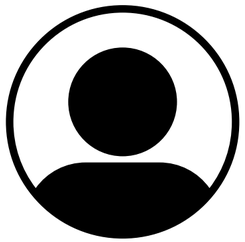
Healthcare Data Analysis Project

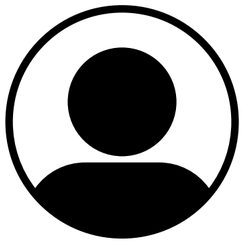
Analyzing Healthcare Data for Insights

By Databricks

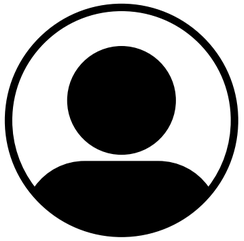
Team Members



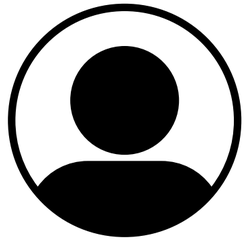
Ritik Patidar



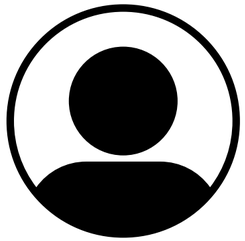
Rishi Gupta



Abhay Dixit



Devansh Panda



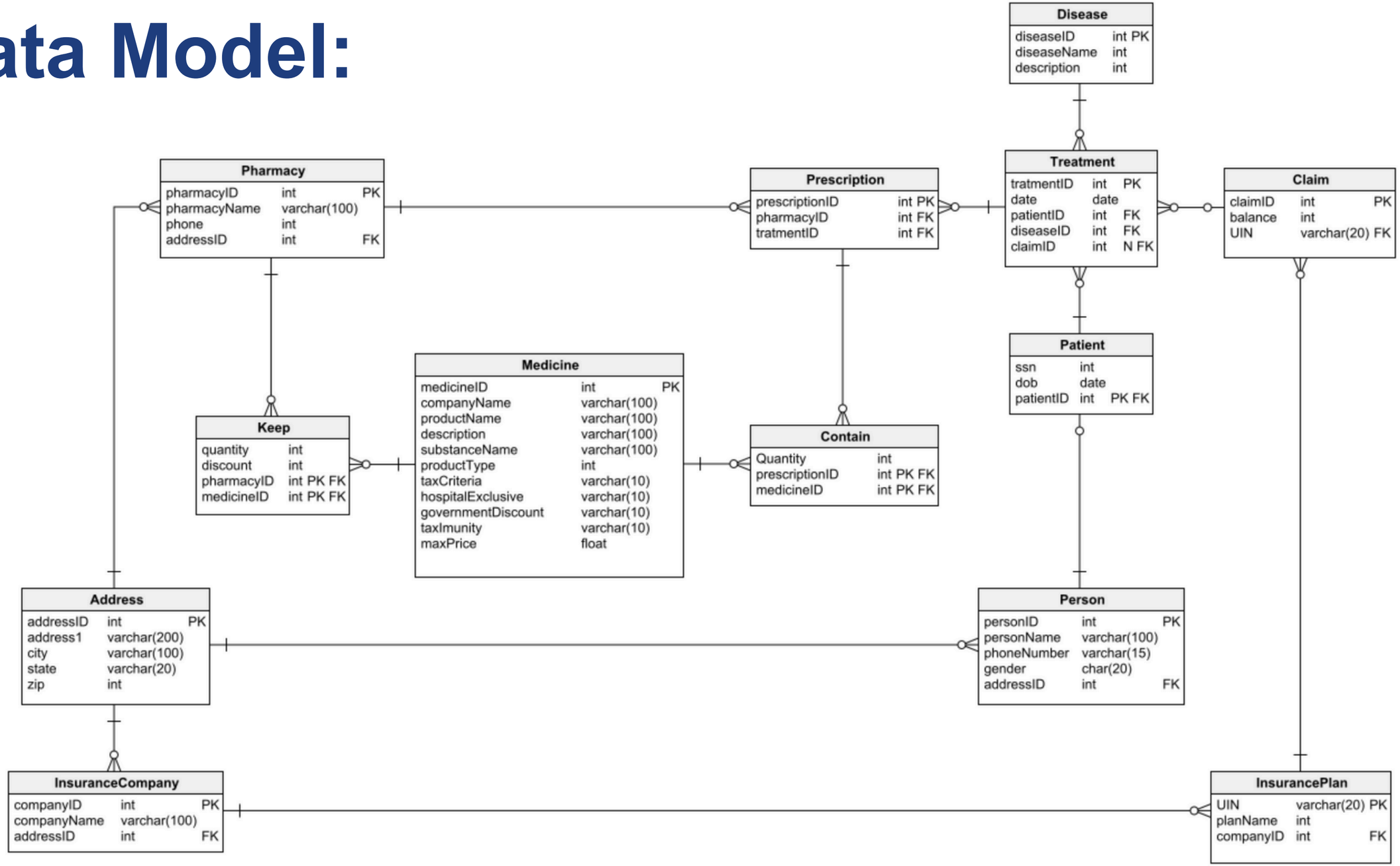
Vibhash Kumar

SIGMOID

Healthcare Data Analysis

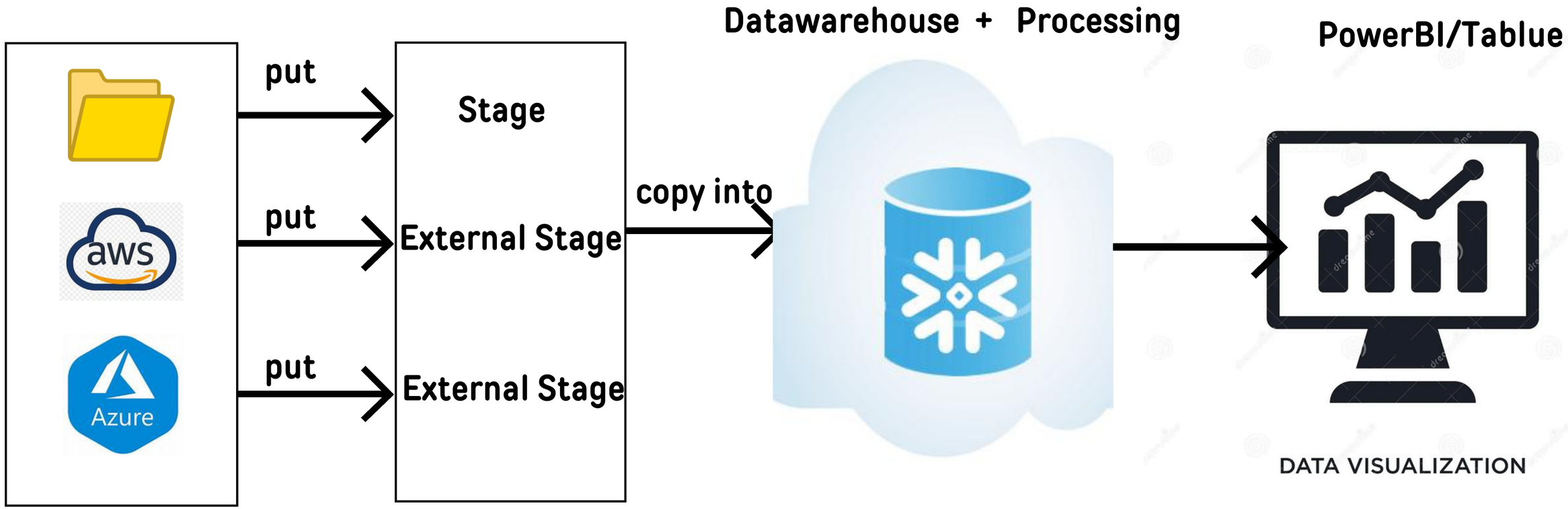
- Leverage data analytics techniques to extract valuable insights
- Identifying trends, patterns, and correlations within the dataset to inform decision-making in the healthcare sector.
- Seek to improve patient care, optimize resource allocation, and enhance overall healthcare outcomes.

