How to Analyze Big Data for Your Business Organization

By: Rauzan Fikri

Hi ....

- OmahKu CTO
- Senior Manager of Testing Center of Excellence (TCoE)
- Engineering Manager at VersaFleet
- Presales Manager at Verint Customer Engagement

Linkedin: https://www.linkedin.com/in/rfi/

Email: rauzan.fikri@omahku-id.com

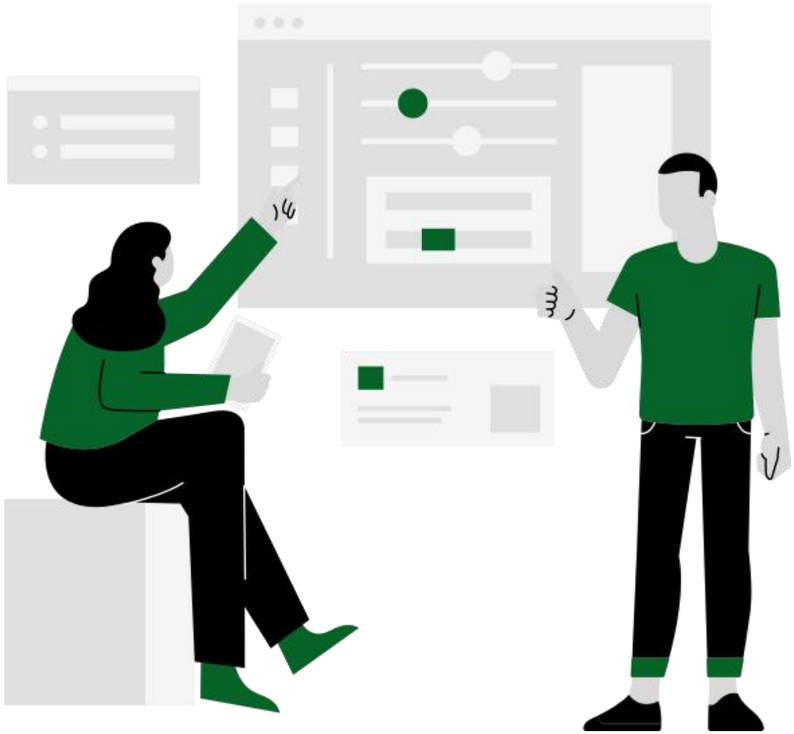
Rauzan Fikri

CTO OmahKu Indonesia

Table Of Content

Tentang Big Data

Data yang selalu bertambah dan tidak dapat diproses dan disimpan dalam satu mesin disebut sebagai Big Data.

