

ETL Data Pipeline in GC

1. Here we'll be doing ETL and Reporting the data
2. We'll learn ETL pipeline and automate it

Overview

1. Initially we get requirements from client and we create dataflow diagrams and identify the resources needed and implement the small architecture for our project

Client requirements

1. Problem statement

Problem Statement

You are tasked with creating a data pipeline to extract employee data from various sources, mask sensitive information within the data, and load it into BigQuery. Additionally, you are required to develop a dashboard to visualize the employee data securely.

Requirements:

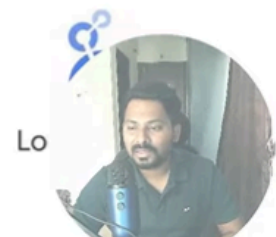
- **Data Extraction:** Extract employee data from multiple sources such as databases, CSV files, or APIs.
- **Data masking:** Identify sensitive information within the employee data, such as social security numbers, salary details, and personal contact information.
- **Data Loading into BigQuery:** Design a process to securely load extracted and masked employee data into Google BigQuery.
- **Dashboard Visualization:** Develop a web-based dashboard using visualization tools (e.g., Google Data Studio, Tableau, or custom dashboards).



2. These are the client requirements and we'll build these

3. Required GCP services

TechStack



4. Python for data extraction and we use cloud storage to store the extracted data
5. We use cloud data fusion as data pipeline ...it is no code feature in google cloud
6. Practical use case of cloud data fusion

Here's a real-time example of Cloud Data Fusion in action:

Imagine a ride-sharing company like Uber. They constantly receive a stream of data in real-time - new ride requests, driver locations, trip completions, etc. This data flows into a pub/sub topic (a message queue in Google Cloud).

Cloud Data Fusion can be configured to listen to this pub/sub topic continuously. As each new message arrives (e.g., a new ride request), Cloud Data Fusion triggers a real-time data pipeline:

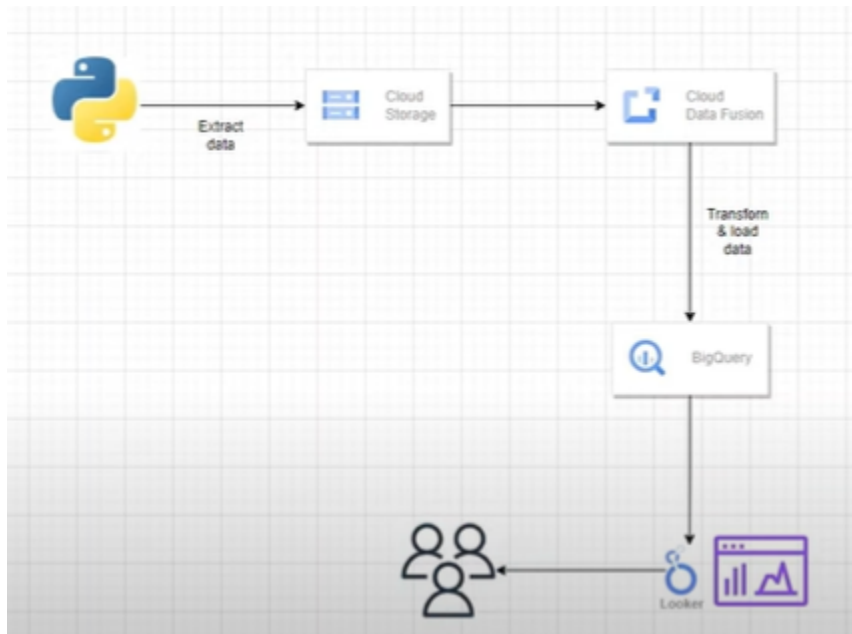
1. **Data Ingestion:** The pipeline reads the new ride request data from the pub/sub topic.
2. **Data Transformation:** The data is transformed to identify the pick-up location, drop-off location, and estimated fare.
3. **Data Enrichment:** Cloud Data Fusion might join this data with another dataset containing real-time traffic information. This enriches the data by predicting the trip duration and suggesting the most efficient route for the driver.
4. **Data Delivery:** Finally, the enriched data is sent to various destinations:
 - A database to store historical ride request information.
 - A dashboard for real-time tracking of ongoing rides.
 - An alert system to notify drivers about new ride requests in their vicinity.