Data Model:



SIGMOID

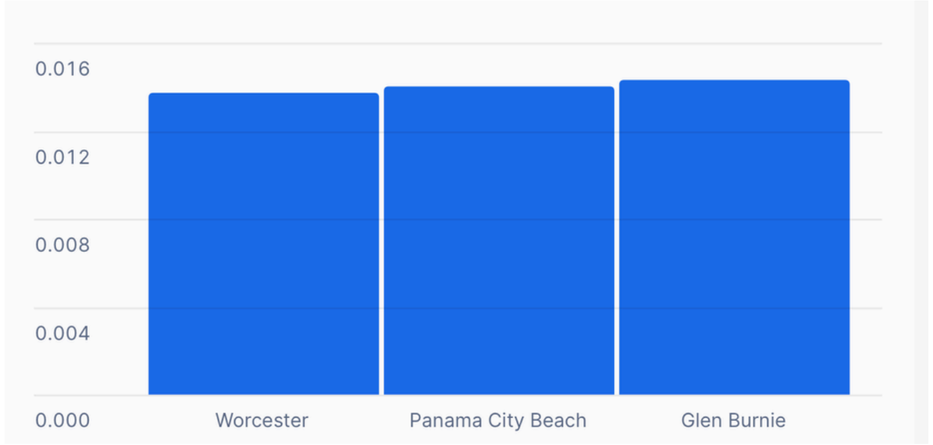
Work Flow:

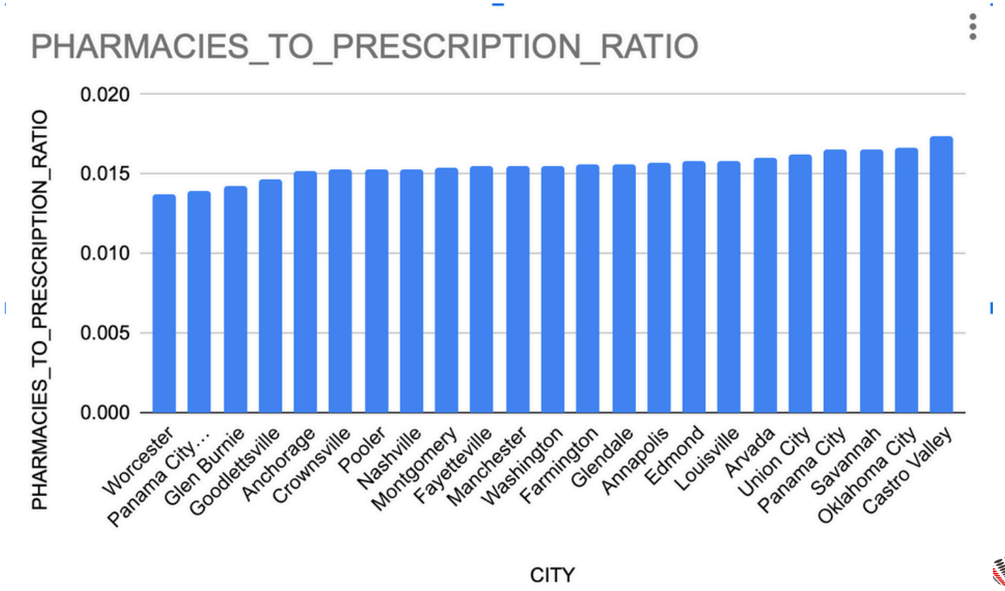


SIGMOID

Problem Statement : A company needs to set up 3 new pharmacies, they have come up with an idea that the pharmacy can be set up in cities where the pharmacy-to-prescription ratio is the lowest and the number of prescriptions should exceed 100. Assist the company to identify those cities where the pharmacy can be set up.

```sql
SELECT
    A.city,
    COUNT(DISTINCT Pr.prescriptionId) AS no_Prescription,
    COUNT(DISTINCT Ph.pharmacyId) / COUNT(DISTINCT Pr.prescriptionId) AS
pharmacies_to_prescription_ratio
FROM
    Address A
JOIN
    Pharmacy Ph ON A.addressID = Ph.addressID
JOIN
    Prescription Pr ON Ph.pharmacyID = Pr.pharmacyID
GROUP BY
    A.city
HAVING
    COUNT(DISTINCT Pr.prescriptionId) > 100
ORDER BY
    pharmacies_to_prescription_ratio
LIMIT
    3;
```





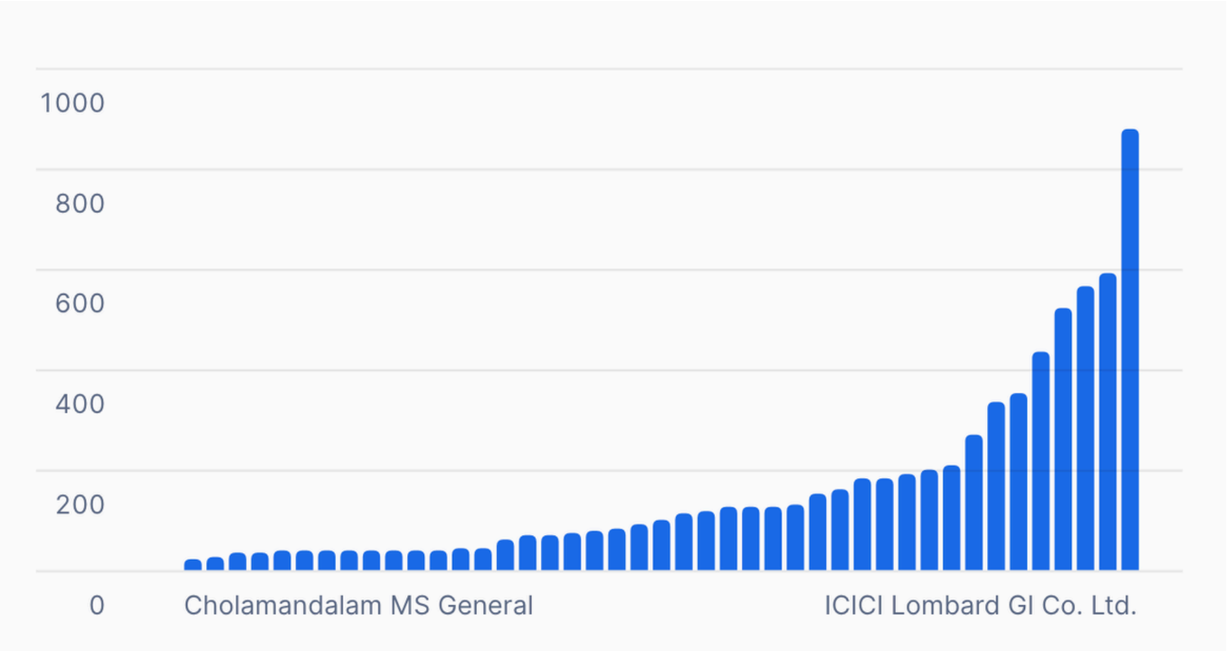
Problem Statement : The State of Alabama (AL) is trying to manage its healthcare resources more efficiently. For each city in their state, they need to identify the disease for which the maximum number of patients have gone for treatment. Assist the state for this purpose. Note: The state of Alabama is represented as AL in Address Table.

```sql
WITH AlabamaPatients AS (
    SELECT
        A.city,
        T.diseaseID,
        D.diseaseName,
        ROW_NUMBER() OVER (PARTITION BY A.city ORDER BY COUNT(*) DESC) AS rn
    FROM
        Address A
    JOIN
        Person P ON A.addressID = P.addressID
    JOIN
        Patient Pt ON P.personID = Pt.patientid
    JOIN
        Treatment T ON Pt.patientID = T.patientID
    JOIN
        Disease D ON T.diseaseID = D.diseaseID
    WHERE
        A.state = 'AL'
    GROUP BY
        A.city, T.diseaseID, D.diseaseName
)
SELECT
    city,
    diseaseID,
    diseaseName
FROM
    AlabamaPatients
WHERE
    rn = 1;
```