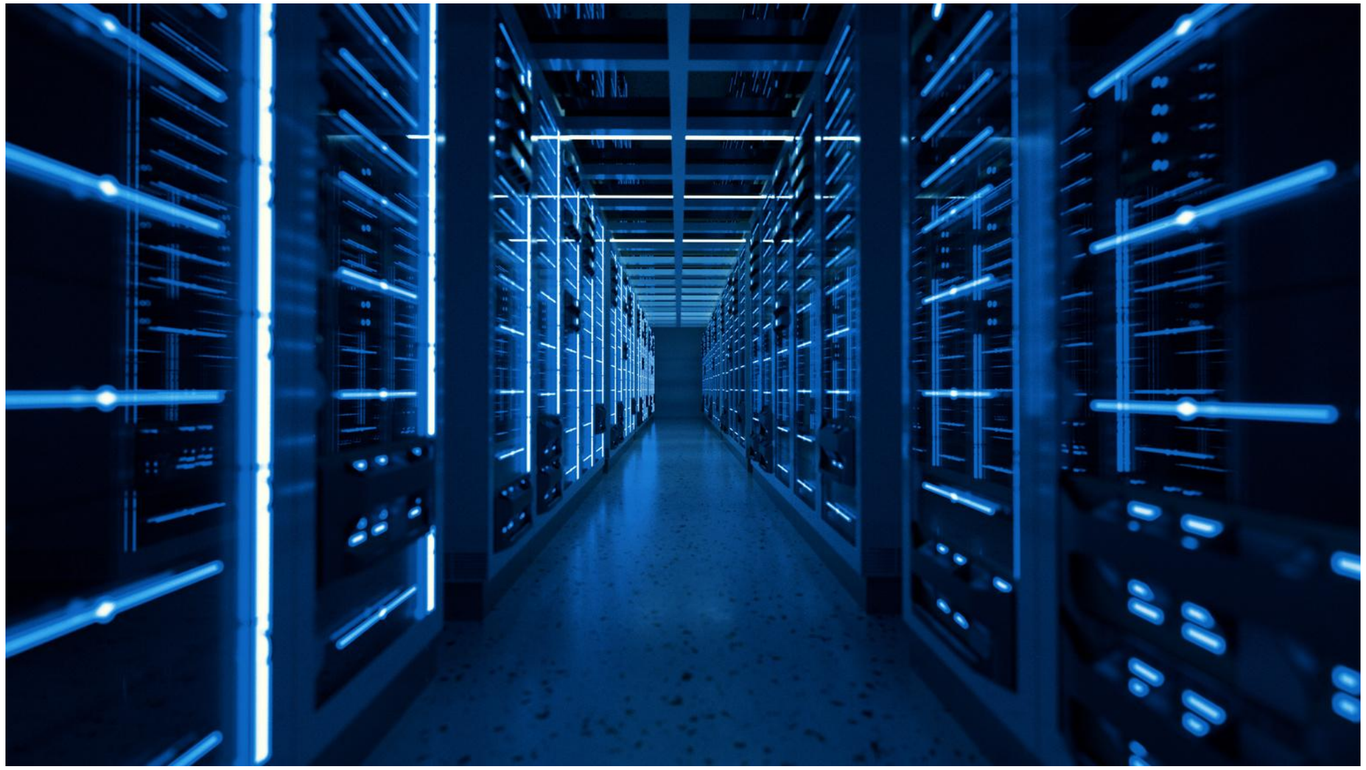


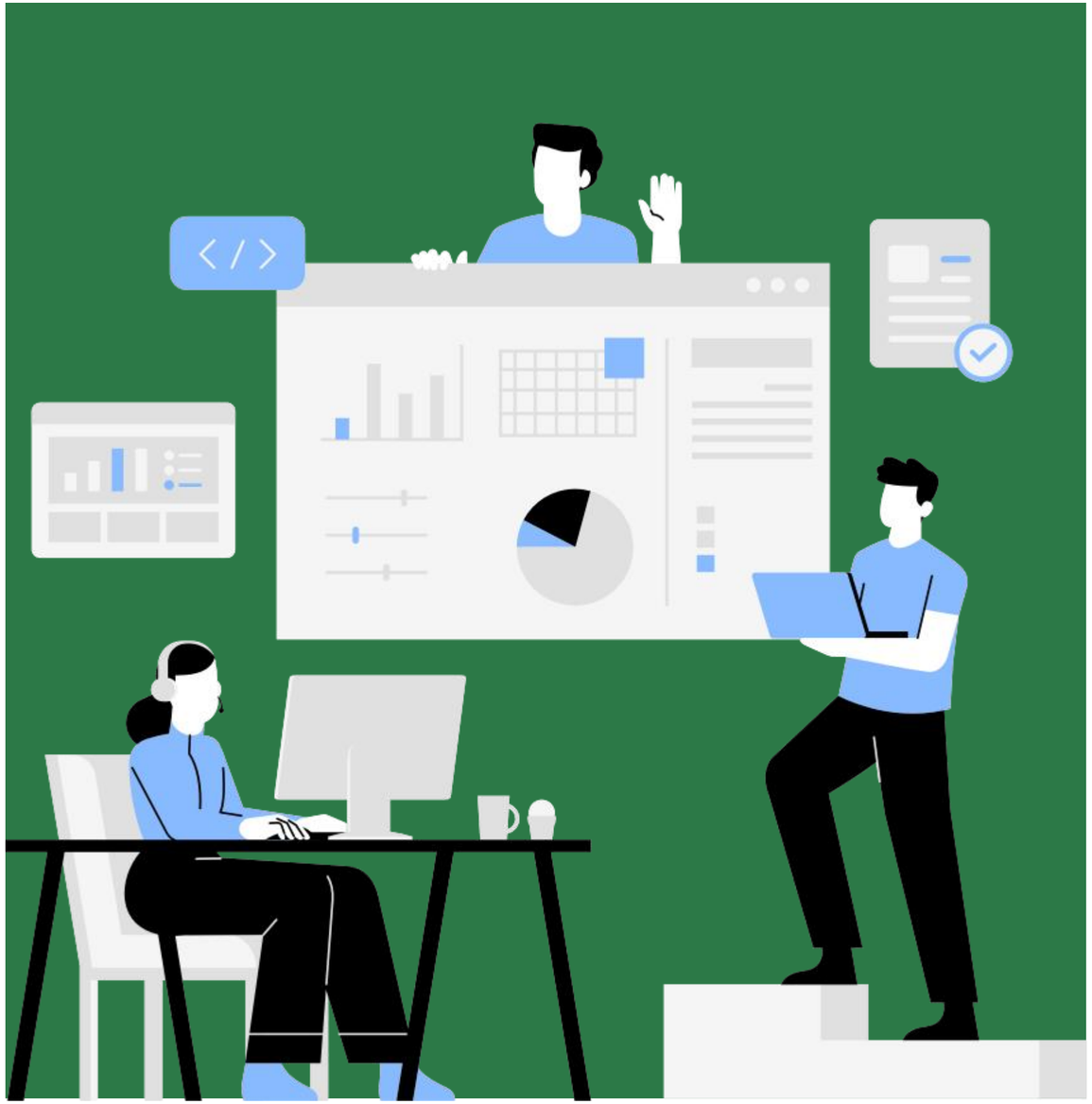
Big Data Merujuk pada Apa?

- Apakah datanya terlalu besar?
- Data yang penting?
- Data yang tidak dapat diproses oleh perangkat lunak komputasi biasa?
- Data yang memberi tahu Anda semua?

Big Data Cluster

- Mesin-mesin terhubung satu sama lain melalui jaringan untuk bertindak sebagai satu sistem.
- Mesin tidak lain adalah perangkat keras (CPU + RAM)
- Perangkat keras ini ditumpuk bersama di atas Rak.
- Rak ini kemudian dipasang di lokasi fisik yang disebut sebagai Pusat Data atau Data Center.





Big Data Pipeline

Big Data Ingestion	Sqoop/Flume
Data Validation, Cleanup & Processing	Spark
Data Analysis	Hive
Data Visualization	Tableau, Power BI

Ekosistem Big Data

Data Source

Ingestion

kafka

APACHE nifi

APACHE SQOOP

aws Glue

Processing

kafka

APACHE Spark

Flink

Cluster Manager

kubernetes

hadoop YARN

Processed Storage

cassandra

mongoDB.