7. We'll load the data into big query ..and we'll orchestrate this by using apache airflow

8. In the end we'll visualize this data in looker studio

Small Architecture

1. We design this architecture in app.diagram website



2. So here initially we store the data in cloud storage...then perform transformation using data fusion...then store this data in bigquery...after that we use looker table to visualize the data
3. Later we give dashboard to our manager's
4. And we use AirFlow/Cloud Composer to orchestrate all this things

Implementation

1. So first here we create composer environment ...and next we create data..refer playlist for composer and dataflow
- 2.

Extract

1. We'll extract the dummy data from python script

2. Basically we'll integrate our cloud storage in VsCode...and just convert our dummy data into CSV and we'll copy that in cloud storage

```
# Upload the CSV file to a GCS bucket
def upload_to_gcs(bucket_name, source_file_name, destination_blob_name):
    storage_client = storage.Client()
    bucket = storage_client.bucket(bucket_name)
    blob = bucket.blob(destination_blob_name)

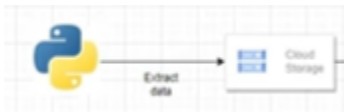
    blob.upload_from_filename(source_file_name)

    print(f'File {source_file_name} uploaded to {destination_blob_name} in {bucket_name}.')

# Set your GCS bucket name and destination file name
bucket_name = 'your_bucket_name'
source_file_name = 'employee_data.csv'
destination_blob_name = 'employee_data.csv'

# Upload the CSV file to GCS
upload_to_gcs(bucket_name, source_file_name, destination_blob_name)
```

3. So here we have extracted the data and loaded into cloud storage



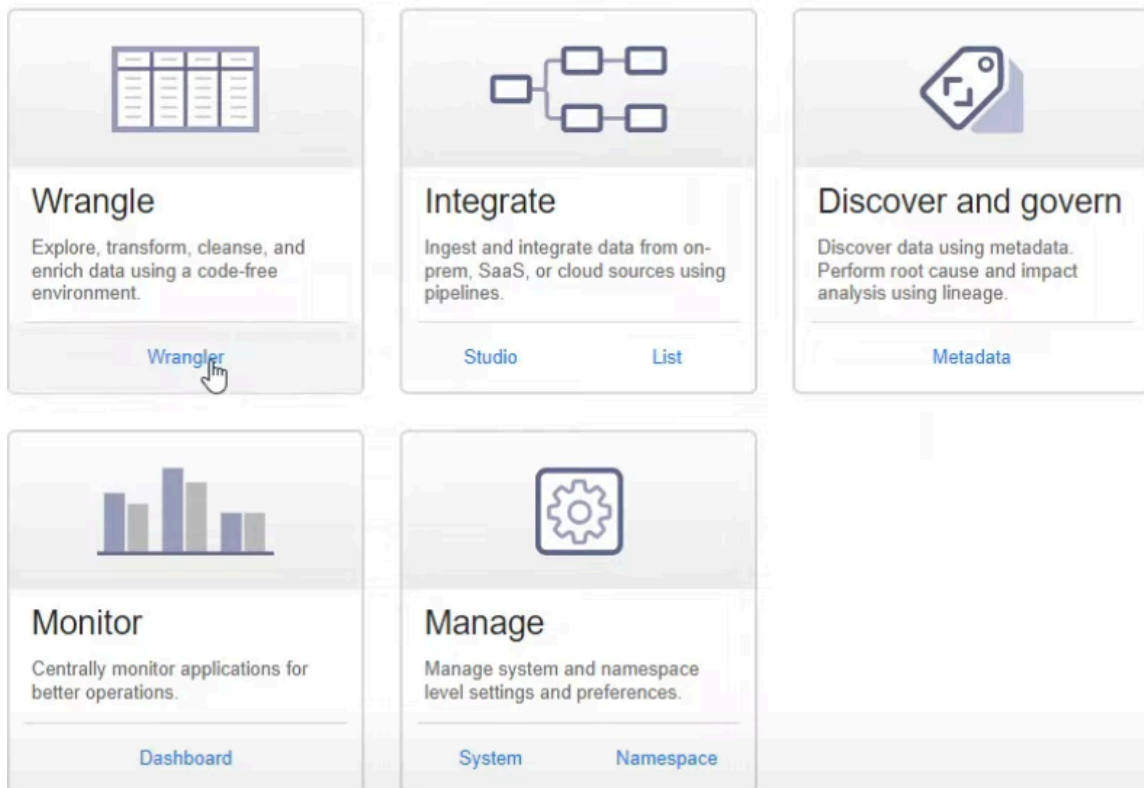
```
first_name,last_name,job_title,department,email,address,phone_number,salary,password
Jessica,Morrison,Electronics engineer,"Surveyor, planning and development",hunter03@example.net,"52328 Clay Highway
Allenfort, WA 62108",5903479394,97422,jFa7DL4N