	CITY	DISEASEID	DISEASENAME
1	Montgomery	22	Guillain–Barré syndrome
2	Indian Springs Village	10	Bipolar disorder
3	Montevallo	36	Schizophrenia

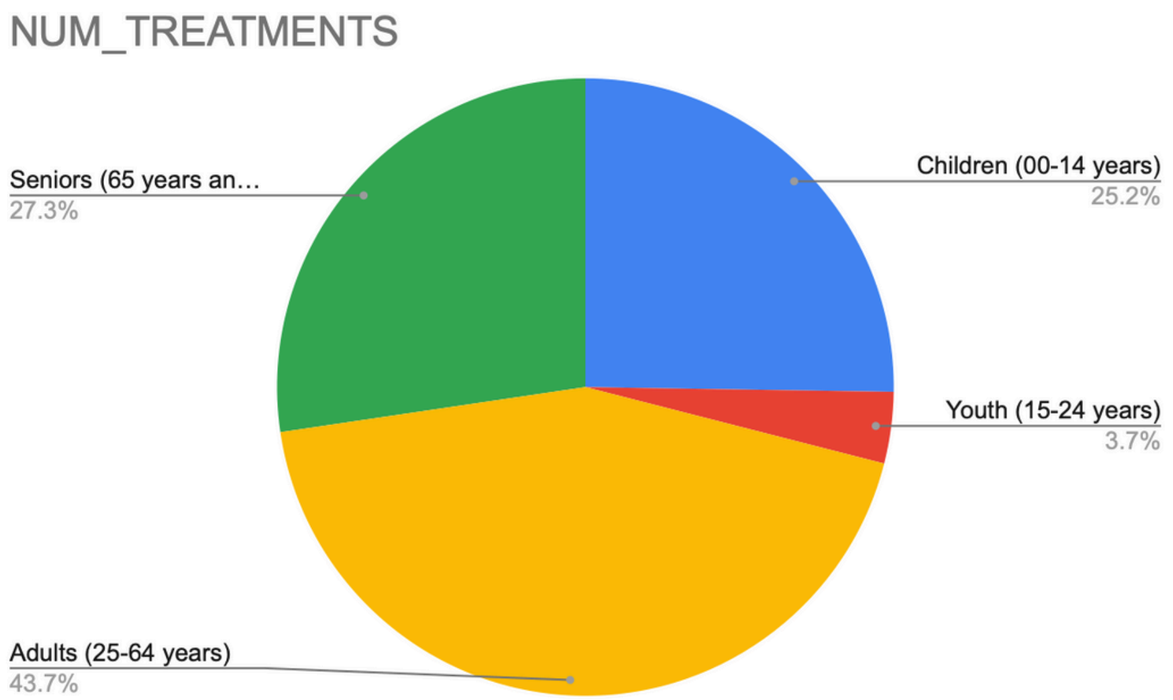
Problem Statement : Insurance companies want to assess the performance of their insurance plans. Generate a report that shows each insurance plan, the company that issues the plan, and the number of treatments the plan was claimed for.

```sql
SELECT
    IP.uin AS insurance_plan_uin,
    IC.companyName AS insurance_company,
    COUNT(T.claimID) AS treatment_count
FROM
    InsurancePlan IP
JOIN
    InsuranceCompany IC
    ON IP.companyID = IC.companyID
LEFT JOIN
    Claim C ON IP.uin = C.uin
LEFT JOIN
    Treatment T ON C.claimID = T.claimID
GROUP BY
    IP.uin,
    IC.companyName;
```



Problem Statement : Jimmy, from the healthcare department, has requested a report that shows how the number of treatments each age category of patients has gone through in the year 2022. The age category is as follows, Children (00-14 years), Youth (15-24 years), Adults (25-64 years), and Seniors (65 years and over). Assist Jimmy in generating the report.

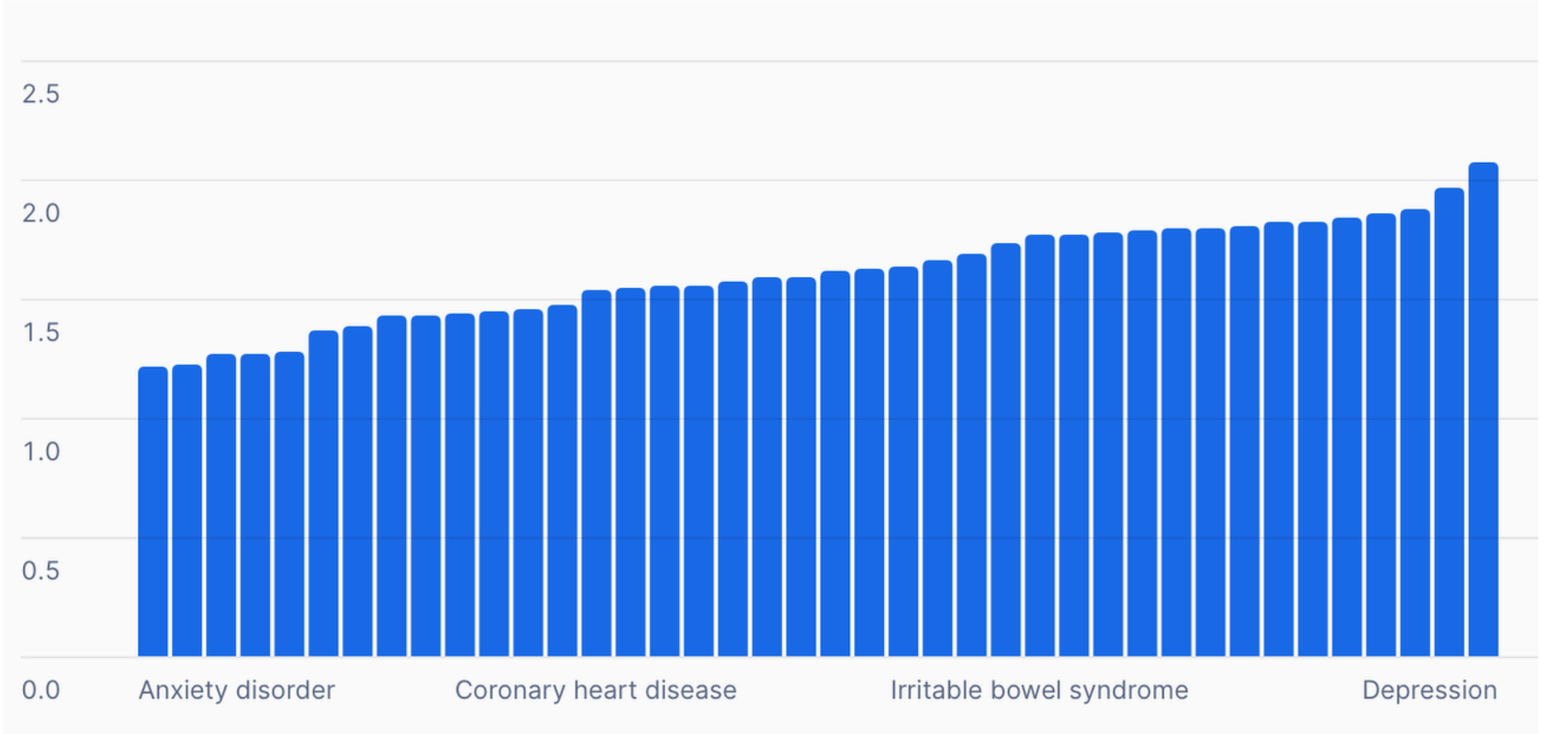
```sql
SELECT
    CASE
        WHEN age <= 14 THEN 'Children (00-14 years)'
        WHEN age <= 24 THEN 'Youth (15-24 years)'
        WHEN age <= 64 THEN 'Adults (25-64 years)'
        ELSE 'Seniors (65 years and over)'
    END AS age_category,
    COUNT(Treatmentid) as num_treatments
FROM
    (SELECT
        Treatmentid,
        DATEDIFF(YEAR, dob, CURRENT_DATE) AS age
    FROM
        TREATMENT
    INNER JOIN
        PATIENT ON TREATMENT.patientid = PATIENT.patientid
    WHERE
        YEAR(date) = 2022
    ORDER BY
        age
    ) AS AgeGroup
GROUP BY
    age_category;
```



