Team 4 Project Report

Data Engineering Project Report

02/2022

Table of Contents

Project objective

User Segmentation is one of the core data analysis problems of any marketing team as well as any company. A data processing system for this purpose, from extracting, loading and transforming data into understandable database tables is built in the scope of the project. The project inquiries users profile, daily transactions and promotions data from a payment application. This features the use of Spark, Airflow and MySQL, executed on Pycharm and VS Code using data consumed from local sources.

Team Members

TungNT121	Nguyễn Thanh Tùng
MinhNT53	Nguyễn Tuấn Minh
ManhPD13	Phan Đức Mạnh
NamDP14	Đinh Phương Nam

Product

Github: Github
GD: Management Folder

Requirements

- Building an User Segment system containing some information about customers
    - Read requirements in the file User_Segment.xlsx
    - Choose databases and design the tables
    - Design and implement the Data Pipeline to:
        - Sync data daily from source folder to destination folder and save as parquet files.
        - Transform data daily from destination folder to the final tables
    - Deploy Data Pipeline on the Airflow

Use Cases

Building an User Segment system containing information about customers	Choose databases and design the tables	Using MySQL and designing 6 tables: user_info, payment, active, userApp, userPmc and userPromotion
	Design and deploy the Data Pipeline	Using Airflow to schedule a pipeline daily
	Sync data daily from source folder to destination folder and save as parquet files	Using Pyspark to write data from source folder to datalake, as parquet files

	Transform data daily from destination folder to the final tables	Using Pyspark to connect and write to MySQL

Environment

Spark: spark-3.2.0-bin-hadoop3.2
Airflow: apache-airflow-2.2.4
JDK: openjdk 11.0.13
Python: Python 3.8.10
OS: Ubuntu 20.04 LTS

Database Design

Schemas

Name	Entity	Definition
User_Schema	userid	Id of each user, unique
	birthdate	date of birth
	profileLevel	1 Dont have phone number 2 Have phone number 3 KYC
	gender	1 Male 2 Female
	updatedTime	Datetime that user updated his/her info
Transactions_Schema	transId	Id of each transaction, unique
	transStatus	-1 Fail 0 - 1 Success
	userId	Id of user did the transaction
	transactionTime	Datetime of the transaction
	appId	10 Telco 11 Lazada 12 Tiki 13 Internet 14 Electricity

		15 Water 16 TV 17 123Phim 18 123Go 19 Shopee 20 Dominos 21 Game 22 Bitis
	transType	1 Fund In 2 Fund out 3 Payment 4 LiXi 5 Transfer
	amount	amount of the transaction
	pmcId	
Promotions_Schema	userid	Id of user having the promotion
	voucherCode	Code of the Voucher
	status	GIVEN/USED
	campaignID	Id of the promotion campaign
	time	
Campaign_Schema	campaignID	Id of the promotion campaign
	campaignType	
	expireDate	Date the campaign ends
	expireTime	Time the campaign ends after user received

Tables

user_info	userId	Id of each user, unique
	birthdate	date of birth
	gender	1 Dont have phone number 2 Have phone number 3 KYC

	profileLevel	1 Male 2 Female
	updatedTime	Datetime that user updated his/her info
active	userId	Id of each user, unique
	fistActiveDate	first day that any transaction of an user is recorded
	lastActiveDate	last day that any transaction of an user is recorded
	lastTransType	latest transaction type that an user is used
payment	userId	Id of each user, unique
	fistPaymentDate	first day that a payment of an user is recorded
	lastPaymentDate	last day that a payment of an user is recorded
	lastPaymentApp	latest app that an user is used
userApp	userId	Id of each user, unique
	appId	app that an user used
userPmc	userId	Id of each user, unique
	pmcId	
userPromotion	userid	Id of each user, unique
	campaignID	id of a promotion campaign
	status	USED/GIVEN
	Expire	TRUE/FALSE (1/0)

Data Pipeline

- Check availability of files
    - If data of a day is empty, pass that day

```
DATE EXECUTION 2022-02-28T09:21:33.054076+00:00
StoreDate in:  2022-02-27
No new users in 2022-02-27
No new transactions in 2022-02-27
No new promotions in 2022-02-27
```