RDBMS

Distributed Storage

Amazon S3

Azure Blob Storage

Google Cloud Storage

Data Analysis

Apache Hive

HIVE

Data Visualization

+ a b | e a u

Power BI

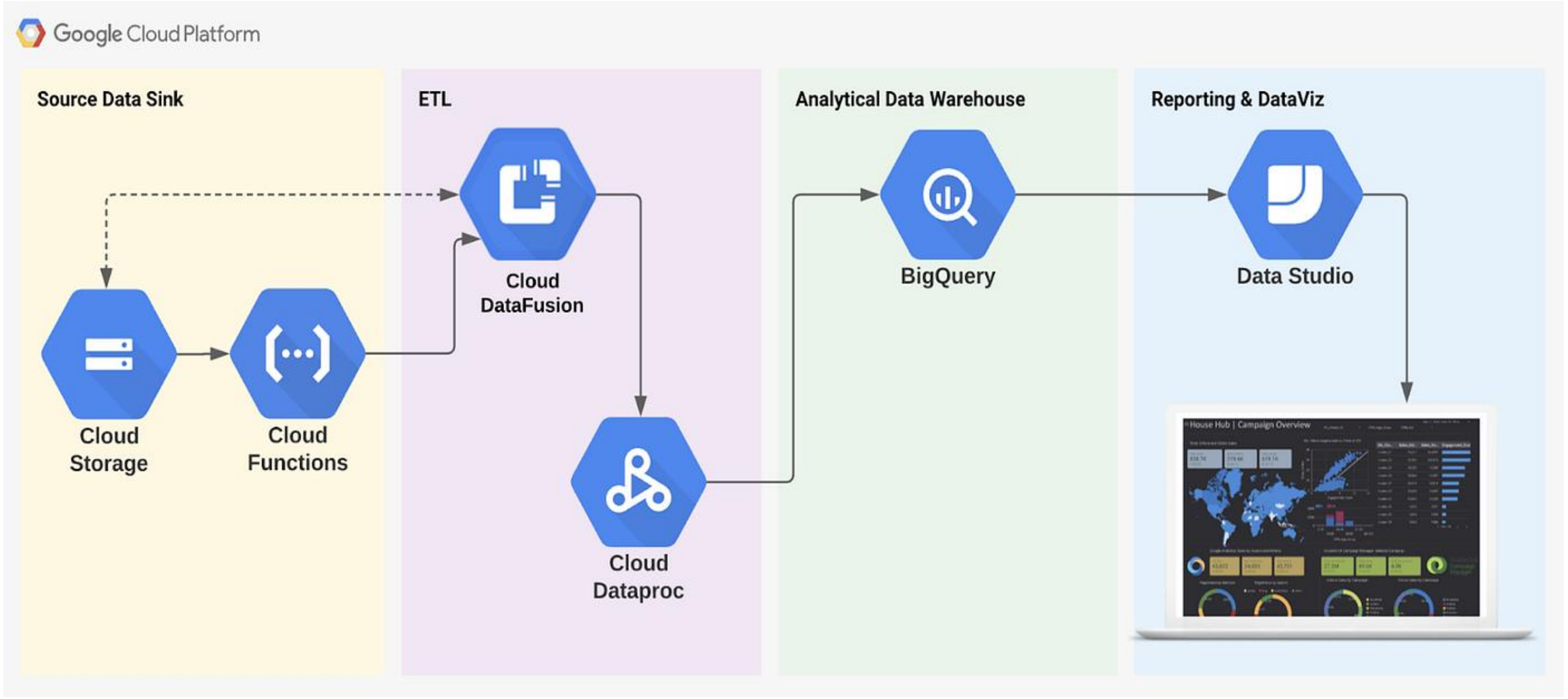
Metabase

Machine Learning

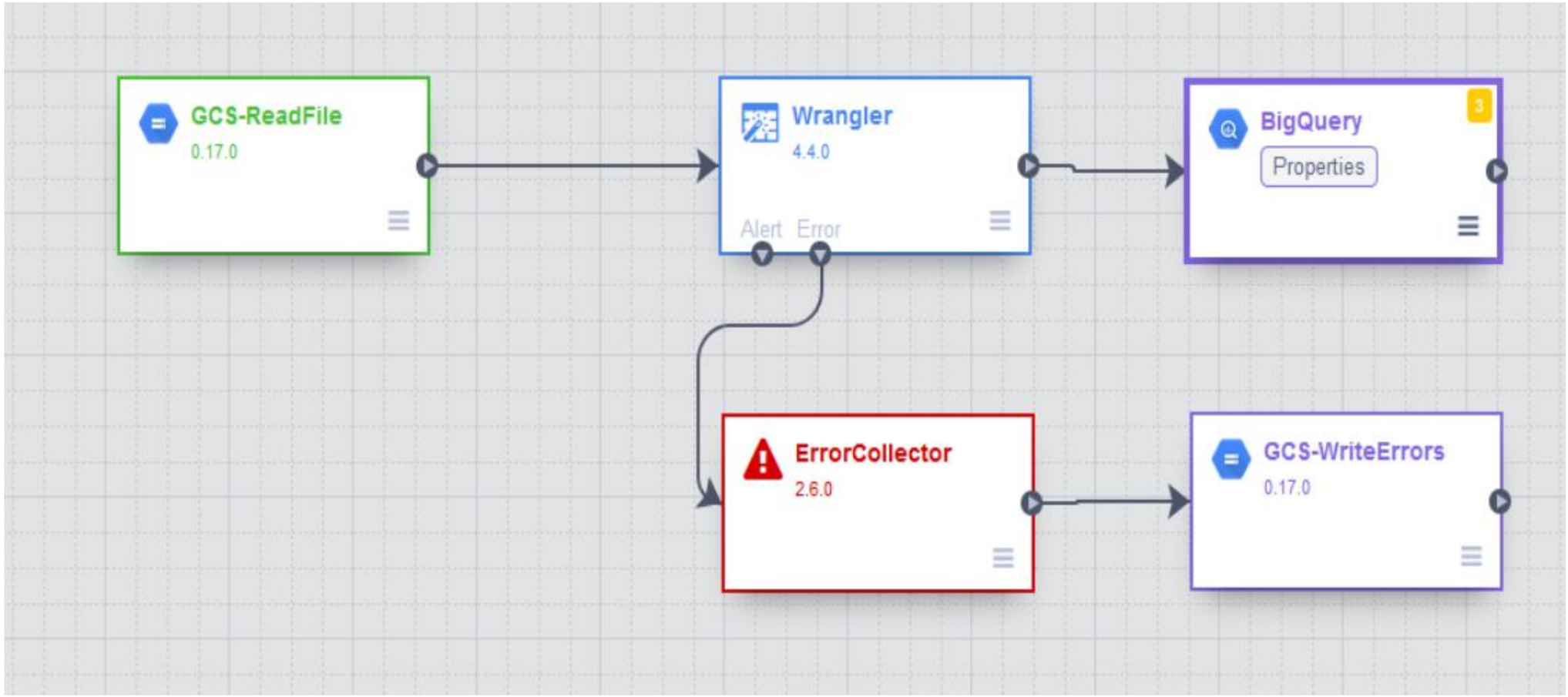
Spark MLlib

Amazon SageMaker

Study Case


Google Cloud Platform
Source Data Sink
ETL
Analytical Data Warehouse
Reporting & DataViz
Cloud Storage
Cloud Functions
Cloud DataFusion
Cloud Dataproc
BigQuery
Data Studio