SIGMOID

Problem Statement : Jimmy, from the healthcare department, wants to know which disease is infecting people of which gender more often. Assist Jimmy with this purpose by generating a report that shows for each disease the male-to-female ratio. Sort the data in a way that is helpful for Jimmy.

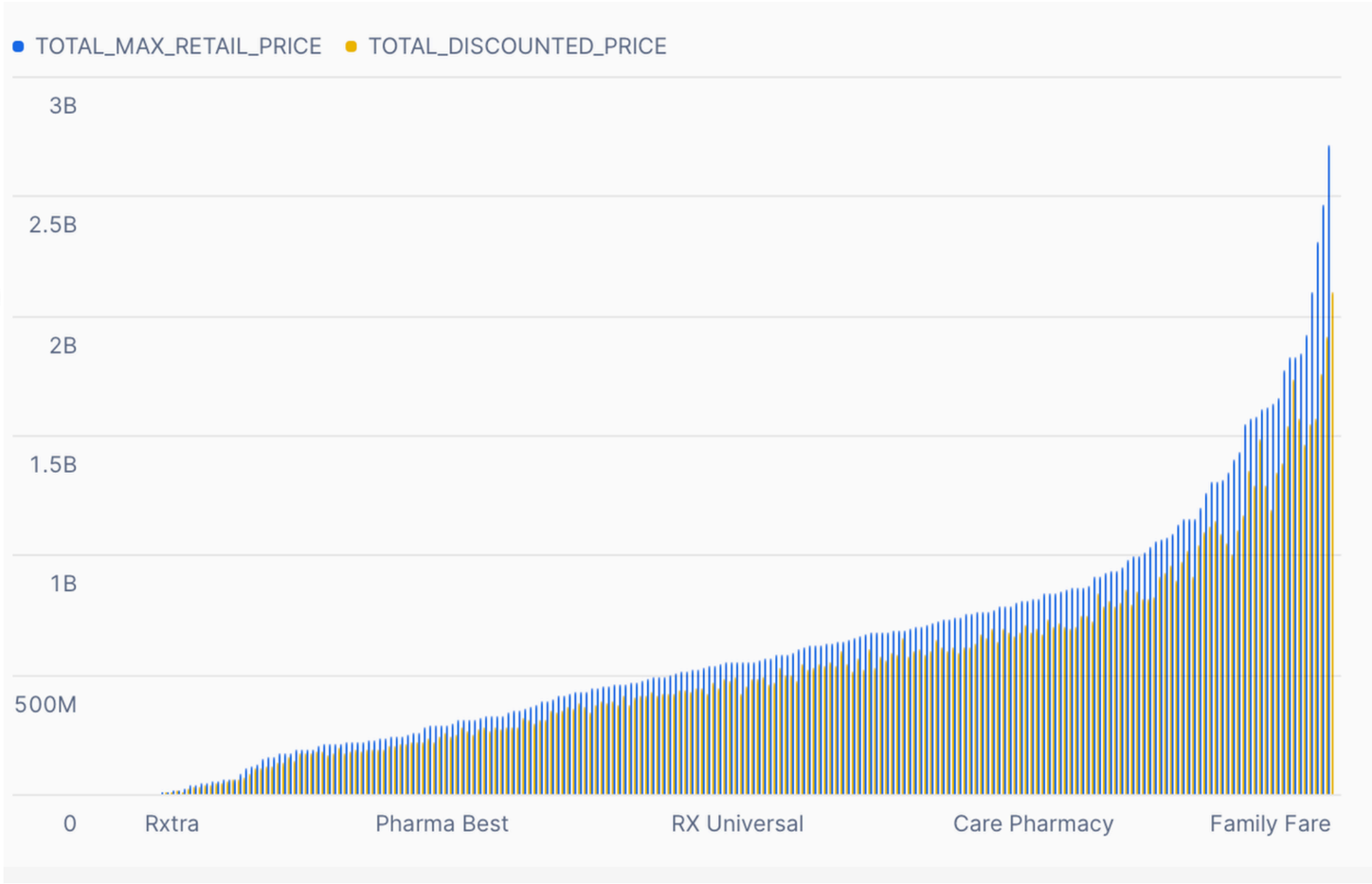
```sql
WITH DISEASE_GENDER_COUNT AS (
    SELECT
        DISEASE.DISEASEID,
        DISEASE.DISEASENAME,
        GENDER,
        COUNT(PERSONID) AS COUNT
    FROM
        PERSON
    JOIN PATIENT ON PERSON.PERSONID = PATIENT.PATIENTID
    JOIN TREATMENT ON PATIENT.PATIENTID = TREATMENT.PATIENTID
    JOIN DISEASE ON DISEASE.DISEASEID = TREATMENT.DISEASEID
    GROUP BY
        DISEASE.DISEASEID,
        DISEASE.DISEASENAME,
        GENDER
    ORDER BY
        DISEASE.DISEASEID,
        DISEASE.DISEASENAME,
        GENDER DESC
)

SELECT
    T1.DISEASEID,
    T2.DISEASENAME,
    T1.COUNT AS MALECOUNT,
    T2.COUNT AS FEMALECOUNT,
    T1.COUNT / T2.COUNT AS MALE_TO_FEMALE_RATIO
FROM
    DISEASE_GENDER_COUNT T1
JOIN
    DISEASE_GENDER_COUNT T2 ON T1.DISEASEID = T2.DISEASEID
WHERE
    T1.GENDER = 'male'
    AND T2.GENDER = 'female'
ORDER BY
    MALE_TO_FEMALE_RATIO DESC;
```



SIGMOID

Problem Statement : The Healthcare department wants a report about the inventory of pharmacies. Generate a report on their behalf that shows how many units of medicine each pharmacy has in their inventory, the total maximum retail price of those medicines, and the total price of all the medicines after discount.Note: discount field in keep signifies the percentage of discount on the maximum price.

