Transition Test → Oct 8 ↓ Oct 16

10 Q. — 90 mins

5E ✓
3M ✓
2H ✓

Manager → Business Problem (Question)

sales last week ✓

last month

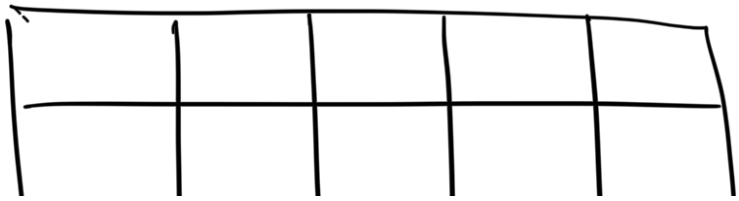
Business Problem → Data Questions

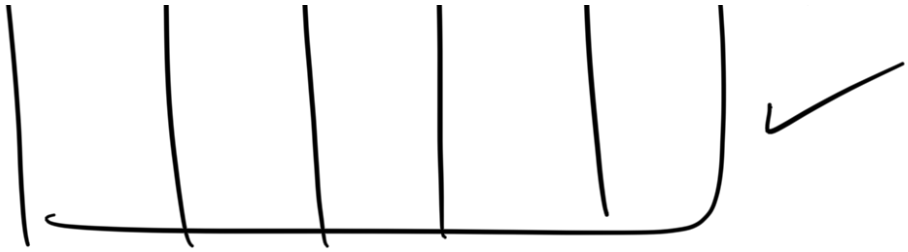
Data Questions → Data Analytics

Data Analytics → Business Insights

Business Insights → Ad-hoc Reporting

Ad-hoc Reporting → Custom Dataset





week, month ← farmer's MKT

CTE

↳ store queries

1. Common Table Expression (CTE)
2. Views

CTEs

Syntax :

→ WITH [query_alias] AS (

[query1]

), [query2_alias] AS (

[query 2] ← Referencing

Top-down

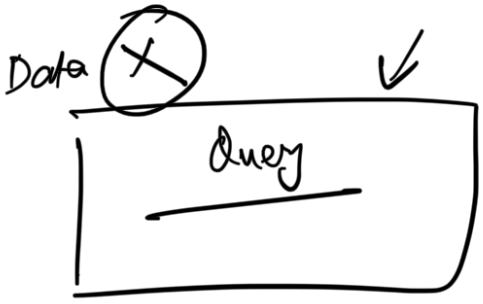
),

⋮

[query_n alias] AS (

[query n]

)