Joshua,Farmer,Community arts worker,Amenity horticulturist,christina16@example.com,"6344 Montoya Isle
North Anthony, KY 35147",+1-479-415-8897,71116,nEXSu0C9
Sharon,Pratt,Forensic psychologist,Public affairs consultant,blake34@example.net,"Unit 6028 Box 1117
DPO AE 72949",001-834-393-8336x6881,82175,pFASD9IP
Jermaine,Bates,Press photographer,Aid worker,butlerstephanie@example.org,"Unit 5840 Box 0230
DPO AP 10736",787-339-3205x01614,86594,rWe8TsXC
Jodi,Brown,"Production designer, theatre/television/film","Editor, commissioning",sgreen@example.com,"Unit 0221 Box 8880
DPO AA 86442",001-381-643-9020x998,4555,qVb4VRxV
Debra,Patel,Art gallery manager,"Psychotherapist, dance movement",qturner@example.org,"51830 Schroeder Drive Apt. 274
Kristinberg, CA 57804",001-898-474-7179,33644,Oshr0EQQ
Glenn,May,Curator,Barrister's clerk,anthony62@example.org,"55667 Howard Road
Weaverbury, DC 11019",001-847-606-0692x5982,53104,JAGOm1r
Kathryn,Huber,"Programmer, systems",Museum/gallery curator,alexandrawilson@example.com,"PSC 2976, Box 0639
APO AP 96589",334-324-3756x22515,55314,Fj4PVFZi
Deborah,Suarez,Intelligence analyst,Educational psychologist,breanna67@example.org,"23183 Benjamin Rest Suite 612
Adamview, AS 82023",+1-463-639-0759x017,65327,7AvAn110
Caleb,Evans,Commissioning editor,Politician's assistant,curtisalyssa@example.com,"193 Mckee Parks Suite 440
West Brittany, CA 41629", (843)752-0511x5877,75813,eXVmq1FT
Derek,Kent,Management consultant,"Pilot, airline",bfox@example.net,"USNV Casey
FPO AE 42552",582-809-6350x554,99503,1ydivbG3
Melissa,Mason,"Teacher, primary school",Gaffer,kayla15@example.net,"410 Brian Forge Apt. 836
North Tyler, MI 64175",312.997.8234x2691,95337,MVms3H11
Amber,Mccarty,Outdoor activities/education manager,Hydrographic surveyor,williamsrichard@example.com,"50275 Palmer Extension
New Spencer, MI 44615",2905146712,22925,tCgXNupu
Brandon,Pena,Dispensing optician,Diagnostic radiographer,theresa28@example.com,"42331 Morgan Branch Suite 540
```