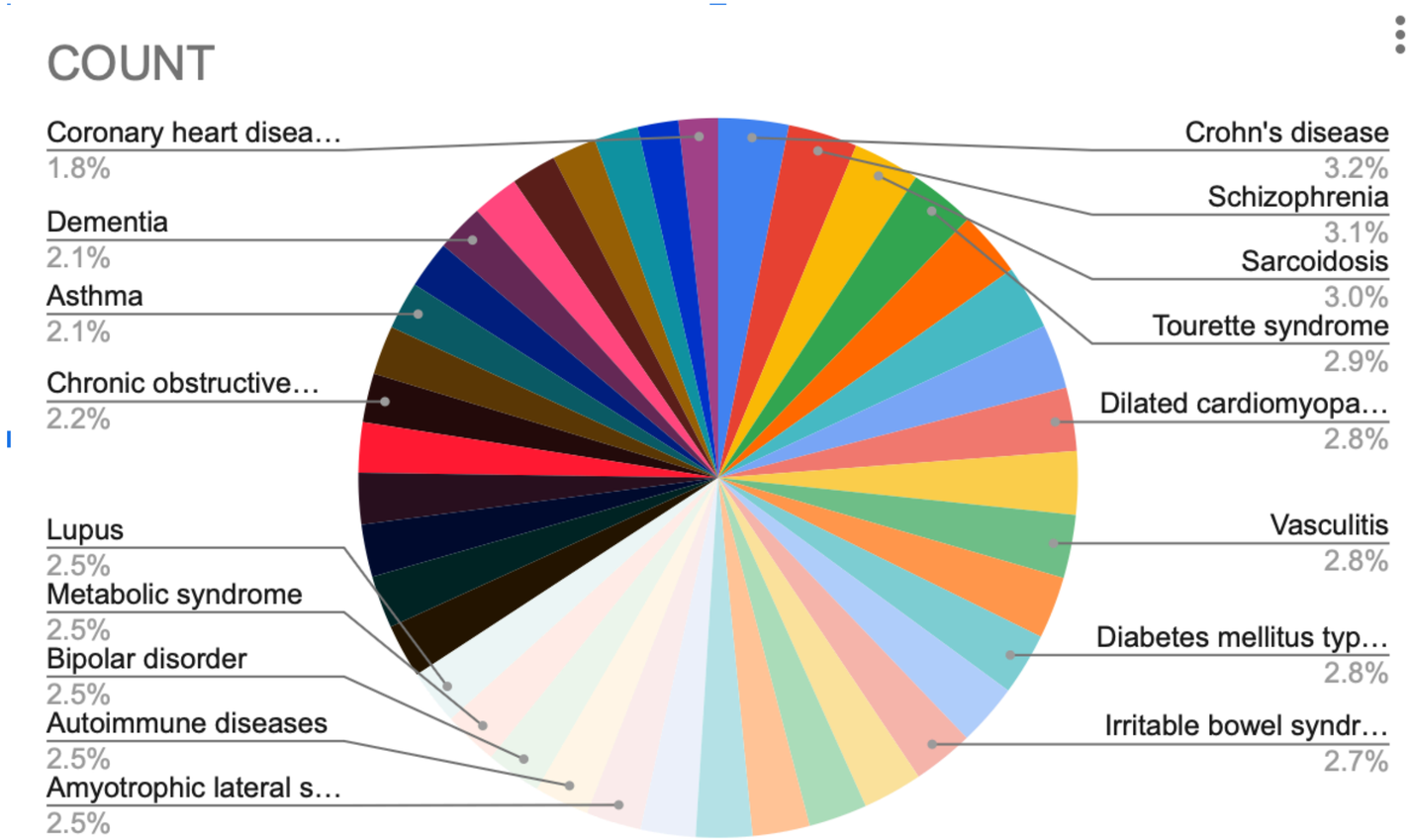
```sql
SELECT
    K.pharmacyID,
    P.pharmacyName,
    SUM(K.quantity) AS total_units,
    SUM(M.maxPrice * K.quantity)
        AS total_max_retail_price,
    SUM((M.maxPrice * (100 - K.discount) / 100) * K.quantity)
        AS total_discounted_price
FROM
    Keep K
JOIN
    Medicine M ON K.medicineID = M.medicineID
JOIN
    Pharmacy P ON K.pharmacyID = P.pharmacyID
GROUP BY
    K.pharmacyID,
    P.pharmacyName;
```



SIGMOID

Problem Statement : The Healthcare department wants to know which disease is most likely to infect multiple people in the same household. For each disease find the number of households that has more than one patient with the same disease. Note: 2 people are considered to be in the same household if they have the same address.

```sql
WITH household_disease AS (
    SELECT
        t.diseaseid AS diseaseid,
        d.diseaseName AS diseaseName,
        p.addressID AS addressid,
        COUNT(p.personID) AS cnt
    FROM
        treatment t
    JOIN
        patient pt ON pt.patientid = t.patientid
    JOIN
        person p ON p.personid = pt.patientid
    JOIN
        disease d ON d.diseaseid = t.diseaseid
    GROUP BY
        1, 2, 3
    HAVING
        cnt >= 2
)
SELECT
    diseaseid,
    diseaseName,
    COUNT(DISTINCT addressid) AS count
FROM
    household_disease
GROUP BY
    diseaseid, diseaseName
ORDER BY
    count DESC;
```



QUERY OPTIMISATION

Using Indexing

Join and Join Order Optimization

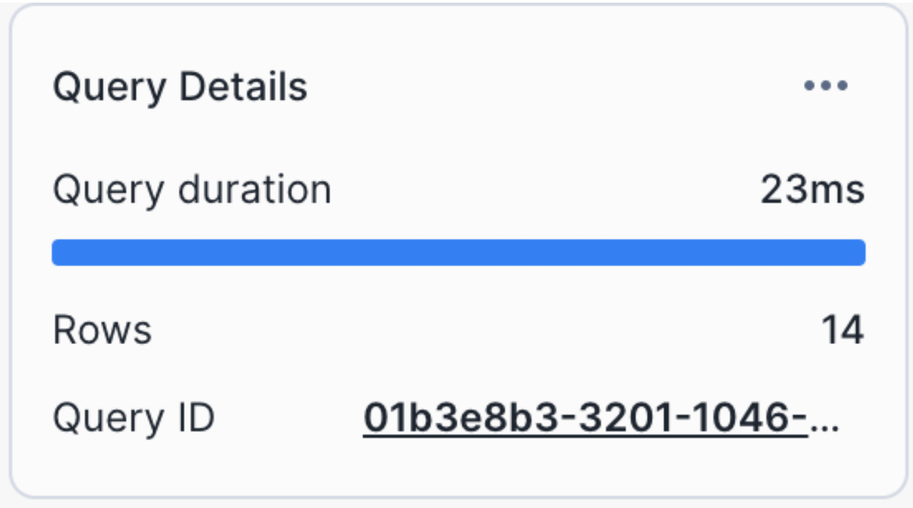
Useage of CTEs

Using Aggregate Functions Wisely

Other Techniques

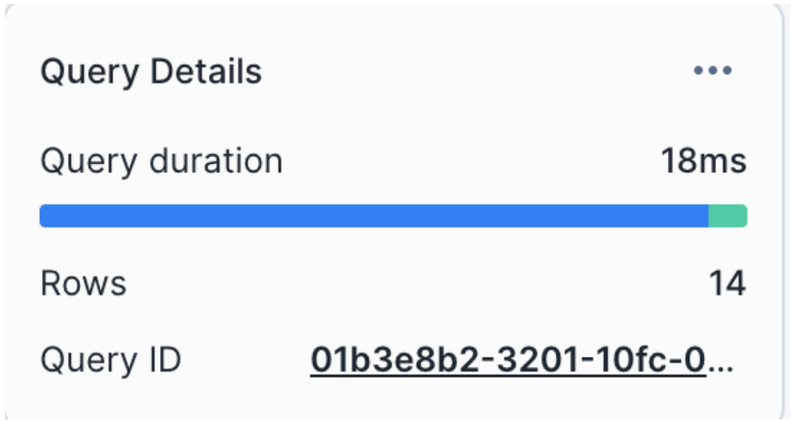
Non-Optimised Code

```sql
SELECT Disease.diseaseName, COUNT(*) as numClaims
FROM Disease
JOIN Treatment ON Disease.diseaseID = Treatment.diseaseID
JOIN Claim On Treatment.claimID = Claim.claimID
WHERE diseaseName IN (SELECT diseaseName from Disease where diseaseNam
GROUP BY diseaseName;
```