Study Case


GCS-ReadFile
0.17.0
Wrangler
4.4.0
Alert
Error
BigQuery
Properties
ErrorCollector
2.6.0
GCS-WriteErrors
0.17.0

Comparing

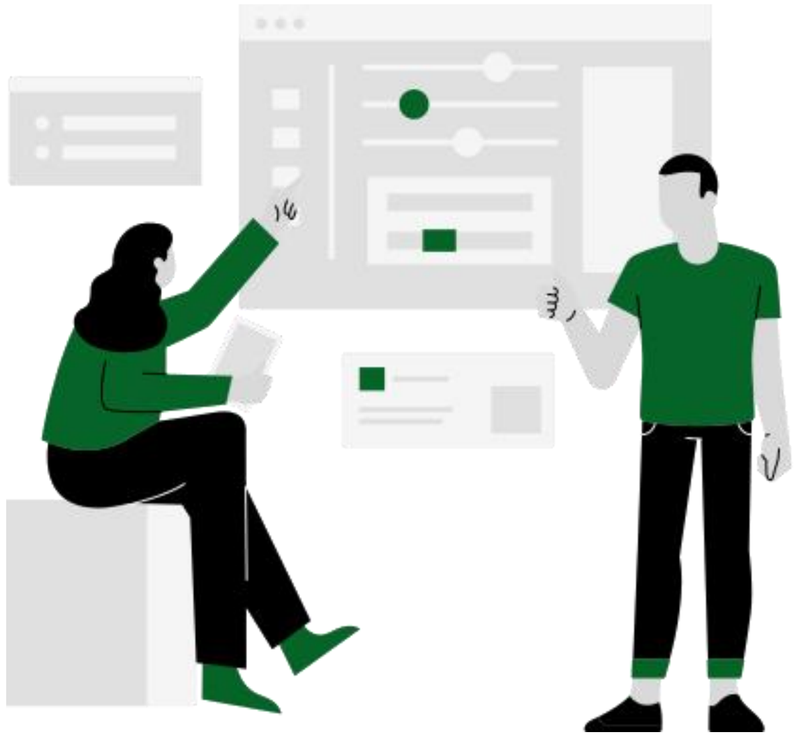
Fitur	Data Fusion	Dataproc	Cloud Functions	BigQuery	Data Studio
Fokus Utama	Integrasi & pipeline data	Pemrosesan skala besar	Pemrosesan event-driven	Analitik data di warehouse	Visualisasi & pelaporan
Jenis Pemrosesan	ETL/ELT	Batch/Streaming	Event-driven	Batch/Real-time	Visualisasi
Use Case Utama	Integrasi multi-sumber	Pemrosesan data lake besar	Transformasi ringan	Query SQL untuk analitik	Dashboard interaktif
Antarmuka	Visual (drag-and-drop)	Command-line/API	Coding (event-driven)	SQL-first	Visual/UI-first
Kelebihan Utama	Mudah untuk ETL/ELT	Fleksibel untuk Hadoop/Spark	Sangat ringan	Analisis data cepat	Mudah untuk pengguna awam
Keterbatasan Utama	Tidak ideal untuk analitik	Perlu pengetahuan Hadoop/Spark	Tidak untuk data besar	Terbatas pada data di warehouse	Hanya untuk visualisasi

Tentang Spark

Spark dirancang untuk menyederhanakan pemrosesan data yang besar dan kompleks dengan menyediakan antarmuka yang mudah digunakan, kinerja tinggi, dan dukungan untuk berbagai jenis aplikasi pemrosesan data.

Spark vs Hadoop MapReduce

1. In memory Computation
2. 100X time faster
3. Lazy Evaluation
4. Support for multiple language Python , Java & Scala



Demo Time !

- Installation Spark
- Spark Command
- Spark Shell on Local Computer

How to Install Docker on Windows

- Download & Install Docker Dekstop
  - Check System Requirements
    - Supported OS: Windows 10 (64-bit) Pro, Enterprise, Education, or Windows 11 (Home is supported with WSL)
    - Hardware: 64-bit processor, 4 GB RAM minimum.
    - Virtualization: Ensure hardware virtualization is enabled in the BIOS.

- Download Docker Desktop
  - Go to the Docker Desktop download page (https://docs.docker.com/desktop/setup/install/windows-install/).
  - Click Download for Windows.

- Run the downloaded installer (Docker Desktop Installer.exe).
  - Follow the installation wizard:
    - Accept the license agreement.
    - Choose WSL 2 or Hyper-V backend (default is WSL 2 for Windows 10 Home and Windows 11).
    - Click Install.
    - Once installation completes, restart your computer if prompted.

How to Install Docker on Windows

- Enable WSL 2 Backend (For Windows Home)
  - Enable WSL
    - Open PowerShell as Administrator and run (wsl --install)
    - If already installed, ensure WSL 2 is the default version: wsl --set-default-version 2
  - Install a Linux distribution (e.g., Ubuntu) from the Microsoft Store.

- Start Docker Desktop
  - Verify Docker is running by opening a terminal (PowerShell or Command Prompt) and running:
    - docker --version
  - Verify docker compose installation by running:
    - docker-compose --version

How to Install Spark on Windows

- Copy my docker-compose into your spark project folder.
- Download MobaXTream or use powershell and edit the docker-compose file.
- Change this part with you folder directory.

```
volumes:
  - /Users/rfikri/Projects/Spark/data:/data # Shared directory
```

- Run: **docker-compose up -d** inside you spark project folder to bring up the container and download the images.
- Run **docker ps** command to view the running containers.
- Copy the sample data into your local directory
- Verify the file with command **docker exec -it spark-master ls /data**
- Run this command to stop the container: **docker-compose down**
- Verify the master and worker Web UI
  - spark master: localhost:8080
  - spark worker: localhost:8081

Run Spark on Windows

- Install the excel depedencies and run on spark session: spark-shell --packages com.crealytics:spark-excel_2.12:3.3.4_0.20.4
- Verify Installation

```
import org.apache.spark.sql._
println("Library loaded successfully!")
```