this is our data

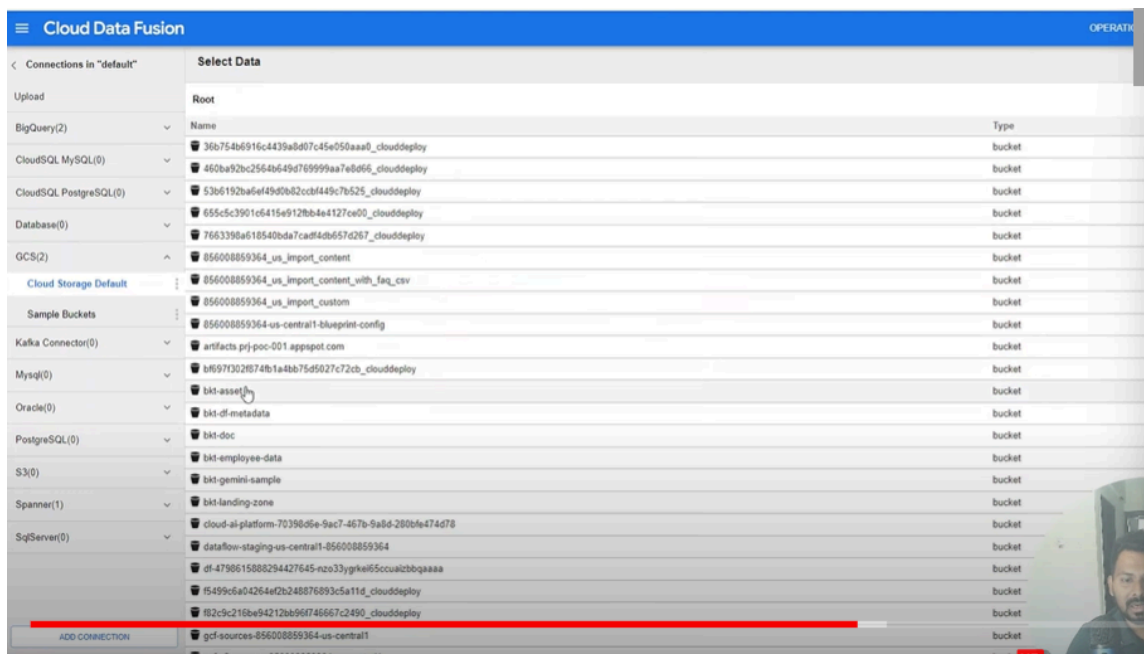
Cloud storage to Data fusion

1. Data fusion is a no code ETL tool provided by google cloud
2. In data flow we have to do our own coding

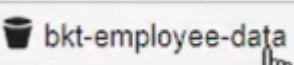
3. So in data fusion we have diff options



4. Here we'll go ahead with wrangler
5. Inside the wrangler the default connection is with Cloud storage



here we can see all the buckets that are in our GCS

6. We'll go ahead with our  and parse this data

7. Here is our data

Cloud Data Fusion | Wrangler

employee_data.csv

Cloud Storage Default - bid-employee-data/employee_data.csv

employee_data.csv Columns: 9 | Rows: 100

	String first_name	String last_name	String job_title	String department	String email	String address
1	Harold	Powell	Medical illustrator	Clinical molecular geneticist	markdavis@example.net	West Janeshire
2	Shelly	Delacruz	Physicist, medical	Scientist, water quality	lara33@example.net	Hayesborough
3	Christopher	Richardson	Astronomer	Chartered legal executive (England and Wales)	asmith@example.net	Weekshaven
4	Daniel	Reed	Sport and exercise psychologist	Equities trader	bakerjulia@example.com	Maryburgh
5	Glen	Berry	Barister	Patent examiner	webbamanda@example.net	Jacquelineside
6	Ashley	Miranda	Software engineer	Media planner	dobrah33@example.org	Stephensonville
7	Jessica	Lawrence	Event organiser	Nurse, adult	wchavez@example.com	Port Laceyberg
8	Karen	Davis	Landscape architect	Investment banker, operational	lauramoon@example.org	Lake James
9	Richard	Turner	Merchandise, retail	Archivist	ucopeland@example.com	Ingrammouth
10	Christie	Schultz	Geneticist, molecular	Therapist, speech and language	michaelperez@example.org	Frostfurt
11	Amy	Rocha	Surveyor, insurance	Animator	xmoore@example.net	Port Tammyfurt
12	Lisa	Hall	Armed forces operational officer	Sales promotion account executive	gary87@example.net	Jenkinsberg
13	Melanie	Liu	Health visitor	Special effects artist	stanley@example.org	New Rebecca
14	Cecilia	Williams	Clinical psychologist	Teaching laboratory technician	emily65@example.org	Metissafurt
15	Heather	Burgess	Naval architect	Podiatrist	jeremy60@example.net	Gallierzhaven

Columns (9)

Search

1 first_name
2 last_name
3 job_title
4 department
5 email
6 address
7 phone_number
8 salary
9 password

8. Now we have to perform some transformation

9. Here we first join first and last name and create full name column by using joins

	String first_name	String last_name
1	Harold	Powell
2	Shelly	Delacruz
3	Christopher	Richardson
4	Daniel	Reed
5	Glen	Berry
6	Ashley	Miranda
7	Jessica	Lawrence
8	Karen	Davis
9	Richard	Turner
10	Christie	Schultz
11	Amy	Rocha
12	Lisa	Hall

String Full_name
Harold.Powell
Shelly.Delacruz
Christopher.Richardson
Daniel.Reed
Glen.Berry
Ashley.Miranda
Jessica.Lawrence
Karen.Davis
Richard.Turner
Christie.Schultz