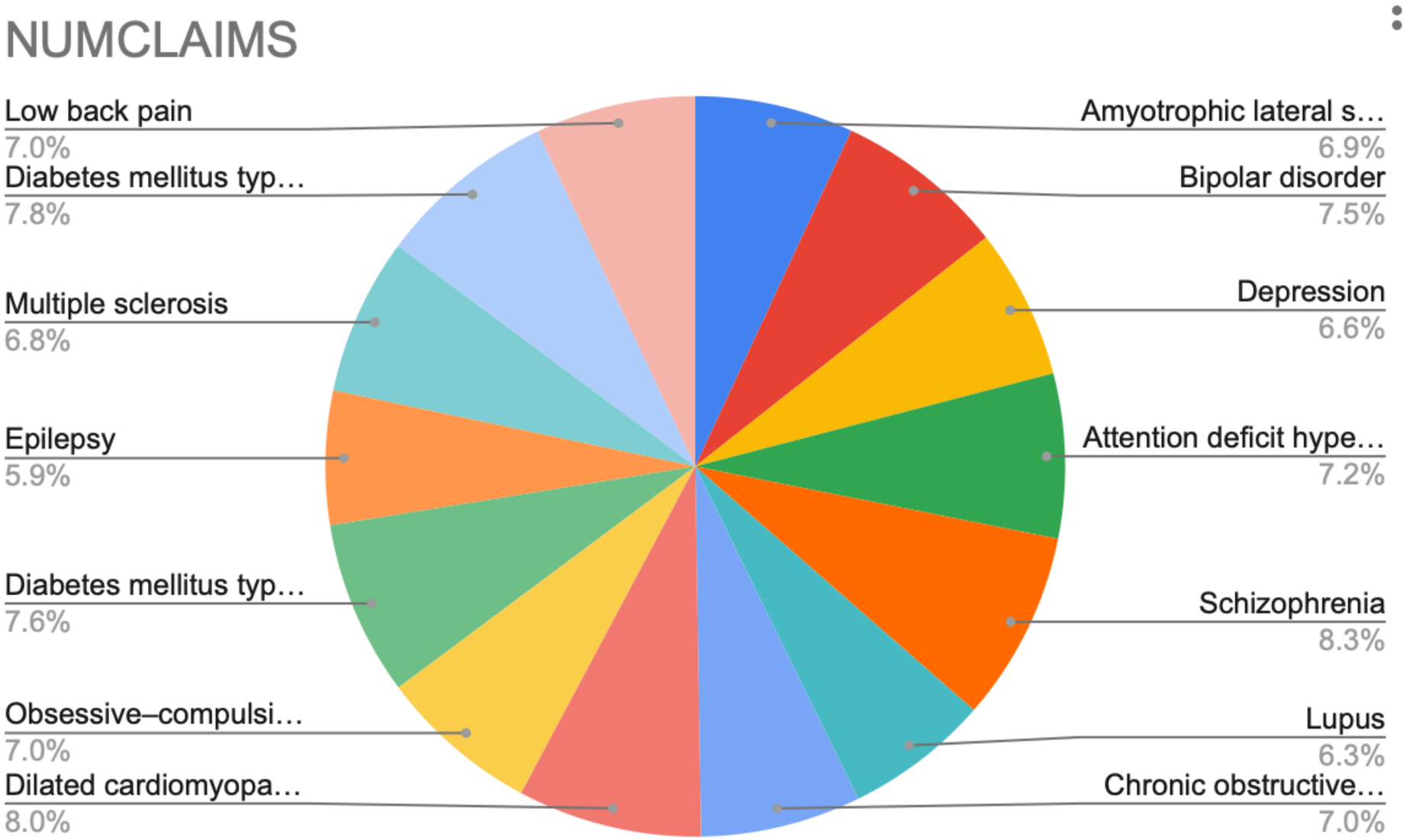
Optimised Code

```sql
SELECT Disease.diseaseName, COUNT(*) as numClaims
FROM Disease
JOIN Treatment ON Disease.diseaseID = Treatment.diseaseID
JOIN Claim On Treatment.claimID = Claim.claimID
WHERE diseaseName like '%p%'
GROUP BY diseaseName;
```



SIGMOID

Visualised Data



SIGMOID

USE CASE OF CLUSTERING

PROBLEM STATEMENT

```
drop table if exists T1;

select Pharmacy.pharmacyID, Prescription.prescriptionID, sum(quantity) as totalQuantity

into T1

from Pharmacy

join Prescription on Pharmacy.pharmacyID = Prescription.pharmacyID

join Contain on Contain.prescriptionID = Prescription.prescriptionID

join Medicine on Medicine.medicineID = Contain.medicineID

join Treatment on Treatment.treatmentID = Prescription.treatmentID

where YEAR(date) = 2022

group by Pharmacy.pharmacyID, Prescription.prescriptionID

order by Pharmacy.pharmacyID, Prescription.prescriptionID;

select * from T1

where totalQuantity > (select avg(totalQuantity) from T1);
```

Solution

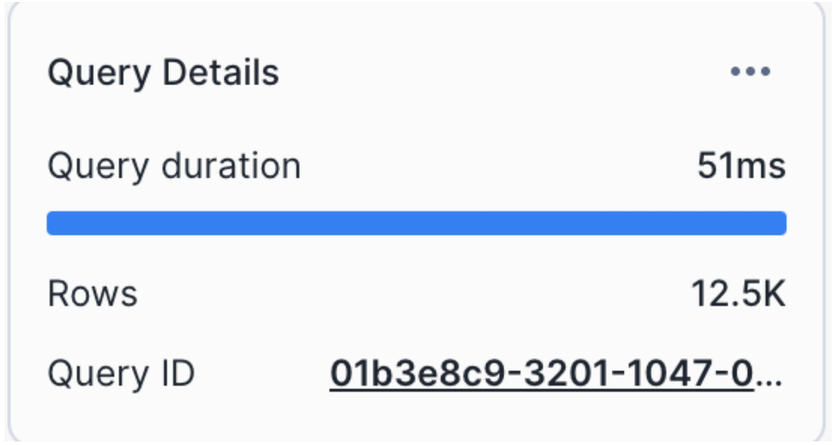
```
SELECT
     P.prescriptionID,
    SUM(C.QUANTITY)as TOTAL_QUANTITY
FROM
    Prescription P
LEFT JOIN
    Contain C ON P.prescriptionID = C.prescriptionID
GROUP BY
    P.PRESCRIPTIONID
HAVING
    SUM(C.QUANTITY) > (SELECT AVG(quantity) FROM Contain)
ORDER BY
     P.PRESCRIPTIONID;
```

