**1. Solution — step by step**

**Data Ingestion**

* Upload raw CSV/JSON/Parquet to **S3** under /input-data/<dataset>/.
* Keep separate folders: **patients**, **subscribers**, **claims**, **group\_subgroup**, **hospitals**, **cities**, **policies**.
* Log file name, size, checksum, and ingest time.

**Data Cleaning with PySpark**

* Count nulls per column and replace string nulls with **"NA"**.
* Drop duplicates using business keys (e.g., claim\_number + service\_date).
* Standardize types, trim whitespace, fix codes (disease/procedure).

**Data Storage**

* Write cleaned data to **Redshift** (staging → dw schema).
* Enforce PK/FK via ETL checks; keep a simple DQ summary table.

**Transformation / Business Queries**

* Use **PySpark/SQL** to build results for each use case.
* Save all outputs in Redshift under schema **Project-Output** (one table per use case).

**Result Consumption**

* Run queries directly in **Redshift Query Editor** or **Databricks SQL**.
* Optional: basic visual snapshots for presentation.

**Agile Workflow**

* **Week 1:** Requirements & Solution Design docs.
* **Week 2:** Implementation & Testing.
* Track user stories/tasks in **Jira**.

**2. Use Cases (tables in Project-Output)**

* **Disease with max claims** — max\_claims\_by\_disease
* **Subscribers <30 subscribed to subgroups** — subscribers\_under30\_with\_subgroups
* **Group with most subgroups** — group\_with\_max\_subgroups
* **Hospital with most patients** — top\_hospitals\_by\_patients
* **Most‑subscribed subgroups** — top\_subgroups\_by\_subscriptions
* **Total rejected claims** — claims\_rejected\_summary
* **City with most claims** — top\_cities\_by\_claims
* **Govt vs Private preference** — policy\_group\_popularity
* **Average monthly premium** — avg\_monthly\_premium
* **Most profitable group** — group\_profitability
* **Patients <18 with cancer** — patients\_under18\_cancer
* **Cashless charges ≥ 50,000** — cashless\_high\_charges
* **Female >40 knee surgery (past year)** — female\_over40\_knee\_past\_year

**3. Database Design (Redshift)**

**a) Tables (minimal metadata)**

* **patients** *(PK: patient\_id)* — patient\_token, dob, gender, city\_id.
* **subscribers** *(PK: subscriber\_id, FK: patient\_id, group\_id, subgroup\_id)* — start\_date, end\_date, status.
* **claims** *(PK: claim\_id, FK: patient\_id, hospital\_id, city\_id, disease\_code, procedure\_code)* — service\_date, status, is\_cashless, total\_charges, paid\_amount.
* **groups** *(PK: group\_id)* — group\_name, group\_type *(Government|Private)*.
* **subgroups** *(PK: subgroup\_id, FK: group\_id)* — subgroup\_name, active\_flag.
* **group\_subgroup** *(PK: group\_id + subgroup\_id + effective\_start)* — effective\_start, effective\_end, active\_flag.
* **hospitals** *(PK: hospital\_id, FK: city\_id)* — hospital\_name, npi\_code.
* **cities** *(PK: city\_id)* — city\_name, state, country.
* **policies** *(PK: policy\_id, FK: group\_id)* — policy\_name, premium\_frequency, base\_premium.
* **subscriber\_policy** *(PK: subscriber\_id + policy\_id + effective\_start)* — monthly\_premium\_derived.
* **fact\_premium** *(PK: premium\_id, FK: subscriber\_id, policy\_id)* — billing\_month, billed\_amount, paid\_amount.
* **diseases** *(PK: disease\_code)* — disease\_name, category.
* **procedures** *(PK: procedure\_code)* — procedure\_name, category.

**Ops / DQ**

* **run\_log** — run\_id, job\_name, start\_ts, end\_ts, status.
* **dq\_run\_metrics** — dataset, metric\_name, metric\_value, status.
* **file\_ingest\_log** — source, path, checksum, size\_bytes, ingest\_ts.

**b) ER diagram — Optional (summary)**A diagram of a company

AI-generated content may be incorrect.

**4. Technologies & Platforms**

* **AWS S3** → raw data storage.
* **Amazon Redshift** → analytics warehouse + Project-Output tables.
* **AWS EMR Studio** or **Databricks (community for testing)** → run PySpark jobs.
* **PySpark** → data cleansing & transformations.
* **Jira** → Agile sprint tracking.
* **GitHub (single repo)** → version control, code + docs.
* *(Security underpinning:* ***IAM + KMS****, secrets via Databricks scopes or AWS Secrets Manager.)*