

Covid-19: Cases Data Project(Batch processing)

What is Batch Processing?

Data which is collected over time and then fed into a system where “meaning” is extracted and further it is stored(warehouses) or it is used to process.









For e.g., *Daily processing of bills in a retail shop.*

Context:

- A new coronavirus designated 2019-nCoV was first identified in Wuhan, the capital of China's Hubei province.
- People developed pneumonia without a clear cause and for which existing vaccines or treatments were not effective.
- The virus has shown evidence of human-to-human transmission.
- Transmission rate (rate of infection) appeared to escalate in mid-January 2020
- As of 15th May 2023, approximately 69,27,378 deaths have been confirmed.

Dataset:

- There are 2,31,745 rows present in the dataset.
- There are 6 columns(Date, Country, State, Confirmed, Recovered, Deaths) in the dataset.
- Data present in the dataset are from **2020 To 2022**.
- We will be considering “time-series-19-covid-combined” table.

 countries-aggregated
 key-countries-pivoted
 reference
 time-series-19-covid-combined
 us_confirmed
 us_deaths
 us_simplified
 worldwide-aggregate

Tech Stack:

- a. Hadoop with two main components(HDFS & YARN) *
- b. Python*
- c. Anaconda(IDE)*
- d. Spark/PySpark*
- e. SQL(PostgreSQL/MySQL)*
- f. NoSQL(Cassandra/HBase)
- g. Dashboarding(Tableau, Power-BI, Grafana*, Kibana)

High-Level Diagram:



Covid – 19 : Cases Data Project



Milestones:



Milestone – 1

- Clone/Download file from Git repo & save it in your local.
- [GitHub - datasets/covid-19: Novel Coronavirus 2019 time series data on cases](https://github.com/datasets/covid-19:NovelCoronavirus2019time-series-data-on-cases)



Milestone – 2

- Setup Hadoop in local.
- Here we need to focus on only two main components of Hadoop(HDFS & YARN)
- HDFS is distributed file system so now copy that file from local to HDFS.
 - `hdfs dfs mkdir /dir_name`
 - `hdfs dfs -ls /`
 - `hdfs dfs -copyFromLocal "path" /dir_name_in_Hadoop`
 - `hdfs dfs -ls /dir_name`



Milestone – 3

- Setup spark in your local.
 - We need to install Spark in our local and further we need to instantiate.
 - To initiate we need we can use "spark" in the CMD, or we can use it in Jupyter Notebook.
- Write a spark application which reads file from HDFS and creates a DataFrame.
 - a. With the help of Spark we could pull the data from HDFS(Hadoop Distributed File System) and create a DataFrame.
- Just looking a quick view, we can understand how many columns are there and we can plan what type of further transformation we could do.



Milestone – 4

- Following basic transformation on the data:
 1. Ingesting only the required columns.
 2. Dropping duplicate values
 3. Null handling transformation for all the columns.
 - String: NA
 - INT: 0

Float: 0.0

Date/Timestamp: 1800-01-01

4. Column renaming

5. Type casting according to the required data type.

Milestone – 5

- Aggregate data in spark application.

1. *How many total cases so far?*

2. *How many recovered cases so far?*

3. *How many deaths so far?*

Milestone – 5

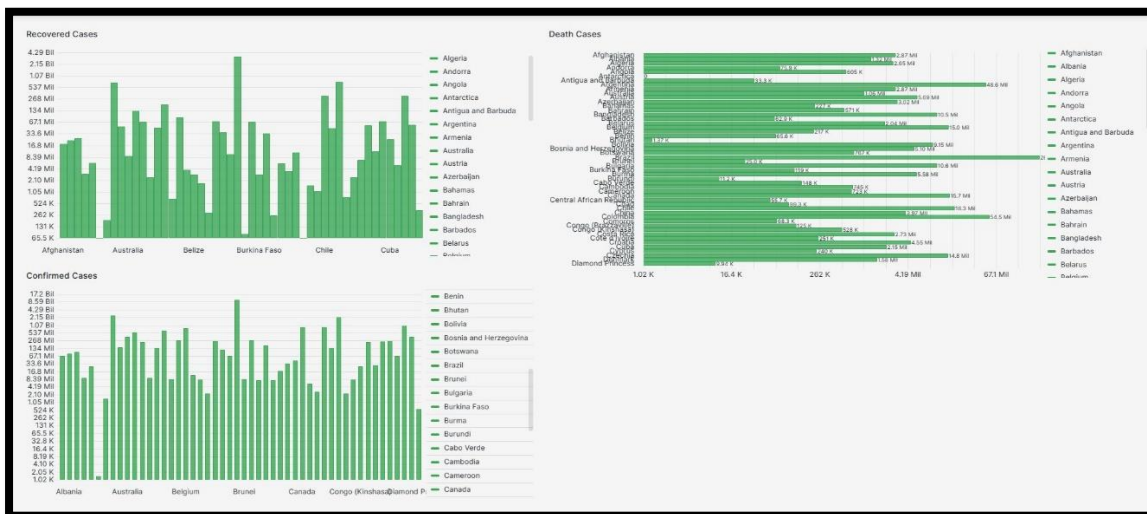
- Write the aggregated data in output table using transactional database like MySQL/Postgres or in NoSQL database.

Milestone – 6

- Use dashboard tool: Grafana

- Plugins: World-map/ Geo-Map

- Dashboard Link: http://localhost:3000/goto/xd4_gDQ4R?orgId=1



Ref Link:

1. [2 Beginner Level Projects For Aspiring Data Engineers ⌚ Covid-19 Data | Real Time Data Processing 📺 - YouTube](#)
2. [PySpark GroupBy Agg | Working of Aggregate with GroupBy in PySpark \(educba.com\)](#)
3. [PySpark Groupby Agg \(aggregate\) - Explained - Spark By {Examples} \(sparkbyexamples.com\)](#)
4. [PySpark Aggregate Functions with Examples - Spark By {Examples} \(sparkbyexamples.com\)](#)
5. [PySpark Aggregate Functions: A Comprehensive Guide | by Ahmed Uz Zaman | Medium](#)
6. [PySpark alias\(\) Column & DataFrame Examples - Spark By {Examples} \(sparkbyexamples.com\)](#)
7. [Pyspark: GroupBy and Aggregate Functions | M Hendra Herviawan \(hendra-herviawan.github.io\)](#)
8. [Load DataFrames To PostgreSQL 10x Faster | Towards Data Science](#)
9. [python - How to write DataFrame to postgres table - Stack Overflow](#)
10. [Pyspark write to postgres - spark write to postgres - Projectpro](#)
11. [postgresql - unable to connect to server for Postgres - Stack Overflow](#)
12. [python - DataFrame constructor not properly called - Stack Overflow](#)
13. [postgresql - OperationalError: \(psycopg2.OperationalError\) could not translate host name "143@postgres" to address: Unknown server error - Stack Overflow](#)