- Load data from excel file sampleData with all sheet loaded

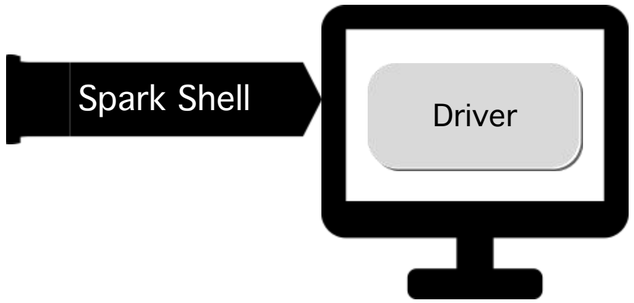
```
val filePath = "/data/sampleData.xlsx"
val sheetNames = Seq("Applications", "Logs")
val allSheets = sheetNames.map(sheetName => sheetName ->
spark.read.format("com.crealytics.spark.excel").option("header", "true").option("inferSchema",
"true").option("dataAddress", s"'$sheetName'!").load(filePath)).toMap

// Access data from "Applications"
allSheets("Applications").show()

filtering logs for QRIS application
val qrisLogs = allSheets("Logs").filter($"Application Name" === "QRIS")
 qrisLogs.show() //show the first 20 rows of the qrisLogs DataFrame
qrisLogs.show(50, false)  // Shows the first 50 rows without truncating the content
```

Thanks

Rauzan Fikri

Appendix

How Spark Works?



Step 1

```
val movieDatas =
spark.read.
option("multiLine",true)
.json("/Users/rfikri/Doc
uments/bigData/movies.
json")
```

Step 2

```
val filter1
= movieDatas
.filter("rating>3.0")
```

Step 3

```
val filter2
= filter1
.groupBy(“year”)
.count()
```

Step 4

```
filter2.show()
.filter("rating>3.0")
.groupBy(“year”)
.count()
.show()
```

name	rating	year
Titanic	4.0	2001
Sunshine	3.5	2004
3 Idiots	4.5	2004
Inception	4.0	2001
Wolf of wall street	3.5	2001

