date end) as start_date_ly
,max(case when cal_year=current_year-1 then cal_date end) as end_date_ly
,min(case when cal_year=current_year-2 then cal_date end) as start_date_lly
,max(case when cal_year=current_year-2 then cal_date end) as end_date_lly
from cal c
union all
select 'QUARTER' as timeframe -- get quarterly dates
,cast(cal_quarter as char(3)) as timeframe_id
,min(case when cal_year=current_year then cal_date end) as start_date_ty
,max(case when cal_year=current_year then cal_date end) as end_date_ty
,min(case when cal_year=current_year-1 then cal_date end) as start_date_ly
,max(case when cal_year=current_year-1 then cal_date end) as end_date_ly
,min(case when cal_year=current_year-2 then cal_date end) as start_date_lly
,max(case when cal_year=current_year-2 then cal_date end) as end_date_lly
from cal c
group by cal_quarter
```



```sql
union all
select 'YTD' as timeframe -- ytd dates
,'YTD' as timeframe_id
,min(case when cal_year=current_year then cal_date end) as start_date_ty
,max(case when cal_year=current_year then cal_date end) as end_date_ty
,min(case when cal_year=current_year-1 then cal_date end) as start_date_ly
,max(case when cal_year=current_year-1 then cal_date end) as end_date_ly
,min(case when cal_year=current_year-2 then cal_date end) as start_date_lly
,max(case when cal_year=current_year-2 then cal_date end) as end_date_lly
from cal c
where cal_year_day <= current_cal_year_day
union all
select 'QTD' as timeframe -- qtd dates
,'QTD' as timeframe_id
,min(case when cal_year=current_year then cal_date end) as start_date_ty
,max(case when cal_year=current_year then cal_date end) as end_date_ty
,min(case when cal_year=current_year-1 then cal_date end) as start_date_ly
,max(case when cal_year=current_year-1 then cal_date end) as end_date_ly
,min(case when cal_year=current_year-2 then cal_date end) as start_date_lly
,max(case when cal_year=current_year-2 then cal_date end) as end_date_lly
from cal c
where cal_quarter = current_quarter and cal_year_day <= current_cal_year_day
```

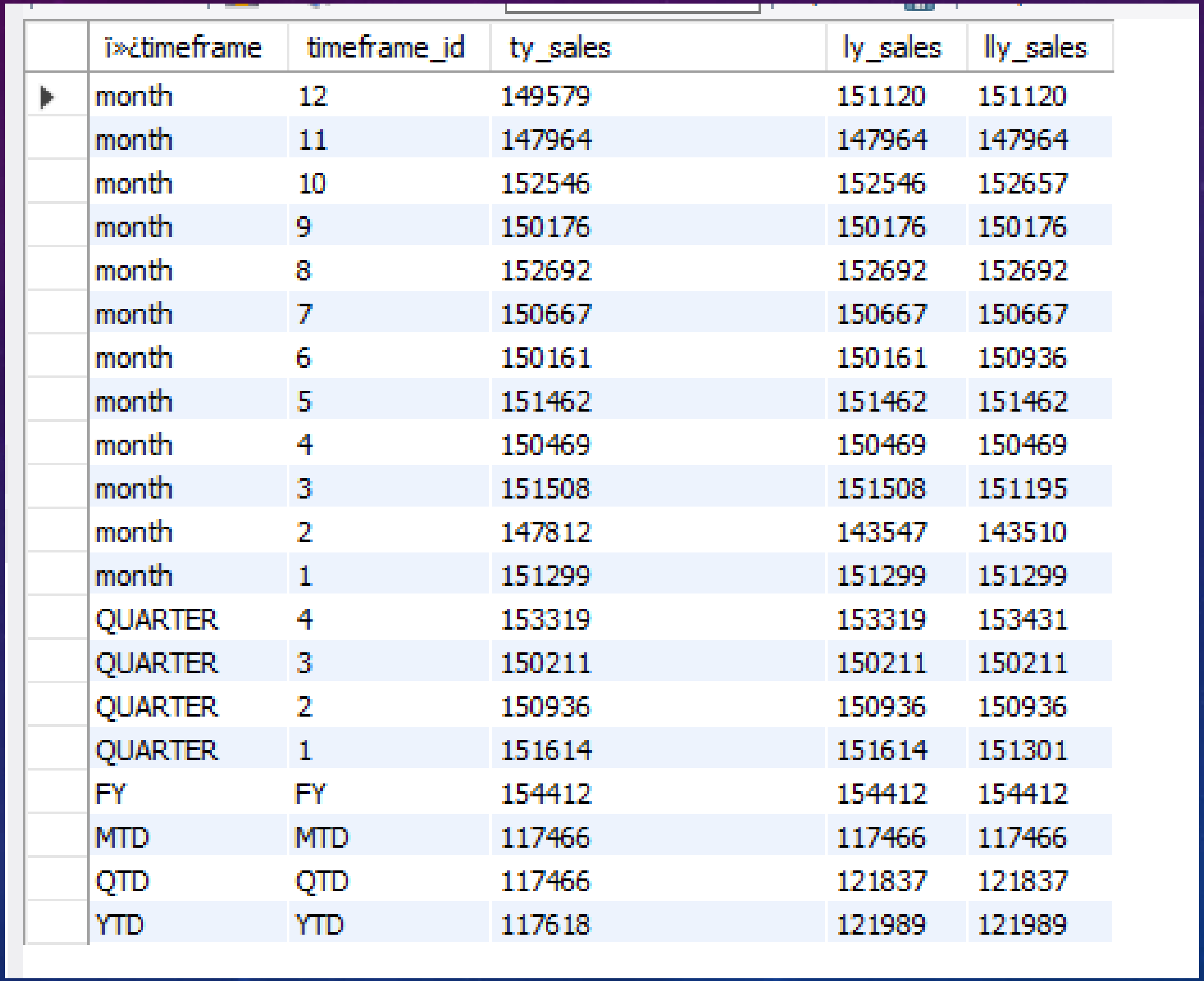


