The Disease Model

Team 6:
Sumant Kumar Gupta
Shashank Shekhar
Regina Thapa

Business problem

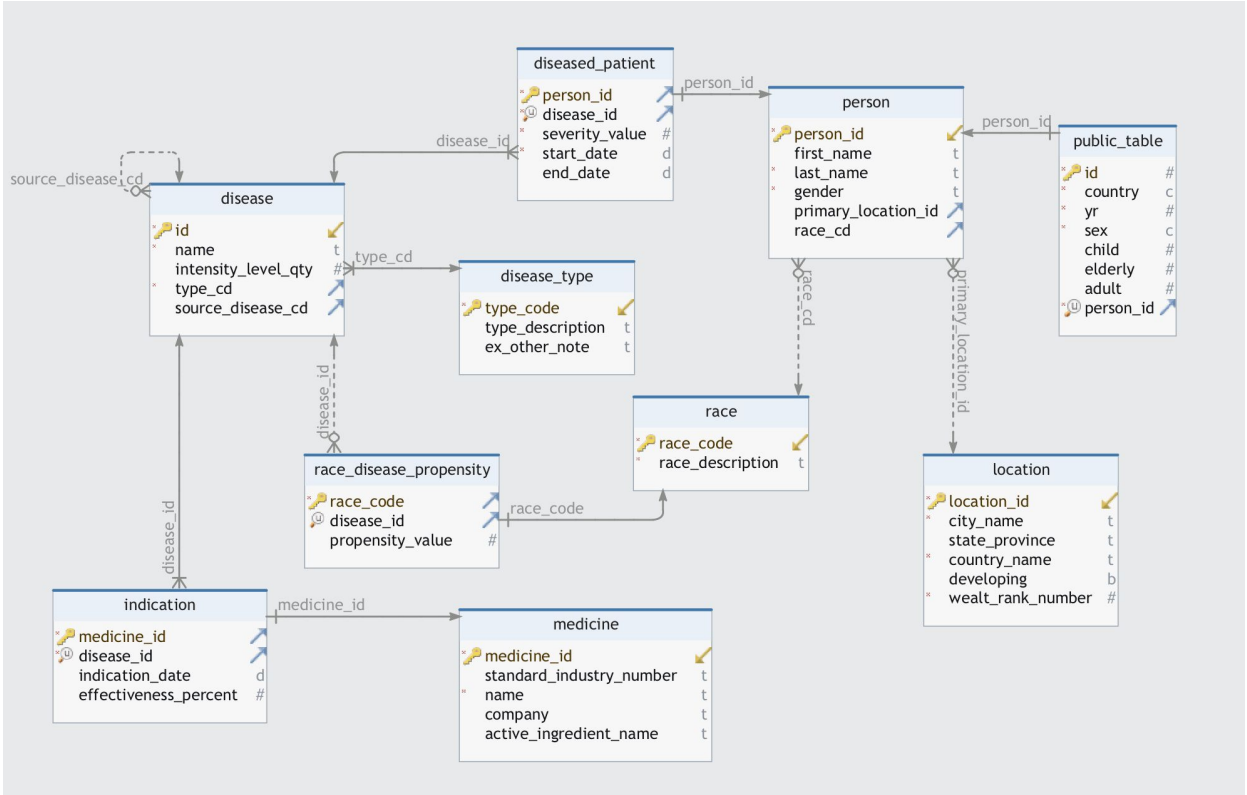
Analyzing disease trends and medication effectiveness across different demographic groups.