10. Next transformation is to mask sensitive data...like salary,phone number etc

String	String	String
phone_number	salary	password
Parse	561	E631bnHJ
Set character encoding	167	qbmQ40Ev
Change data type	452	cmawMyao
Format	521	chWbCn
Calculate	850	CU2GT5mZ
Custom transform	042	GuP8dW2s
Filter	472	VH1JkPkI
Send to error	252	VYD9km3Z
Find and replace	646	zyzBLJOF
Fill null or empty cells	970	coq9e7OB
Copy column	09	OP11wSsZ
Delete column	491	ghzHC5Ch
Keep column	576	M5JclAFQ
Join two columns	629	xNmFLxH
Swap two column names	Show last 4 characters only	
Extract fields	Show last 2 characters only	
Explode	Custom selection	
Define variable	By shuffling	
Set counter		
Mask data		
Encode		

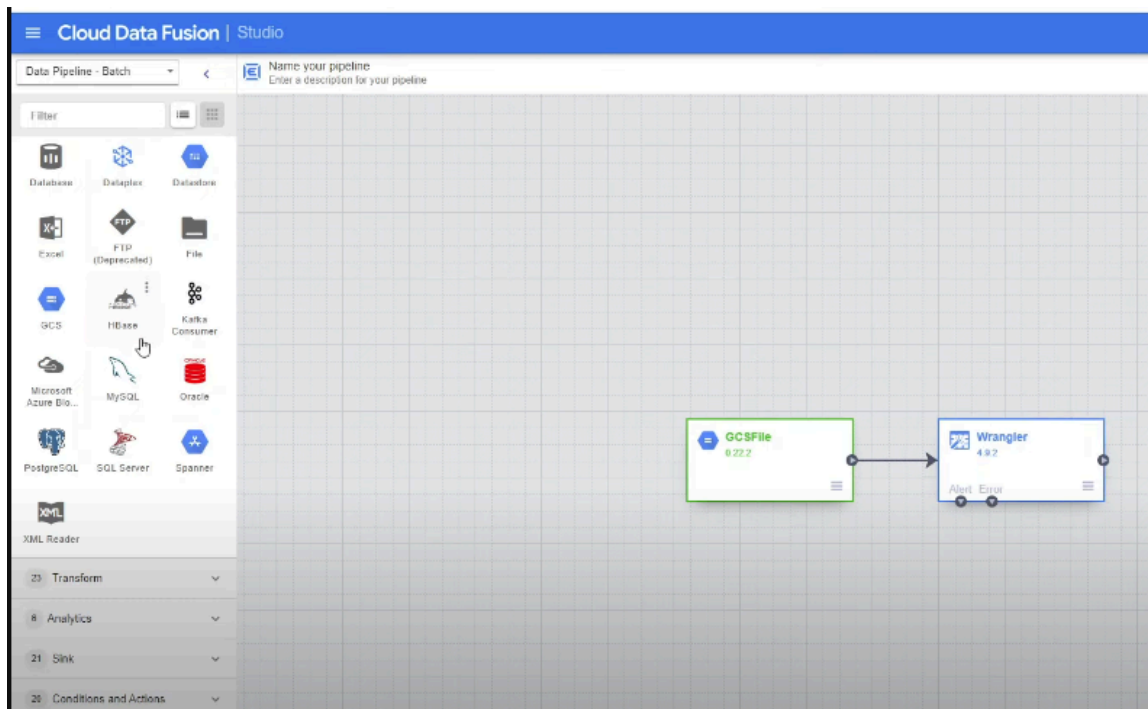
11. For password column ..we apply some encoding or some hashing to mask this

12. We can do what ever transformation we want by right clicking a column

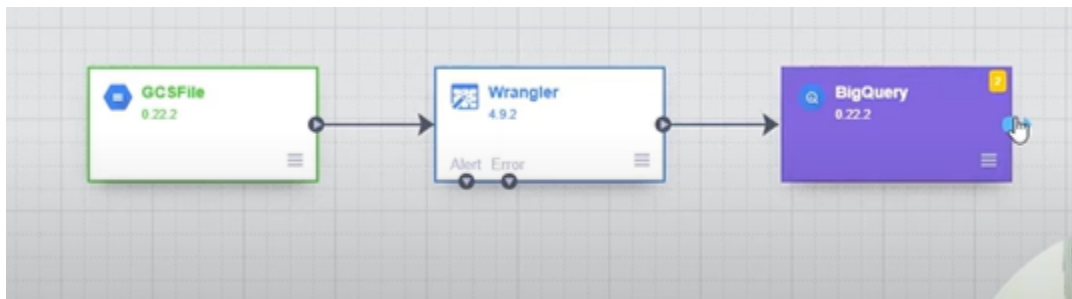
13. And here we have made 3 transformation