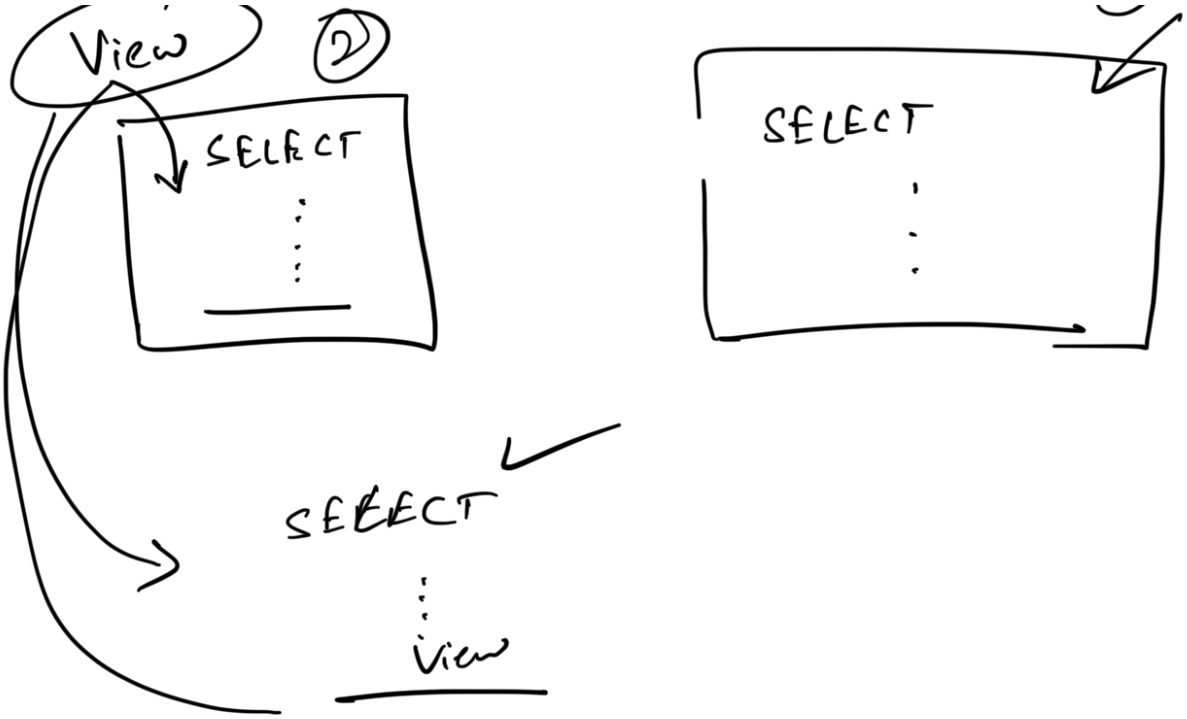
subquery & CTE

SELECT ---

- → Readable
- → structured approach
- → Recursive CTE .

View

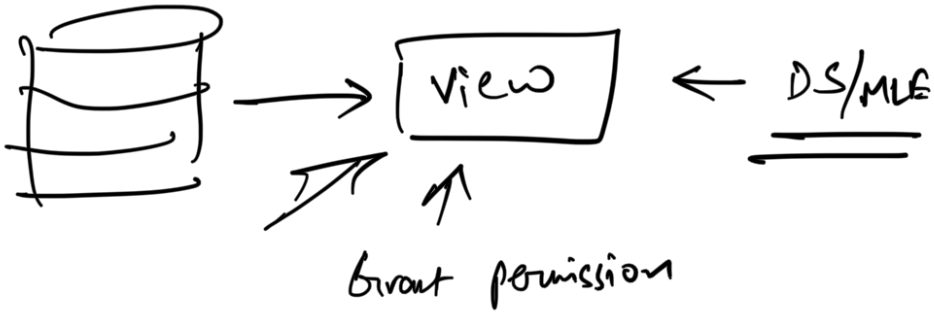
1. Stores the query as a DB view.
2. Dynamically generate the result set
3. Fresh Data.
4. View can take longer



CREATE VIEW DB. AS

CTE vs View — When to use Each one.

1. Ad-hoc queries : CTE (infrequently)
2. Frequently run queries : View



3. Access management Revoke "
View

Stored Procedures

→ Make changes to the Database
  ↳ Alter
  ↳ create

→ ETL pipelines

→ Routines

Syntax:

```
CREATE PROCEDURE ____ ( [P1], [P2] )
Begin
    SQL queries
End
```

→ CALL [Procedure Name] (parameters)

→ 1. IN → user enters the input

→ 2. OUT → Procedure returns the o/p and returns a parameter.

3. INOUT → Both

(IN | OUT | INOUT [param_name] [D.TYPE])

Q: Easy :

Fb_comments_count

Date-time

User-ID	Created-at	Num_of_cmt

Q1. Return the total no. of comments received for each user in the last 30 days. Assume today is 2022-06-15.

2022-05-15

```
SELECT
    user_id
    SUM (counts)
FROM
WHERE  created_at  BETWEEN
                "2022-06-15" AND
                    ↓
                DATE_SUB ("2 , INTERVAL
```

30 DAY)

GROUP BY user_id ;

Q2.