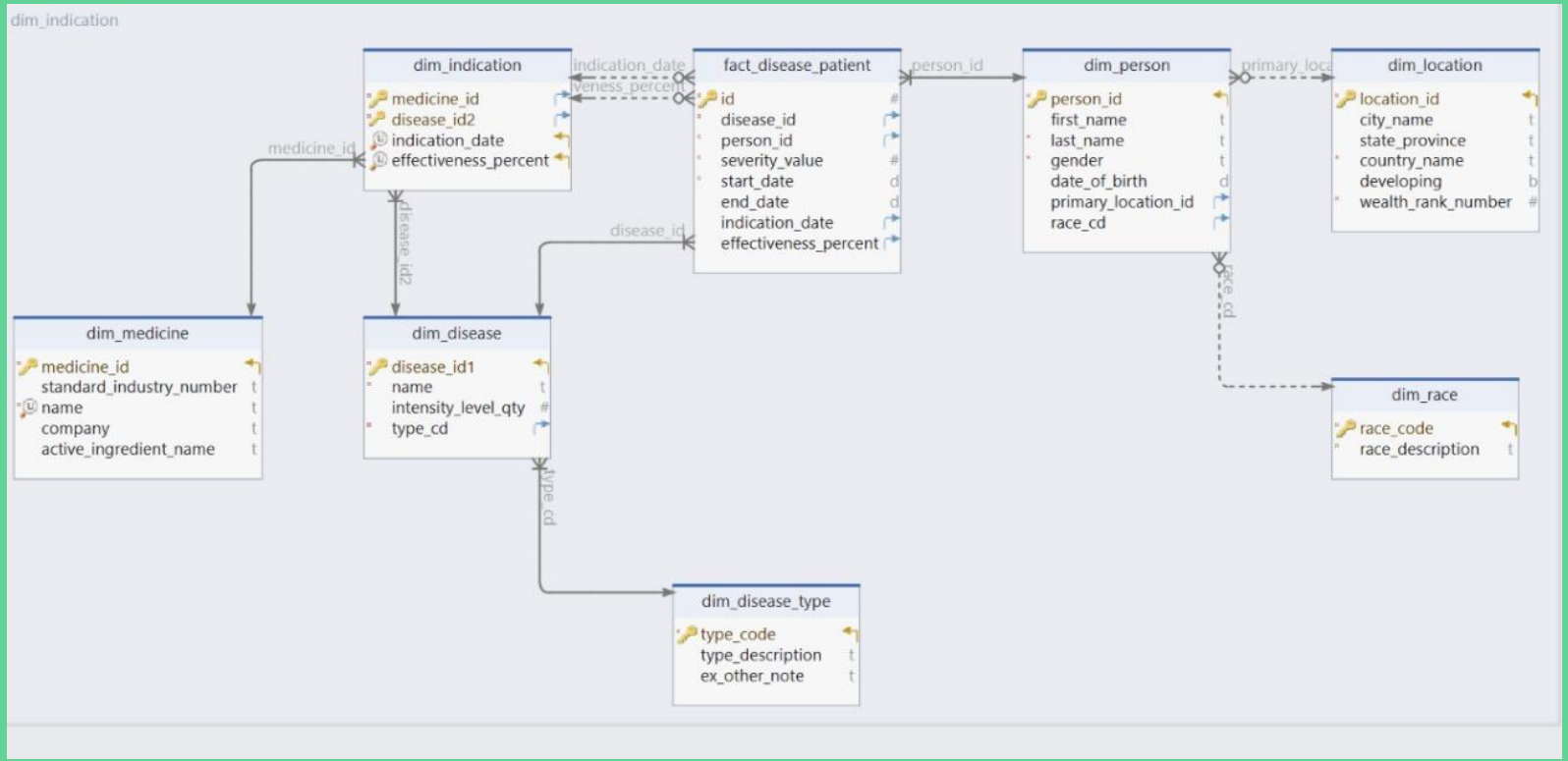
This involve studying the presence of various diseases among different races, genders, and age groups.

By analyzing this data, healthcare organizations and pharmaceutical companies can gain insights into disease patterns, medication efficacy, and patient demographics, ultimately informing decision-making processes related to healthcare policy, treatment strategies, and resource allocation.

The disease EER diagram

10 Tables with relationships





The Fact disease model

Data Loading

- Mock data to insert in tables.
- Dbschema for ER diagrams
- pgAdmin to run queries to upload.