		Bytes	String		
		password	Full_name	Columns (10)	Transformation steps (3)
		Non-displayable object	Harold,Powell	# Transformations	
		Non-displayable object	Shelly,Delacruz	1 merge_first_name_last_name_Full_name_	
		Non-displayable object	Christopher,Richard	2 mask-number_salary xxxxx	
		Non-displayable object	Daniel,Reed	3 hash_password MD5 value	

14. Next we click on create pipeline ..and create a batch pipeline



15. And next we have to choose a sink..and we select big query as our sink



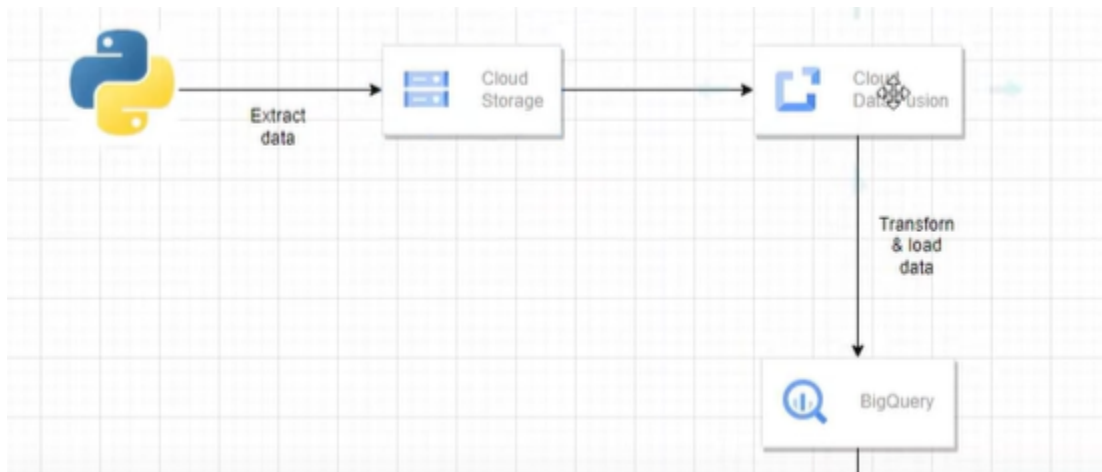
16. Next we create the big query dataset..in which we load our data

17. Next we give our pipeline name and dploy the pipeline..and run it

18. If there are any errors in our pipeline execution ..then we can go to logs and debug the errors

19. If the pipelines successfully runs...then we can our table inside the dataset that we have created in the big query

20. We have completed transformation and loaded the data in big query



Visualizations

1. Now from the data which is in bigquery...we will create lookup tables

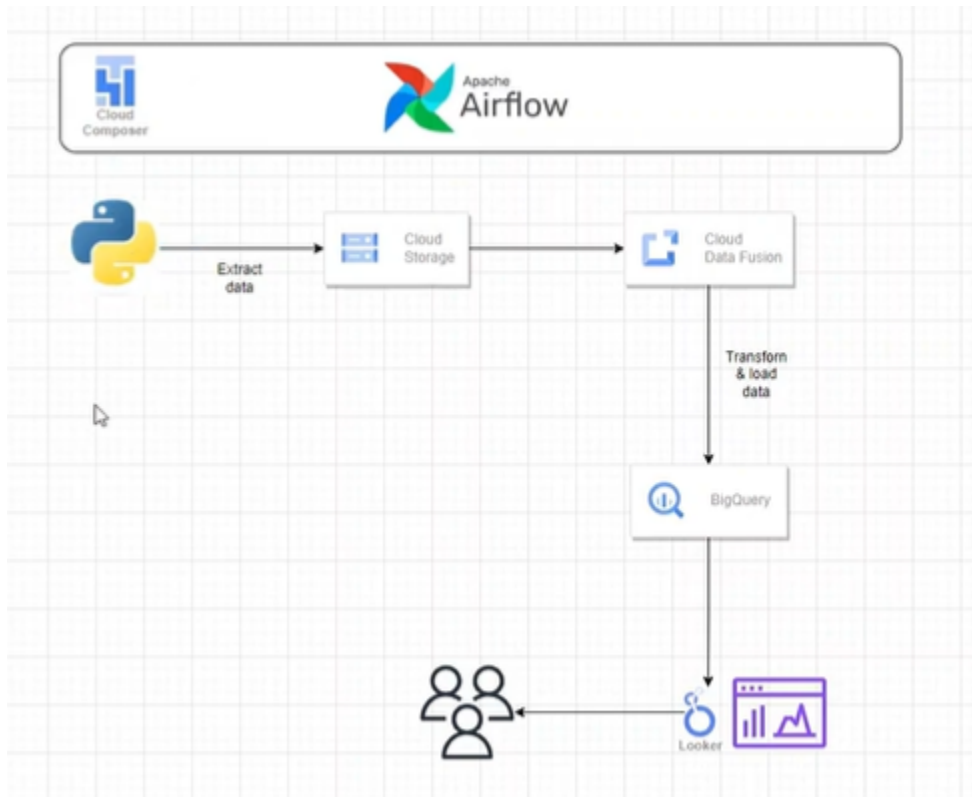


2.