Comment count

User_ID	Created_at	# Comments

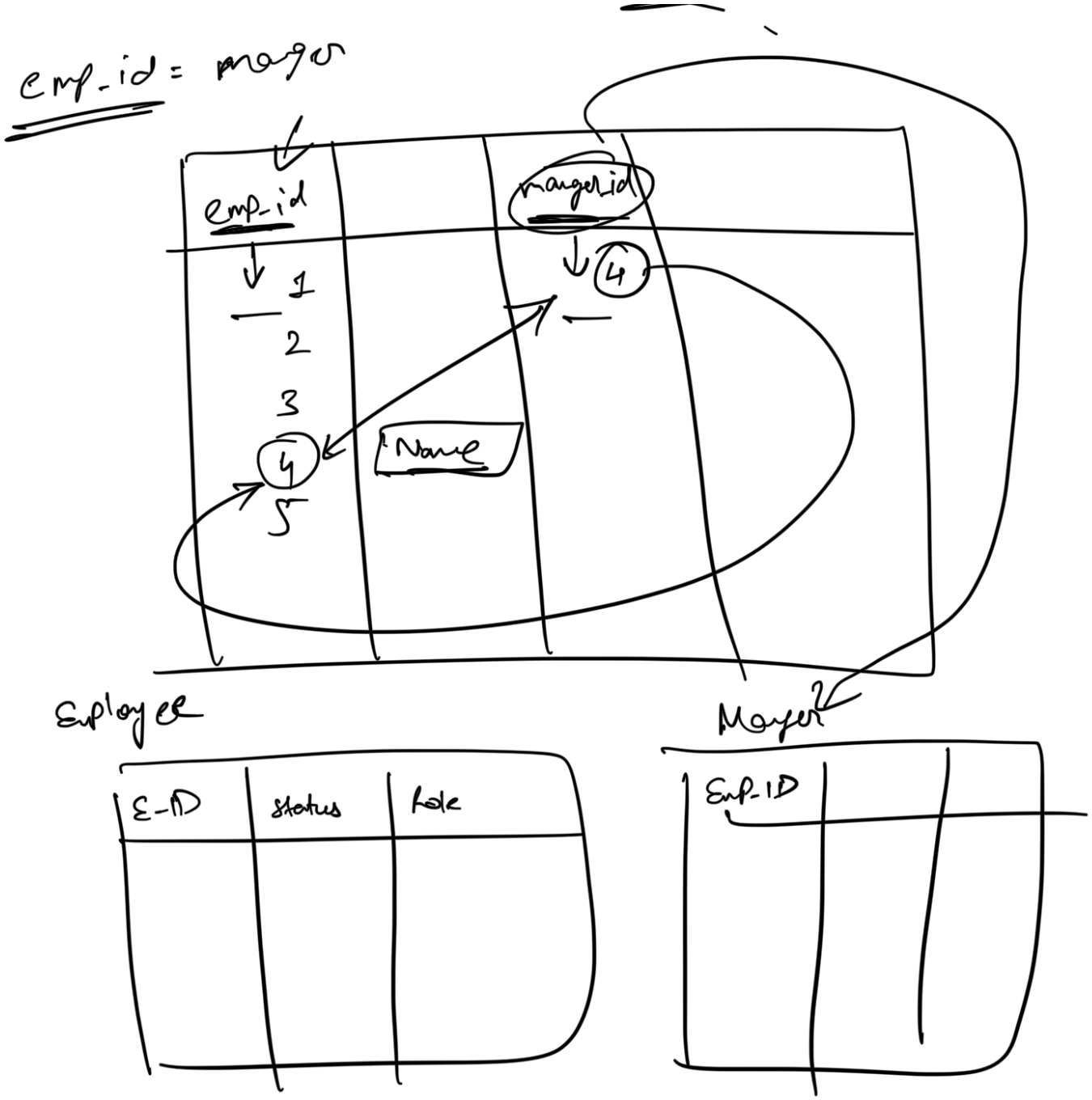
Active Users

User_id	Name	Status	Country

→ Which countries have risen in the rankings based on the number of comments between Dec 2021 vs Jan 2022.