→

Titanic, Inception, Walf
| 4.0, 4.0, 3.5 | 2001

Sunshine, 3 Idiots
| 3.5, 4.5 | 2004

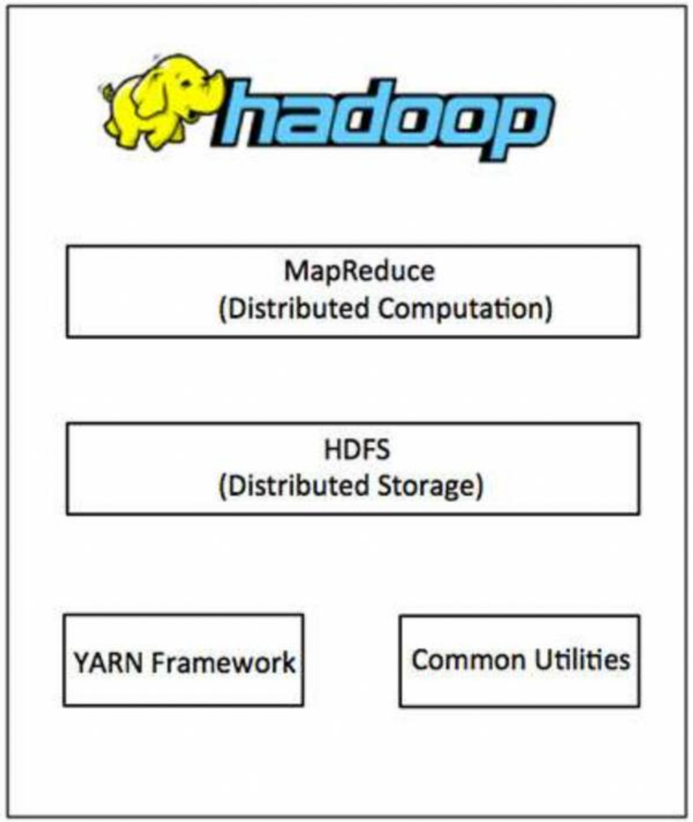
→

year	count
2004	2
2001	3

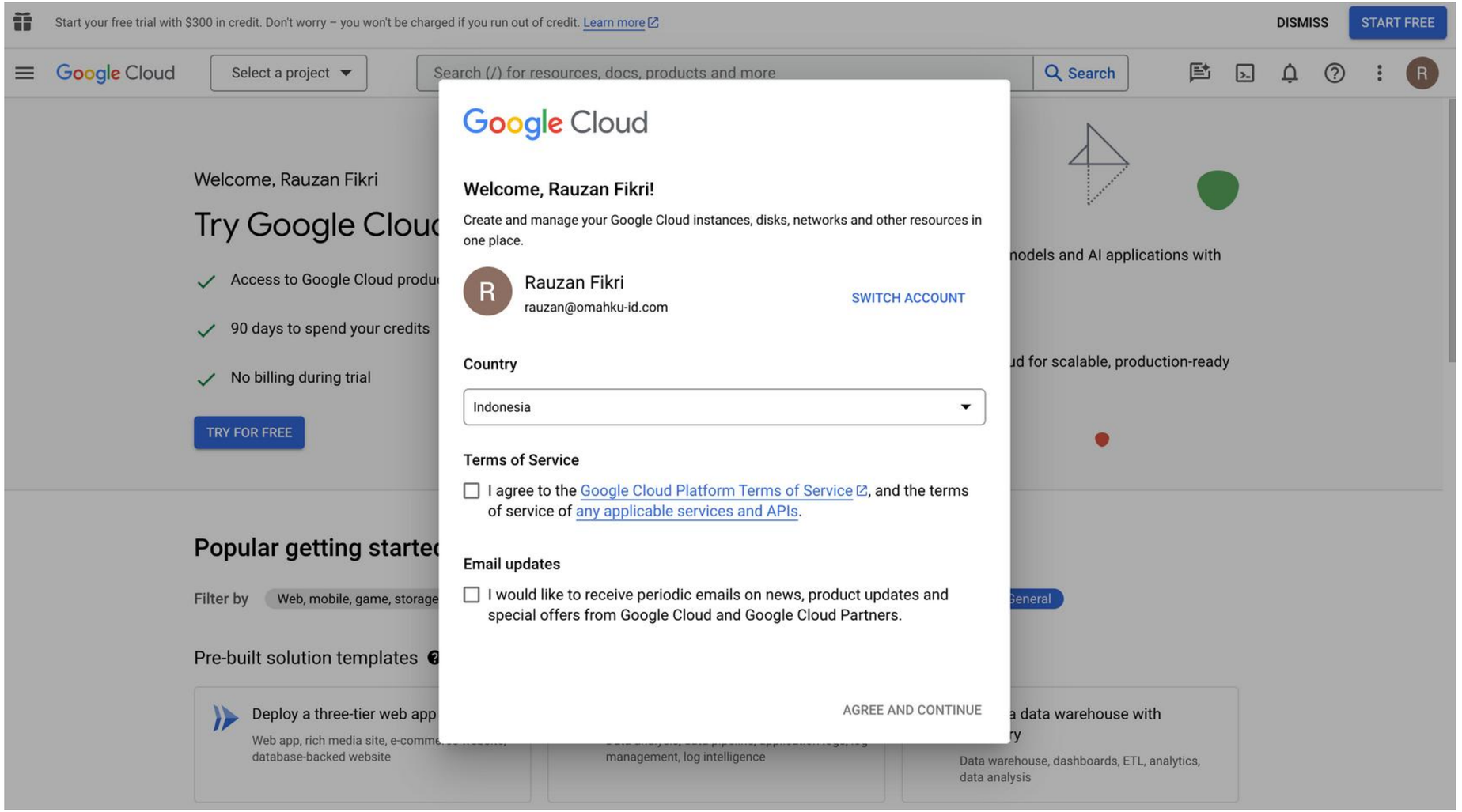
Tentang Hadoop Yarn

Hadoop YARN (Yet Another Resource Negotiator) adalah komponen utama dari ekosistem Hadoop yang bertanggung jawab atas manajemen sumber daya dan penjadwalan tugas pada platform Hadoop.

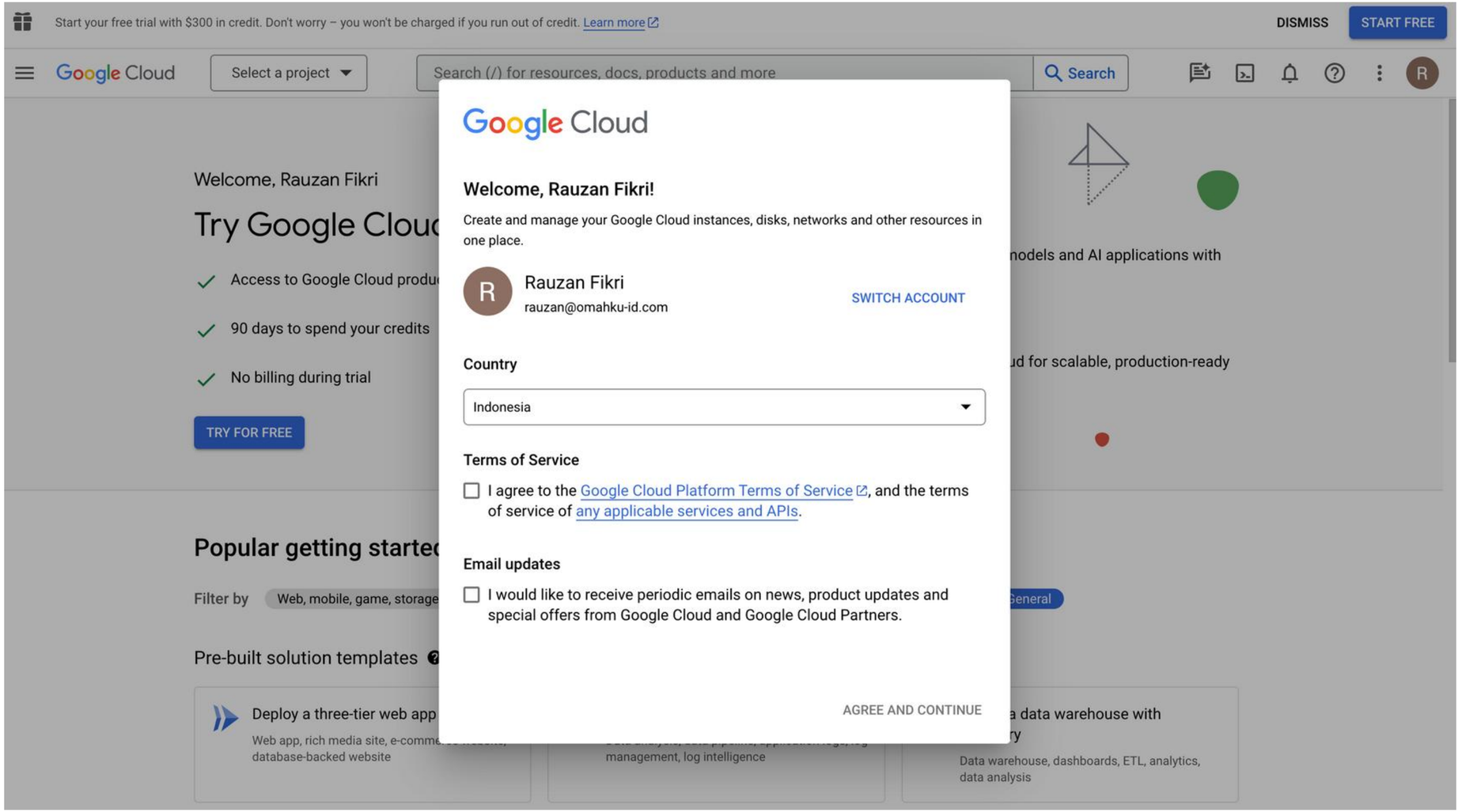
Secara singkat Yarn akan berperan penting dalam hal manajemen cluster atau node manager yang digunakan untuk melakukan pemprosesan data