here we have created a basic chart

Airflow Orchestration

1. Now we want to automate this architecture using Airflow

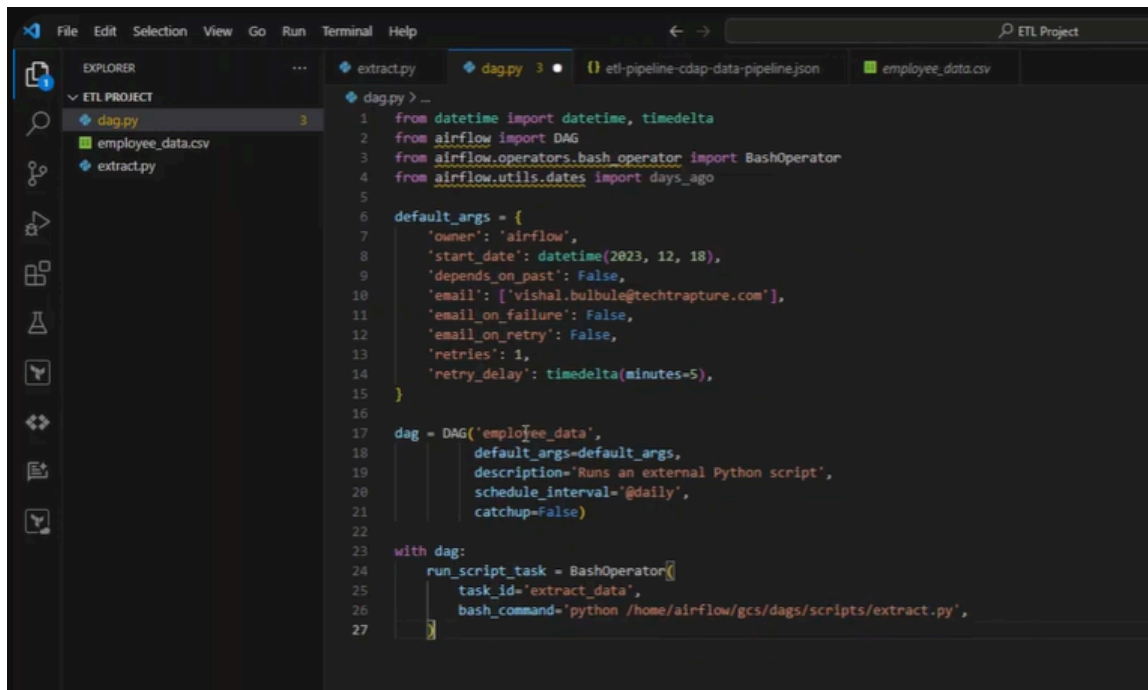


using cloud

composer environment

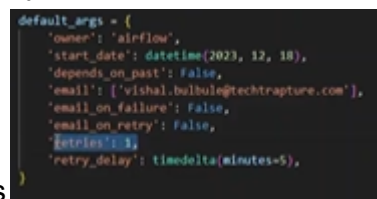
2. Here we created a cloud composer with no autoscaling and with default configs..and airflow will be hosted on this cloud composer environment
3. Next we create DAG in airflow to automate this
4. Follow his videos to create DAG