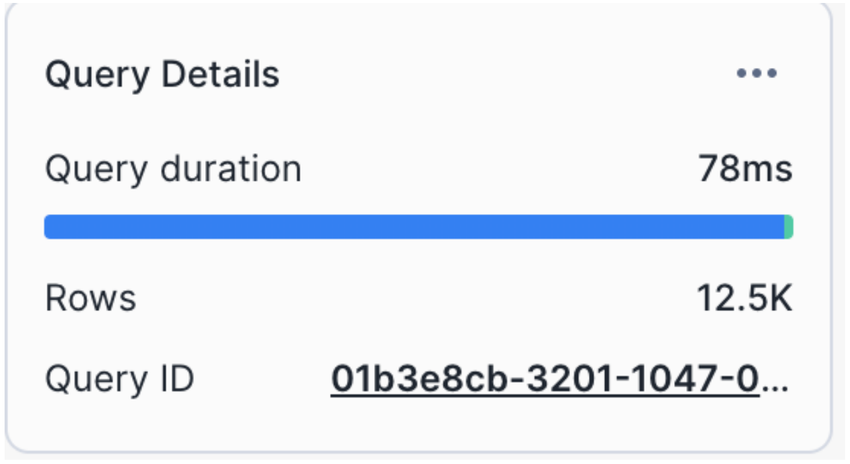

SIGMOID

CLUSTERING

Snowflake clustering is a technique employed in Snowflake tables to group related rows together within the same micro-partition, thereby enhancing query performance for accessing these rows

Clustered



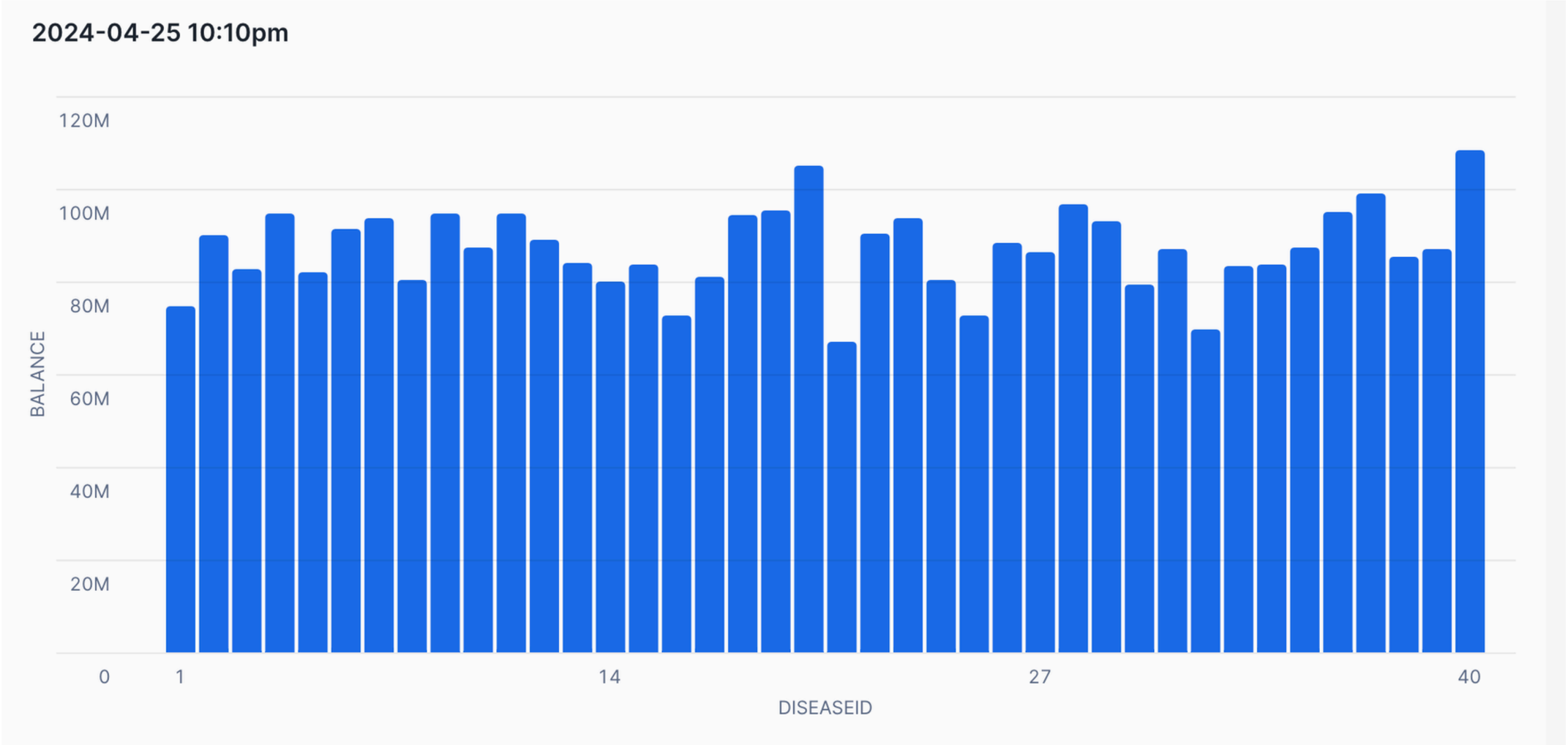


Non- clustered

Problem Statement : Insurance companies want to know if a disease is claimed higher or lower than average. Write a stored procedure that returns “claimed higher than average” or “claimed lower than average” when the diseaseID is passed to it.

```
CREATE OR REPLACE PROCEDURE CheckBalanceStatus(diseaseID INT)
RETURNS VARCHAR
LANGUAGE SQL
AS
DECLARE
    avg_balance FLOAT;
    current_balance FLOAT;
    res VARCHAR(100);
BEGIN
    -- Calculate the average balance for the given diseaseID
    WITH temp AS (
        SELECT TREATMENT.DISEASEID AS diseaseid, AVG(CLAIM.balance) AS avg_balance
        FROM CLAIM
        JOIN TREATMENT ON CLAIM.CLAIMID = TREATMENT.CLAIMID
        GROUP BY TREATMENT.DISEASEID
    )
    SELECT avg_balance INTO avg_balance FROM temp WHERE diseaseid = diseaseID;

    -- Get the current balance for the passed diseaseID
    SELECT SUM(CLAIM.balance) INTO current_balance
    FROM CLAIM
    JOIN TREATMENT ON CLAIM.CLAIMID = TREATMENT.CLAIMID
    WHERE TREATMENT.DISEASEID = diseaseID;

    -- Compare current balance with the average
    IF (current_balance > avg_balance) THEN
        res := 'Balance higher than average';
    ELSE
        res := 'Balance lower than or equal to average';
    END IF;

    -- Return the result
    RETURN res;
END;
```