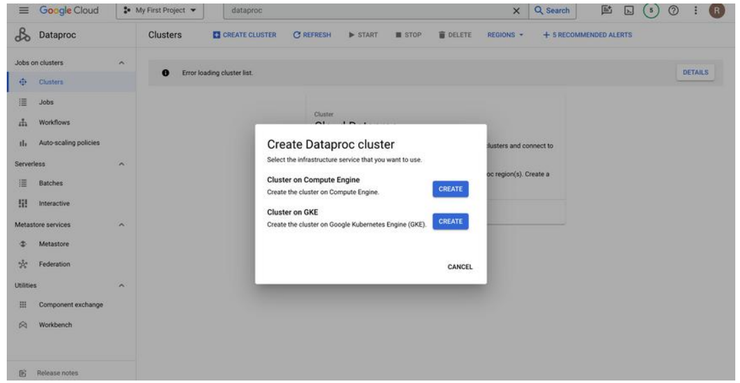
Register for Trial

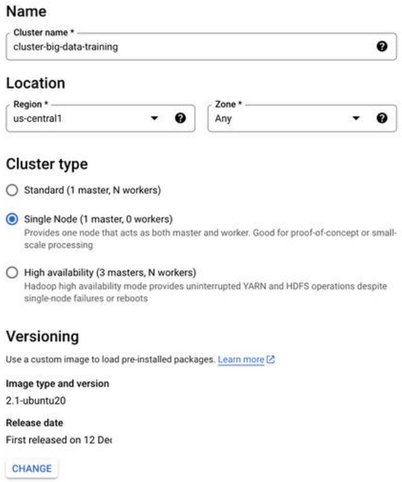


Register for Trial



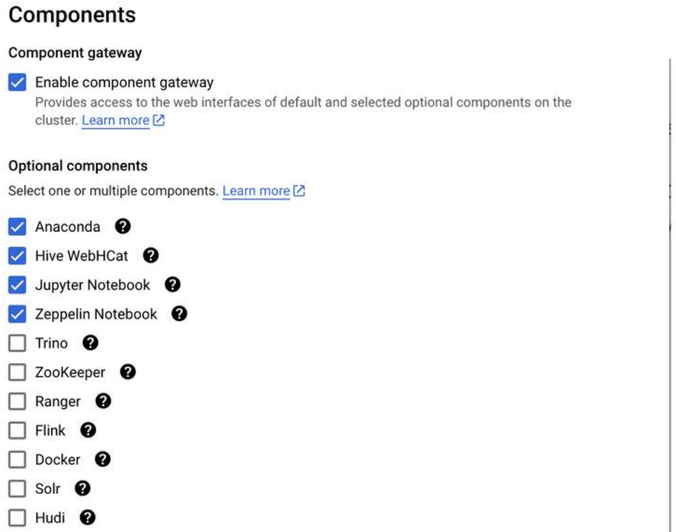
Create Dataproc Cluster

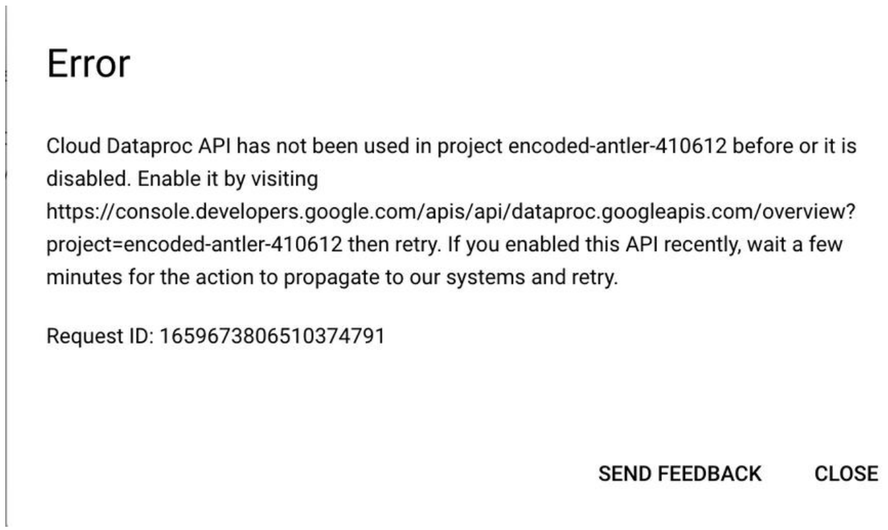




Region	us-central1
Zone	us-central1-c
Image version	1.5.90-ubuntu18
Auto-scaling	Off
Dataproc Metastore	None
Scheduled deletion	Off
Confidential Computing enabled?	Disabled
Master node	Single Node (1 master, 0 workers)
Machine type	n2-standard-4
Number of GPUs	0
Primary disk type	pd-standard
Primary disk size	500GB
Local SSDs	0
Secure Boot	Disabled
VTPM	Disabled
Integrity Monitoring	Disabled
Cloud Storage staging bucket	dataproc-staging-us-central1-635290206177-pxebjxh1
Network	default
Network tags	None
Internal IP only	No
Created	Unknown
Optional components	JUPYTER HIVE_WEBHCAT ZEPPELIN ANACONDA