We created **Triggers** to automatically log the changes in the disease patient audit table.

```
-- Trigger in public schema
CREATE OR REPLACE FUNCTION public.log_diseased_patient_changes()
RETURNS TRIGGER AS
$$
BEGIN
    IF TG_OP = 'INSERT' THEN
        INSERT INTO diseased_patient_audit (action, patient_id, action_timestamp)
        VALUES ('INSERT', NEW.id, NOW());
    ELSIF TG_OP = 'UPDATE' THEN
        INSERT INTO diseased_patient_audit (action, patient_id, action_timestamp)
        VALUES ('UPDATE', NEW.id, NOW());
    ELSIF TG_OP = 'DELETE' THEN
        INSERT INTO diseased_patient_audit (action, patient_id, action_timestamp)
        VALUES ('DELETE', OLD.id, NOW());
    END IF;
    RETURN NEW;
END;
$$
LANGUAGE plpgsql;
```

Comparison with nosql model

PROS

Document based

Flexible

Auto sharding

High performance

Horizontal scalability

```
[
    {
        "ID": 1,
        "First Name": "John",
        "Last Name": "Doe",
        "Gender": "M",
        "Location ID": 1,
        "Race": "WHITE"
    },
    {
        "ID": 2,
        "First Name": "Jane",
        "Last Name": "Smith",
        "Gender": "F",
        "Location ID": 2,
        "Race": "BLACK"
    }
]
```

CONS

Complex querying

Lack of transactions

Data modeling complexity

Data Retrieval and Analysis

Data retrieval

We used **views** and **functions** for easy retrieval of data through set of queries that's being frequently used.

```
CREATE OR REPLACE FUNCTION public.get_diseased_patient_count()
RETURNS INTEGER AS
$$
DECLARE
    patient_count INTEGER;
BEGIN
    SELECT COUNT(*) INTO patient_count FROM diseased_patient;
    RETURN patient_count;
END;
$$
LANGUAGE plpgsql;
--Calling Function
Select * from  public.get_diseased_patient_count();
```

```
--Views
-- View in public schema
CREATE VIEW public.vw_diseased_patients AS
SELECT *
FROM diseased_patient;

-- Call the stored procedure in the public schema
select * from public.vw_diseased_patients;
```

General queries for analysis

Report on Medicines and Indications:

Query | Query History

```
425 --Report on Medicines and Indications:
426 --This report lists medicines along with the diseases they're indicated for and their effectiveness.
427 SELECT m.name AS medicine_name, d.name AS disease_name, i.indication_date, i.effectiveness_percent
428 FROM public.indication i
429 JOIN public.medicine m ON i.medicine_id = m.medicine_id
430 JOIN public.disease d ON i.disease_id = d.id;
431
432 --Report on Race Disease Propensity:
```

Data Output | Messages | Notifications

	medicine_name character varying (250)	disease_name character varying (100)	indication_date date	effectiveness_percent double precision
1	Tylenol	Common Cold	2024-01-20	90.5
2	Advil	Diabetes	2024-02-25	85.3
3	Zyrtec	Hypertension	2024-03-15	92.1
4	Claritin	Asthma	2024-04-10	88.7
5	Benadryl	Arthritis	2024-05-15	91.2
6	Aspirin	Lung Cancer	2024-06-05	88.9
7	Nexium	Alzheimer's Disease	2024-06-20	85.2
8	Prozac	Psoriasis	2024-07-05	90.1

Distribution of Patients by Race:

Query | Query History

```
450 --This query provides insights into the distribution of patients across different races.
451
452 SELECT r.race_description, COUNT(*) AS patient_count
453 FROM public.person p
454 JOIN public.race r ON p.race_cd = r.race_code
455 GROUP BY r.race_description;
456 --Effectiveness of Medicines by Disease Type:
```

Data Output | Messages | Notifications

	race_description character varying (100)	patient_count bigint
1	Hispanic or Latino	1
2	Other	1
3	Asian	2
4	Middle Eastern	1
5	Chinese	1
6	White	1
7	Black or African American	1
8	Latinx	1

Successfully run. Total query runtime: 196 msec. 8 rows affected.