5. This is our DAG.py file



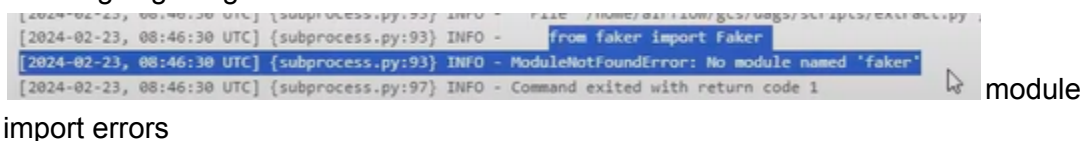
```
1 from datetime import datetime, timedelta
2 from airflow import DAG
3 from airflow.operators.bash_operator import BashOperator
4 from airflow.utils.dates import days_ago
5
6 default_args = {
7     'owner': 'airflow',
8     'start_date': datetime(2023, 12, 18),
9     'depends_on_past': False,
10    'email': ['vishal.bulbule@techrapture.com'],
11    'email_on_failure': False,
12    'email_on_retry': False,
13    'retries': 1,
14    'retry_delay': timedelta(minutes=5),
15 }
16
17 dag = DAG('employee_data',
18         default_args=default_args,
19         description='Runs an external Python script',
20         schedule_interval='@daily',
21         catchup=False)
22
23 with dag:
24     run_script_task = BashOperator(
25         task_id='extract_data',
26         bash_command='python /home/airflow/gcs/dags/scripts/extract.py',
27     )
```

6. Dag will get triggered by this and it runs extract.py file



```
default_args = {
    'owner': 'airflow',
    'start_date': datetime(2023, 12, 18),
    'depends_on_past': False,
    'email': ['vishal.bulbule@techrapture.com'],
    'email_on_failure': False,
    'email_on_retry': False,
    'retries': 1,
    'retry_delay': timedelta(minutes=5),
}
```

7. Default args
8. Our Dag is getting failed due to



```
[2024-02-23, 08:46:30 UTC] {subprocess.py:93} INFO - from faker import Faker
[2024-02-23, 08:46:30 UTC] {subprocess.py:93} INFO - ModuleNotFoundError: No module named 'faker'
[2024-02-23, 08:46:30 UTC] {subprocess.py:97} INFO - Command exited with return code 1
```

9. Best way is to deploy this modules/dependencies in composer env

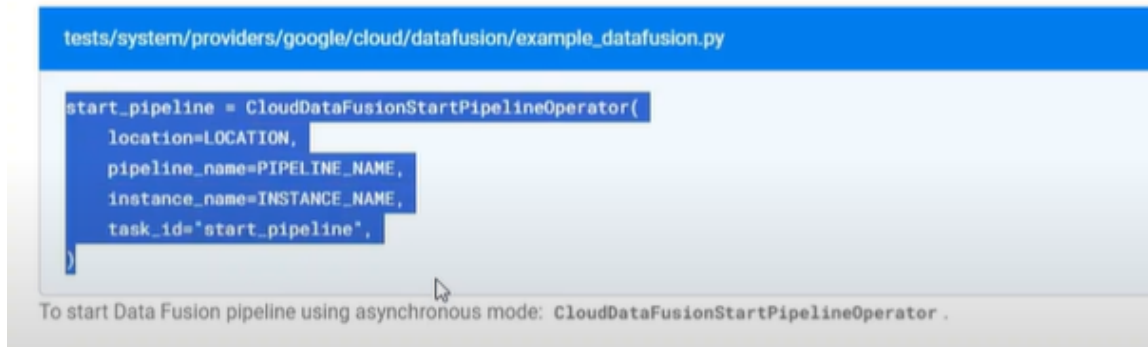


10. Now here we have completed this part in our DAG
11. Now we have to trigger the data fusion pipeline

12. To start the existing pipeline we use

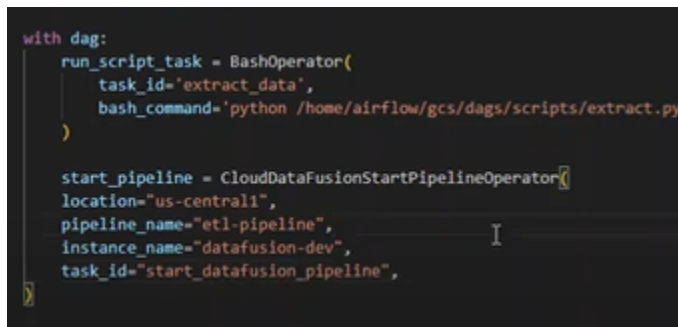
Start a DataFusion pipeline

To start Data Fusion pipeline using synchronous mode: `CloudDataFusionStartPipelineOperator` .



13. Find more on airflow providers google documentation

14. And here we have added our 2nd task



which starts the datafusion pipeline

after com

```
run_script_task >> start_pipeline
```

15. We'll add the dependencies and Next we'll update our DAG.py file inside airflow

16. We can see the status and identify the pipeline status here

deferred failed queued removed restarting running scheduled shutdown skipped success up_for_reschedule up_for_retry upstream_failed no_status