# Solution –

* Upload datasets to AWS S3 (input-data folder).( Claims, disease,group,grpsubgrp,hospital,Patient\_records,Subgroup,subscriber)
* Use PySpark on Databricks to clean(removing null value from the unique id) the data (null handling, duplicate removal).
* Creating star schema for simplifies Querying for every use case. Also improved performance and optimized for analytics
* Upload the cleaned data to Redshift into domain-specific tables.
* Create analytical queries and store results in output tables under project\_output schema.
* Use BI tools for reporting, and Jira/GitHub for tracking and collaboration.

# Use Cases –

* Identify disease with maximum claims. – with the claims dataset use group by to find disease with maximum claims
* Find subscribers under age 30 with subgroups.- using subscribe dataset filter to find subscribers under age 30
* Track group with maximum subgroups. With the grpsubgrp we can use group by to count maximum subgroup
* Identify hospital with highest patient count.- joining patients\_records and hospital after that using group by with aggregation
* Finding which subgroup subscribe most number of times. With the subscriber dataset we can perform group by and we can short those on descending order
* Calculate total rejected claims. with the claims dataset we can find total claims
* Analyze city-wise claim distribution. joining on claims and patient\_records after that applying group by on city with aggregation on claim that are approved
* Compare subscriptions between government and private policies. With the grpsubgrp,subscribe and group join this this three dataset and performing groupby on group typ
* Compute average monthly premium. With the subscriber and subgroup join this two dataset and finding average monthly premium
* Identify most profitable group. With the group data set we an short the premium in descending order to find most profitable group
* List cancer patients below age 18. Filtering patient\_records that have cancer and they are below age 18
* Filter cashless patients with charges >= 50,000. Joining three data set claims subscriber and patient\_records after that we can filter out cashless patients that have charges greater than 50000
* List females 40+ who had knee surgery in last year. With the patient\_record we can filter out the females patients who are 40+ and those have knee surgery

1. Database Design - List down all possible db(Redshift) tables here

## Tables Metadata Info with Pk/FK relationship –

|  |  |  |  |
| --- | --- | --- | --- |
| Dataset Name | Columns | Primary Key(s) | Foreign Key(s) |
| claims | claim\_id, patient\_id, disease\_name,  SUB\_ID, Claim\_Or\_Rejected, claim\_type,  claim\_amount, claim\_date | claim\_id | patient\_id ->patient\_records(Patient\_id); SUB\_ID ->subscriber(sub \_id) |
| disease | SubGrpID, Disease\_ID, Disease\_name | Disease\_ID | SubGrpID ->subgroup(SubGrp\_id) |
| group | Country, premium\_written, zipcode, Grp\_Id,  Grp\_Name, Grp\_Type, city, year | Grp\_Id |  |
| grpsubgrp | SubGrp\_ID, Grp\_Id | SubGrp\_ID, Grp\_Id | SubGrp\_ID -> subgroup(SubGrp\_id); Grp\_Id -> group(Grp\_Id) |
| hospital | Hospital\_id, Hospital\_name, city, state, country | Hospital\_id |  |
| patient\_records | Patient\_id, Patient\_name, patient\_gender,  patient\_birth\_date, patient\_phone, disease\_name,  city, hospital\_id | Patient\_id | hospital\_id ->hospital(Hospital\_id) |
| subgroup | SubGrp\_id, SubGrp\_Name, Monthly\_Premium | SubGrp\_id |  |
| subscriber | sub \_id, first\_name, last\_name, Street, Birth\_date,  Gender, Phone, Country, City, Zip Code,  Subgrp\_id, Elig\_ind, eff\_date, term\_date | sub \_id | Subgrp\_id ->subgroup(SubGrp\_id) |

## ER diagram - *Optional*

# Technologies and Platforms to be used in this solution –

* 1. **Cloud Platform:** AWS (S3, Redshift)
  2. **Processing Framework:** PySpark
  3. **Data Engineering:** Databricks
  4. **Version Control & Task Management:** GitHub, Jira
  5. **Storage & Query Engine:** AWS Redshift
  6. **Optional BI Tools:** Amazon QuickSight, Tableau

## 