Hint: Avoid gaps between ranks.

USA (Dec)	USA (Jan)
100K (5)	150K (9↑) (8)




```
DELIMITER //

CREATE PROCEDURE all_customers()
BEGIN
	SELECT customer_first_name FROM customer;
END //

DELIMITER ;


-- Parameterised  - IN parameter
```

```
DELIMITER //

CREATE PROCEDURE period_count_sales(IN s_date VARCHAR(15), IN e_date VARCHAR(15))
BEGIN
	SELECT COUNT(*) AS period_sales FROM customer_purchases
	WHERE market_date BETWEEN s_date AND e_date;
END //
DELIMITER ;


CALL period_count_sales("2019-05-01", "2019-05-10")


SELECT COUNT(*) FROM customer_purchases WHERE market_date = "2019-05-04";




-- Parameterised  - OUT parameter
DELIMITER //
CREATE PROCEDURE count_sales_out(OUT total_sales INT, IN m_date varchar(15))
BEGIN
	SELECT COUNT(*) INTO total_sales FROM customer_purchases WHERE market_date =
m_date;
END //
DELIMITER ;


CALL count_sales_out(@total_sales, "")
SELECT @total_sales AS sales_count




-- Step 1: Join the two tables

SELECT *
FROM comment_count AS a
```

```
LEFT JOIN active_users as b
ON a.user_id = b.user_id

-- Step 2 - Filter the data for Dec and Jan

WITH dec_summary AS (
        SELECT *
        FROM comment_count AS a
        LEFT JOIN active_users as b
        ON a.user_id = b.user_id
    WHERE created_at BETWEEN "2021-12-31" AND "2021-12-01"
),

jan_summary AS (
        SELECT *
        FROM comment_count AS a
        LEFT JOIN active_users as b
        ON a.user_id = b.user_id
    WHERE created_at BETWEEN "2022-01-31" AND "2022-01-01"
)

-- Step 3: sum the number of comments per country
WITH dec_summary AS (
        SELECT country,
                SUM(num_comment) AS no_of_comments_dec
        FROM comment_count AS a
        LEFT JOIN active_users as b
        ON a.user_id = b.user_id
    WHERE created_at BETWEEN "2021-12-31" AND "2021-12-01"
    GROUP BY country

),

jan_summary AS (
        SELECT country,
                SUM(num_comment) AS no_of_comments_jan
        FROM comment_count AS a
        LEFT JOIN active_users as b
        ON a.user_id = b.user_id
    WHERE created_at BETWEEN "2022-01-31" AND "2022-01-01"
    GROUP BY country
```

```
)

-- Step 4: Rank 2021 comments count and 2022 jan comment count
WITH dec_summary AS (
        SELECT country,
                SUM(num_comment) AS no_of_comments_dec,
        DENSE_RANK() OVER (ORDER BY SUM(num_comment) DESC) AS country_rank_dec
        FROM comment_count AS a
        LEFT JOIN active_users as b
        ON a.user_id = b.user_id
    WHERE created_at BETWEEN "2021-12-31" AND "2021-12-01"
    GROUP BY country

),

jan_summary AS (
        SELECT country,
                SUM(num_comment) AS no_of_comments_jan,
        DENSE_RANK() OVER (ORDER BY SUM(num_comment) DESC) AS country_rank_jan
        FROM comment_count AS a
        LEFT JOIN active_users as b
        ON a.user_id = b.user_id
    WHERE created_at BETWEEN "2022-01-31" AND "2022-01-01"
    GROUP BY country
)



-- Step 6 - filtering on ranking decline

SELECT
        j.country
FROM jan_summary AS j
LEFT JOIN dec_summary AS d
ON j.country = d.country
WHERE j.country_rank_jan < d.country_rank_dec OR d.country IS NULL
```