Total rows: 8 of 8 | Query complete 00:00:00.196 | Ln 455, Col 29

Using CTE

Query | Query History

```
282 WITH DiseaseIntensityCTE AS (
283     SELECT
284         d.name AS disease_name,
285         AVG(d.intensity_level_qty) AS avg_intensity_level
286     FROM
287         FactDiseaseModel.dim_disease d
288     JOIN
289         FactDiseaseModel.fact_disease_patient fp ON d.disease_id1 = fp.disease_id
290     GROUP BY
291         d.name
```

Data Output | Messages | Notifications

	disease_name character varying (100)	avg_intensity_level numeric
1	Lung Cancer	5.0000000000000000
2	Cystic Fibrosis	4.0000000000000000
3	Hypertension	4.0000000000000000
4	Alzheimer's Disease	4.0000000000000000
5	Psoriasis	3.0000000000000000

Total rows: 5 of 5 | Query complete 00:00:00.126 | Ln 301, Col 7

We used **CTE** (common table expression) to temporarily store the result of frequently used data.

It works like a cache.

Tableau for visualization

Effectiveness of medicines by disease type

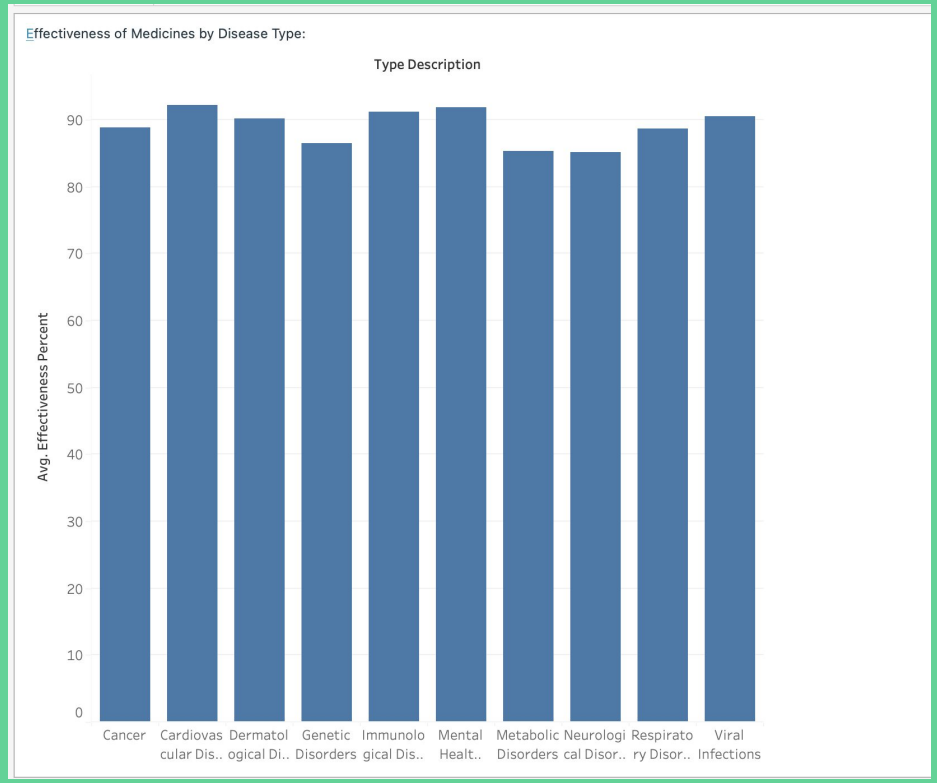


Tableau for visualization

Patients with ongoing disease

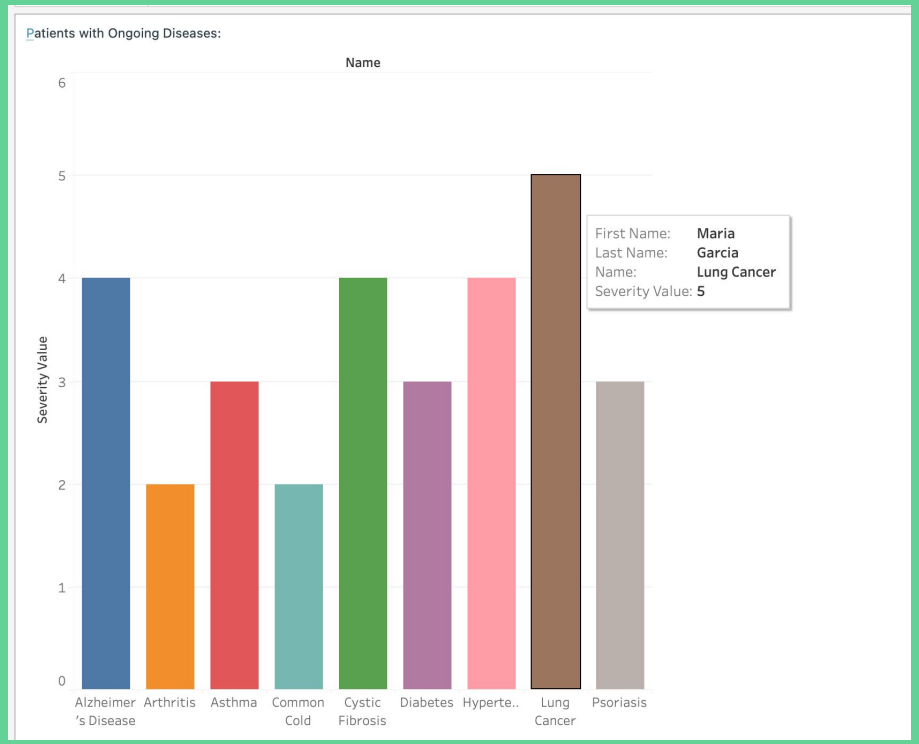
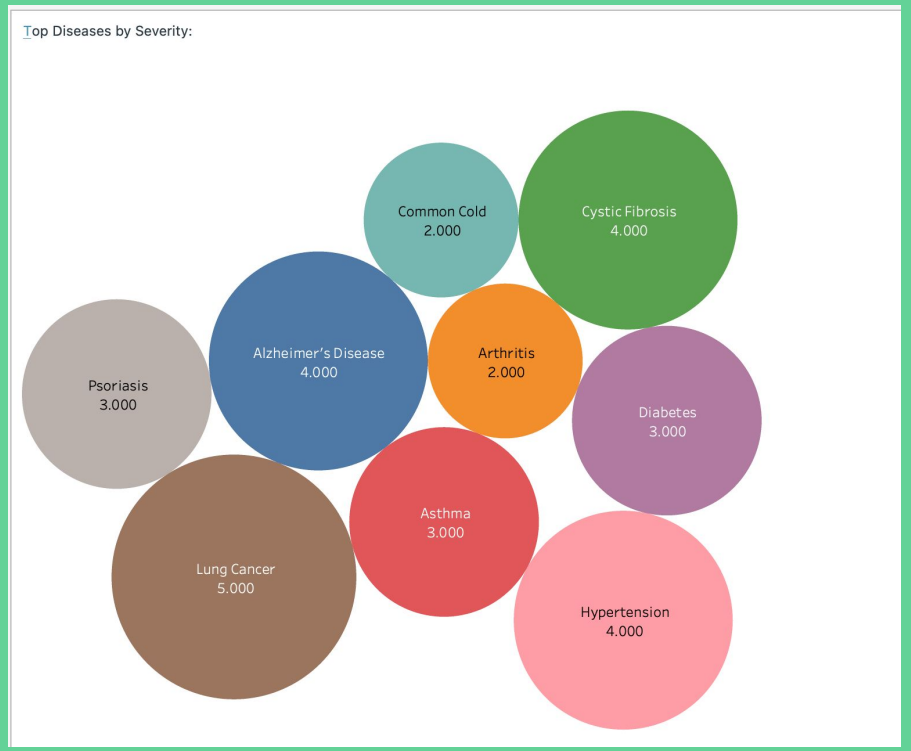
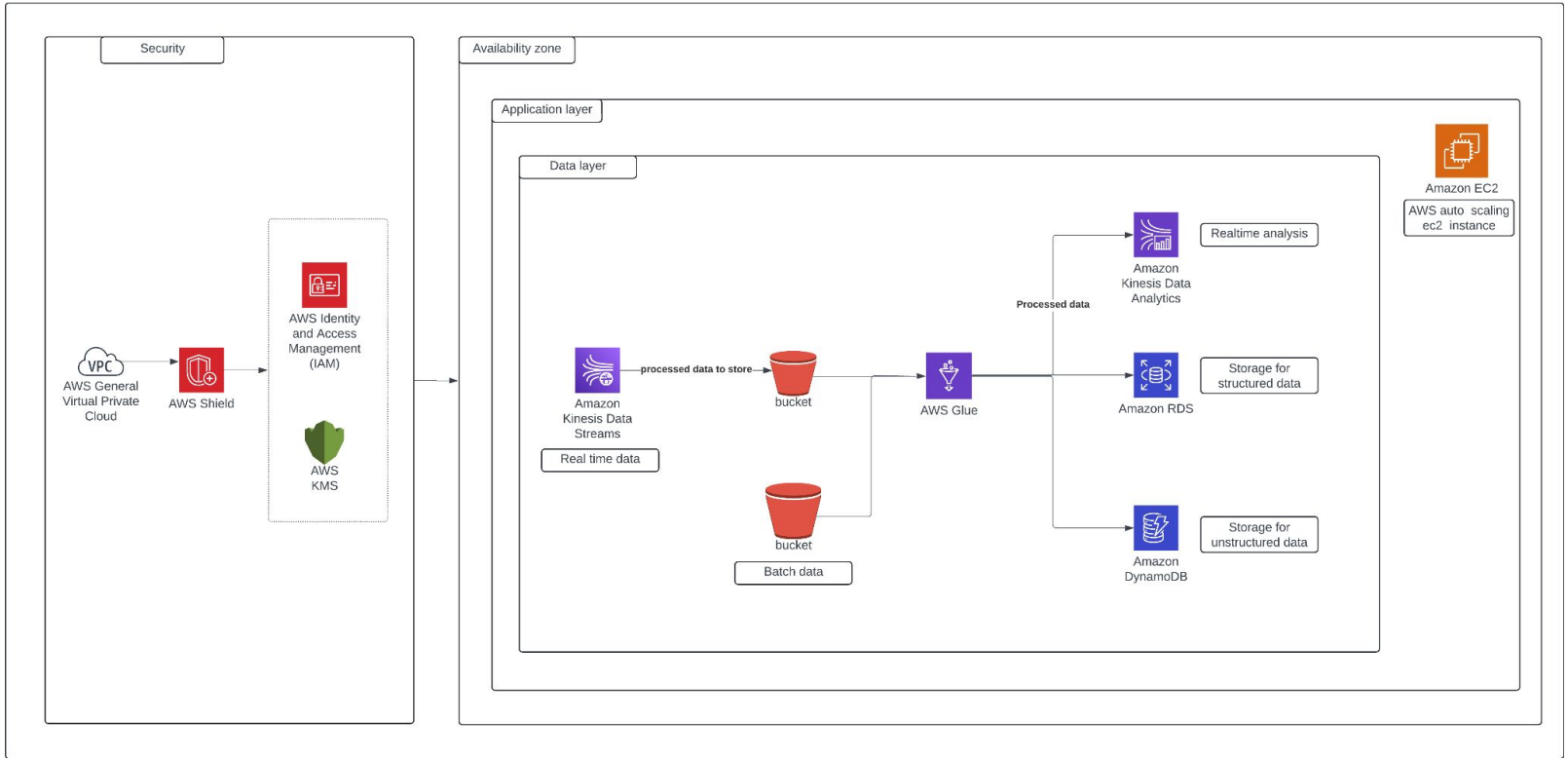


Tableau for visualization

Top disease by Severity



AWS architecture



Advantages of snowflake

1. Scalability
2. Concurrency
3. Separation of storage and compute
4. Managed service
5. Automatic optimization
6. Built in security (encryption, role based access)

Thank you!