Create Dataproc Cluster

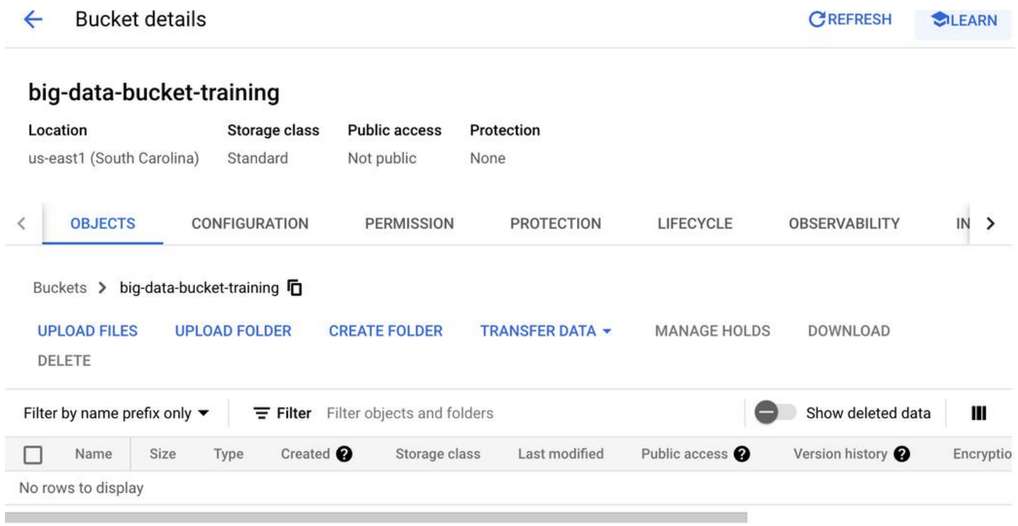




Failed to validate permissions required for default service account: '635290206177-compute@developer.gserviceaccount.com'. Cluster creation could still be successful if required permissions have been granted to the respective service accounts as mentioned in the document https://cloud.google.com/dataproc/docs/concepts/configuring-clusters/service-accounts#dataproc_service_accounts_2. This could be due to Cloud Resource Manager API hasn't been enabled in your project '635290206177' before or it is disabled. Enable it by visiting 'https://console.developers.google.com/apis/api/cloudresourcemanager.googleapis.com/overview?project=635290206177'.

MORE

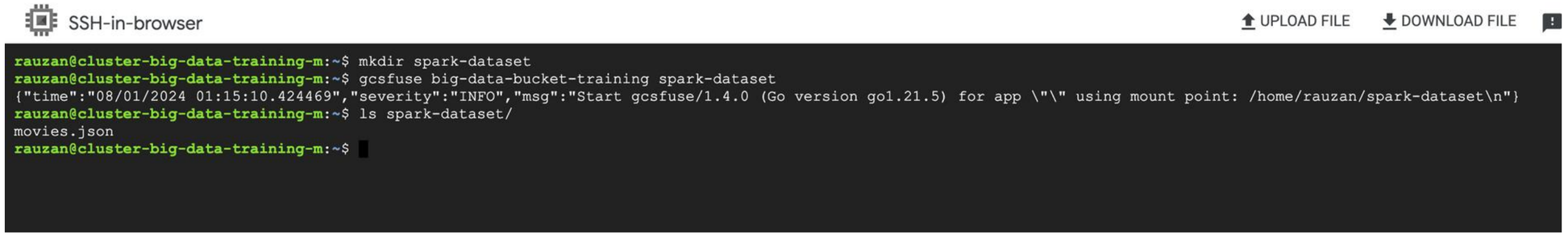
Create Storage Bucket

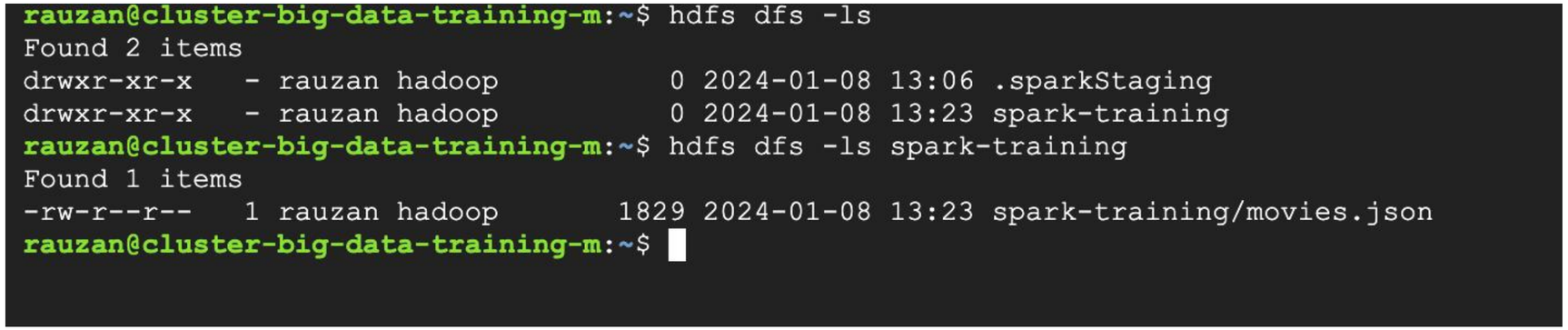



```
-big-data-training-m:~$ hdfs dfs -ls /user

hdfs   hadoop          0 2024-01-08 12:53 /user/dataproc
hdfs   hadoop          0 2024-01-08 12:53 /user/hbase
hdfs   hadoop          0 2024-01-08 12:53 /user/hdfs
hdfs   hadoop          0 2024-01-08 12:53 /user/hive
hdfs   hadoop          0 2024-01-08 12:53 /user/kafka
hdfs   hadoop          0 2024-01-08 12:53 /user/mapred
hdfs   hadoop          0 2024-01-08 12:53 /user/pig
rauzan hadoop          0 2024-01-08 13:00 /user/rauzan
hdfs   hadoop          0 2024-01-08 12:53 /user/solr
hdfs   hadoop          0 2024-01-08 12:53 /user/spark
hdfs   hadoop          0 2024-01-08 12:53 /user/yarn
hdfs   hadoop          0 2024-01-08 12:53 /user/zeppelin
hdfs   hadoop          0 2024-01-08 12:53 /user/zookeeper
```

Mount Storage via gcsfuse

export GCSFUSE_REPO=gcsfuse-`lsb_release -c -s`
echo "deb https://packages.cloud.google.com/apt $GCSFUSE_REPO main"|sudo tee /etc/apt/sources.list.d/gcsfuse.list
curl https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -
sudo apt-get update
sudo apt-get install gcsfuse
sudo usermod -a -G fuse $USER
exit





Command Sample

```
val movieDatas = spark.read.option("multiLine",true).json("hdfs://cluster-big-data-training-m:8020/user/rauzan/spark-training/movies.json")
```

```
movieDatas.sho
w()
```

```
val movieFilRatingrMoreThanThree =
movieDatas.filter("rating>3.0")
```

```
val movieFilYearMore2001=
movieFilRatingrMoreThanThree.filter("year>2001")
```

https://sparkbyexamples.com/spark/spark-shell-usage-with-examples/