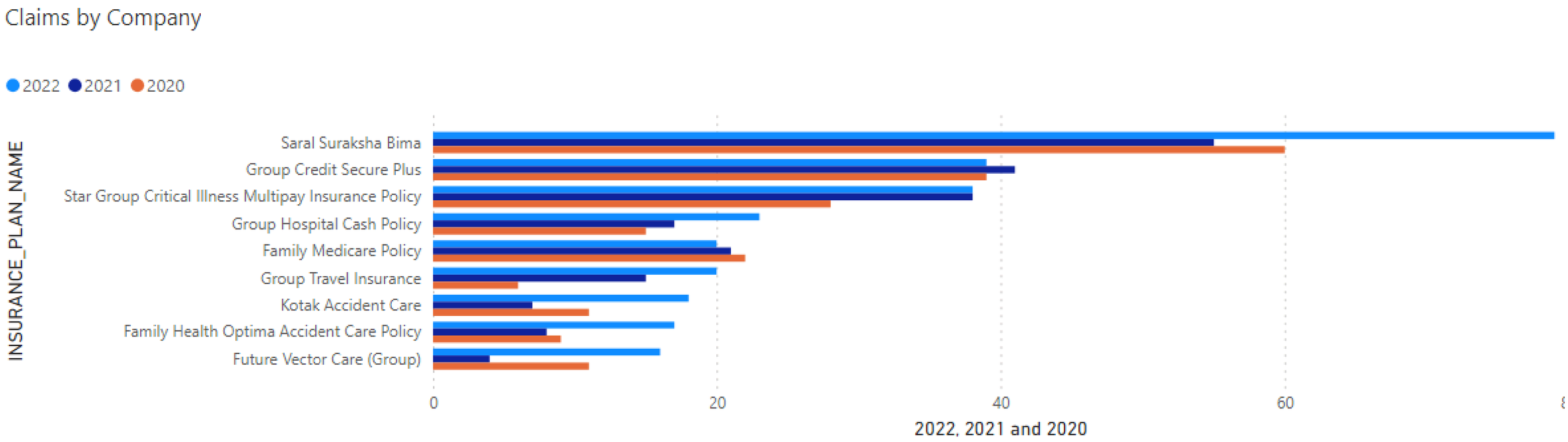
Visualisation



Problem Statement: Insurance companies want to evaluate the performance of different insurance plans they offer. Generate a report that shows each insurance plan, the company that issues the plan, and the number of treatments the plan was claimed for. The report would be more relevant if the data compares the performance for different years(2020, 2021 and 2022) and if the report also includes the total number of claims in the different years, as well as the total number of claims for each plan in all 3 years combined.

```sql
SELECT
    IP.uin AS Insurance_Plan_UIN,
    IP.planName AS Insurance_Plan_Name,
    IC.companyName AS Company_Name,
    SUM(CASE WHEN YEAR(T.date) = 2020 THEN 1 ELSE 0 END) AS Claims_2020,
    SUM(CASE WHEN YEAR(T.date) = 2021 THEN 1 ELSE 0 END) AS Claims_2021,
    SUM(CASE WHEN YEAR(T.date) = 2022 THEN 1 ELSE 0 END) AS Claims_2022,
    COUNT(CL.claimID) AS Total_Claims
FROM
    InsurancePlan IP
JOIN
    InsuranceCompany IC ON IP.companyID = IC.companyID
JOIN
    Claim CL ON IP.uin = CL.uin
JOIN
    Treatment T ON CL.claimID = T.claimID
GROUP BY
    IP.uin, IP.planName, IC.companyName
ORDER BY
    Insurance_Plan_UIN;
```

Visualised Data



Conclusion

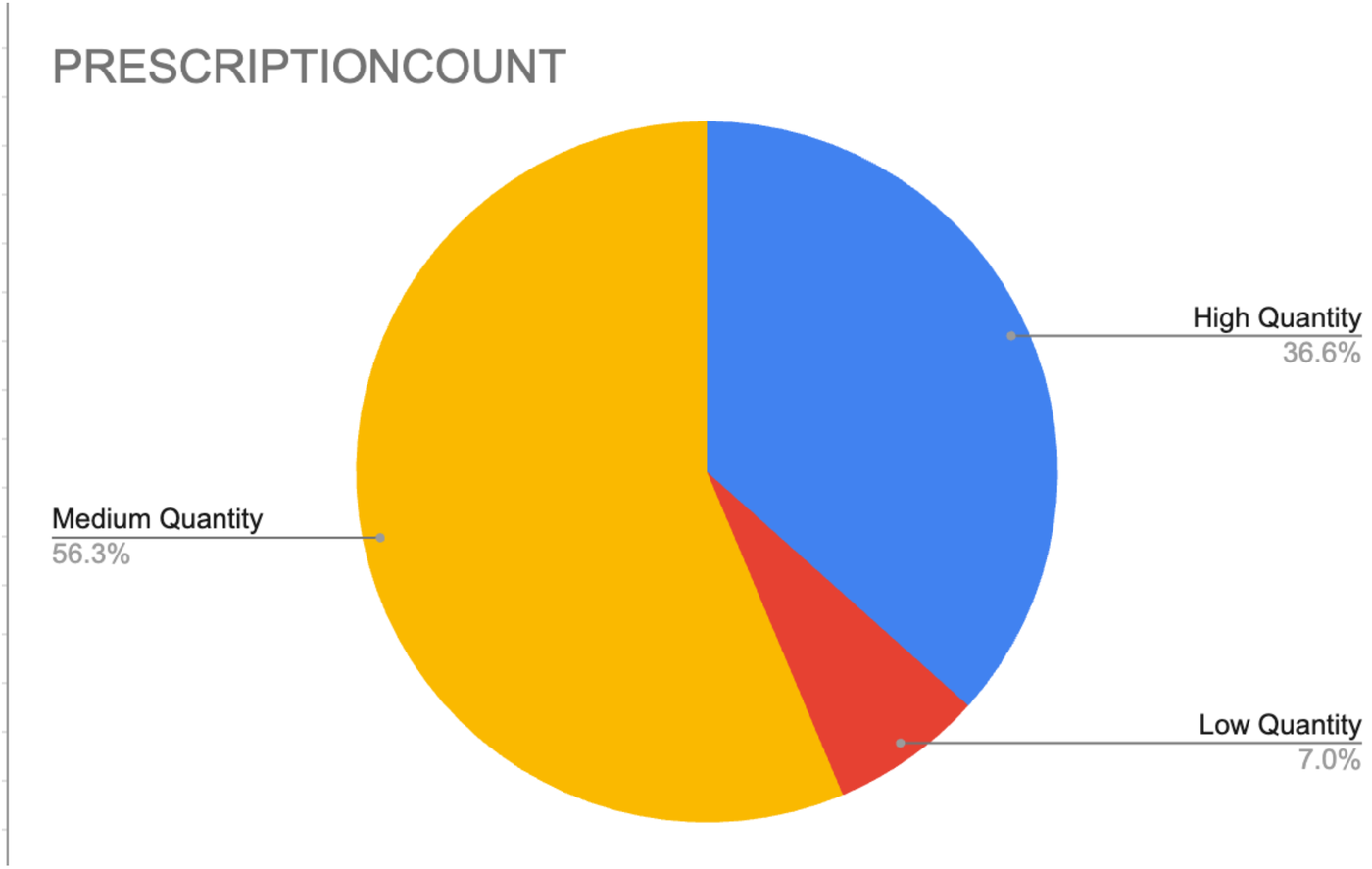
We can see most of the policies have higher claims on 2022

We also can observe in most polices claims were increasing year wise

Problem Statement: 'Ally Scripts' pharmacy company wants to find out the quantity of medicine prescribed in each of its prescriptions. Write a query that finds the sum of the quantity of all the medicines in a prescription and if the total quantity of medicine is less than 20 tag it as “low quantity”. If the quantity of medicine is from 20 to 49 (both numbers including) tag it as “medium quantity“ and if the quantity is more than equal to 50 then tag it as “high quantity”. Show the prescription Id, the Total Quantity of all the medicines in that prescription, and the Quantity tag for all.

```sql
SELECT
    p.prescriptionID,
    SUM(c.Quantity) AS totalQuantity,
    CASE
        WHEN SUM(c.Quantity) < 20 THEN 'Low Quantity'
        WHEN SUM(c.Quantity) BETWEEN 20 AND 49 THEN 'Medium Quantity'
        ELSE 'High Quantity'
    END AS Tag
FROM
    prescription p
JOIN
Pharmacy ph ON p.pharmacyID = ph.pharmacyID
JOIN
    contain c ON p.prescriptionID = c.prescriptionID
JOIN
    Medicine m ON c.medicineID = m.medicineID
WHERE
    ph.pharmacyName = 'Ally Scripts'
GROUP BY
    p.prescriptionID
ORDER BY
    Tag;
```

Visualised Data



Conclusion

Our integration of Snowflake and Power BI optimized analytics across healthcare, insurance, and pharmacy, delivering real-time insights and enhancing operational efficiency while maintaining rigorous security. This project marks a transformative leap in data-driven strategies for healthcare management.

Thank You!

FROM
TEAM DATABRICKS