```sql
union all
select 'MTD' as timeframe -- mtd dates
,'MTD' as timeframe_id
,min(case when cal_year=current_year then cal_date end) as start_date_ty
,max(case when cal_year=current_year then cal_date end) as end_date_ty
,min(case when cal_year=current_year-1 then cal_date end) as start_date_ly
,max(case when cal_year=current_year-1 then cal_date end) as end_date_ly
,min(case when cal_year=current_year-2 then cal_date end) as start_date_lly
,max(case when cal_year=current_year-2 then cal_date end) as end_date_lly
from cal c
where cal_month = current_month and cal_month_day <= current_cal_month_day
union all
select 'month' as timeframe -- monthly dates
,cast(cal_month as char(2)) as timeframe_id
,min(case when cal_year=current_year then cal_date end) as start_date_ty
,max(case when cal_year=current_year then cal_date end) as end_date_ty
,min(case when cal_year=current_year-1 then cal_date end) as start_date_ly
,max(case when cal_year=current_year-1 then cal_date end) as end_date_ly
,min(case when cal_year=current_year-2 then cal_date end) as start_date_lly
,max(case when cal_year=current_year-2 then cal_date end) as end_date_lly
from cal c
group by cal_month
)
```

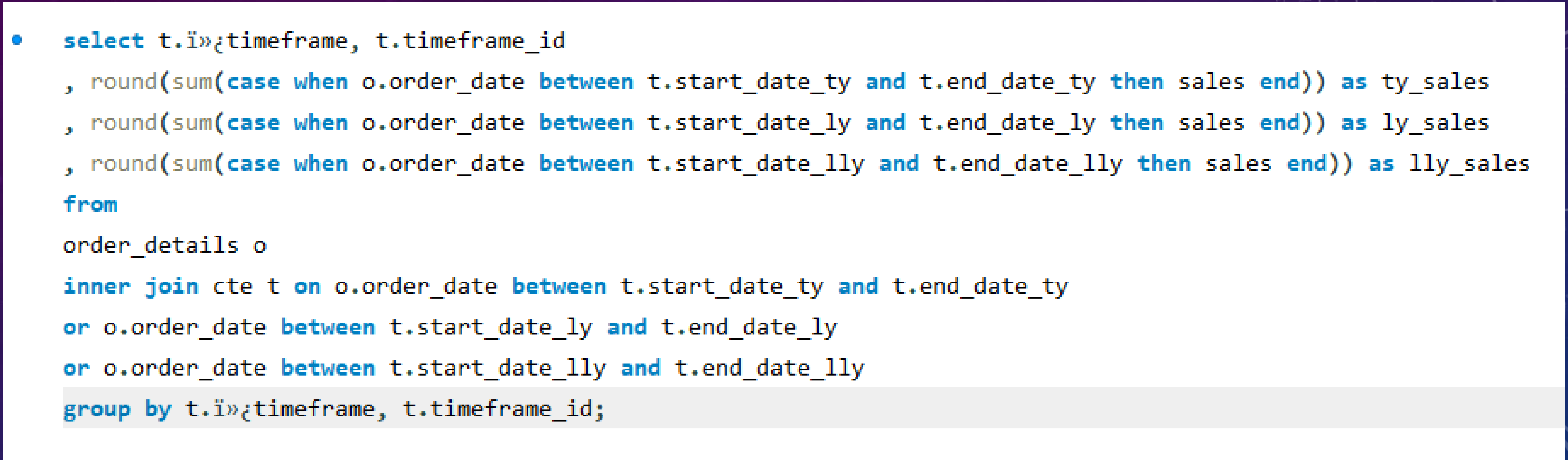


```sql
select * into timeframes from cte;
```

STEP 3– calculate MTD, QTD, YTD FY, etc as per requirement



ï»¿timeframe	timeframe_id	ty_sales	ly_sales	lly_sales
month	12	149579	151120	151120
month	11	147964	147964	147964
month	10	152546	152546	152657
month	9	150176	150176	150176
month	8	152692	152692	152692
month	7	150667	150667	150667
month	6	150161	150161	150936
month	5	151462	151462	151462
month	4	150469	150469	150469
month	3	151508	151508	151195
month	2	147812	143547	143510
month	1	151299	151299	151299
QUARTER	4	153319	153319	153431
QUARTER	3	150211	150211	150211
QUARTER	2	150936	150936	150936
QUARTER	1	151614	151614	151301
FY	FY	154412	154412	154412
MTD	MTD	117466	117466	117466
QTD	QTD	117466	121837	121837
YTD	YTD	117618	121989	121989



```
select t.ï»¿timeframe, t.timeframe_id
, round(sum(case when o.order_date between t.start_date_ty and t.end_date_ty then sales end)) as ty_sales
, round(sum(case when o.order_date between t.start_date_ly and t.end_date_ly then sales end)) as ly_sales
, round(sum(case when o.order_date between t.start_date_lly and t.end_date_lly then sales end)) as lly_sales
from
order_details o
inner join cte t on o.order_date between t.start_date_ty and t.end_date_ty
or o.order_date between t.start_date_ly and t.end_date_ly
or o.order_date between t.start_date_lly and t.end_date_lly
group by t.ï»¿timeframe, t.timeframe_id;
```

THANK YOU

MAYURI DANDEKAR