    - If any data updated, update info in the database

```
INFO - DATE EXECUTION 2022-02-04T00:00:00+00:00
INFO - StoreDate in:  2022-02-03
INFO - Find new users
INFO - Find new transactions
INFO - Find new promotions
```

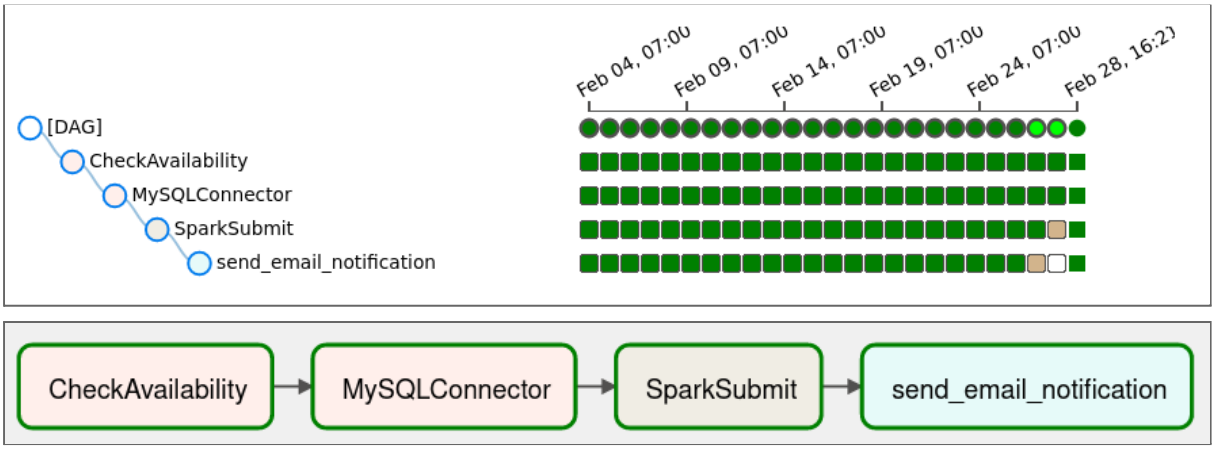
- Extract files from source
    - Read all data by date from directory
- Save files to data lake as parquet files
- Process data to suitable format
    - Update original columns to needed ones (fistPaymentDate, fistActiveDate, etc.)
- Load to database
    - Connect to MySQL database
    - Update tables in MySQL when any info is amended

```
INSERT NEW ID  22677
[ DB: 2022-01-10 07:51:23 | root: 2022-02-05 19:54:32 ]
26 days, 12:03:09
Update ID 22683
[ DB: 2021-11-21 21:36:44 | root: 2022-02-05 12:38:36 ]
75 days, 15:01:52
Update ID 22755
INSERT NEW ID  22767
[ DB: 2022-01-28 22:30:27 | root: 2022-02-05 21:11:47 ]
```

- Send email notification
    - Send email to people in charge if all tasks run successfully or failed

```
AIRFLOW_CTX_DAG_EMAIL=createdforthesis@gmail.com,manhpd.esp@gmail.com,tuanminh7122001@gmail.com
AIRFLOW_CTX_DAG_OWNER=airflow
AIRFLOW_CTX_DAG_ID=DEProject_Final
AIRFLOW_CTX_TASK_ID=send_email_notification
AIRFLOW_CTX_EXECUTION_DATE=2022-02-04T00:00:00+00:00
AIRFLOW_CTX_DAG_RUN_ID=scheduled__2022-02-04T00:00:00+00:00
```

Airflow scheduler



User Guide

Installation

Airflow

```
pip install "apache-airflow[celery]==2.2.4" --constraint
"https://raw.githubusercontent.com/apache/airflow/constraints-2.2.
4/constraints-3.8.txt"
airflow db init
airflow webserver --port 8080
airflow scheduler
```

Achieve Airflow UI at http://localhost:8080

MySQL

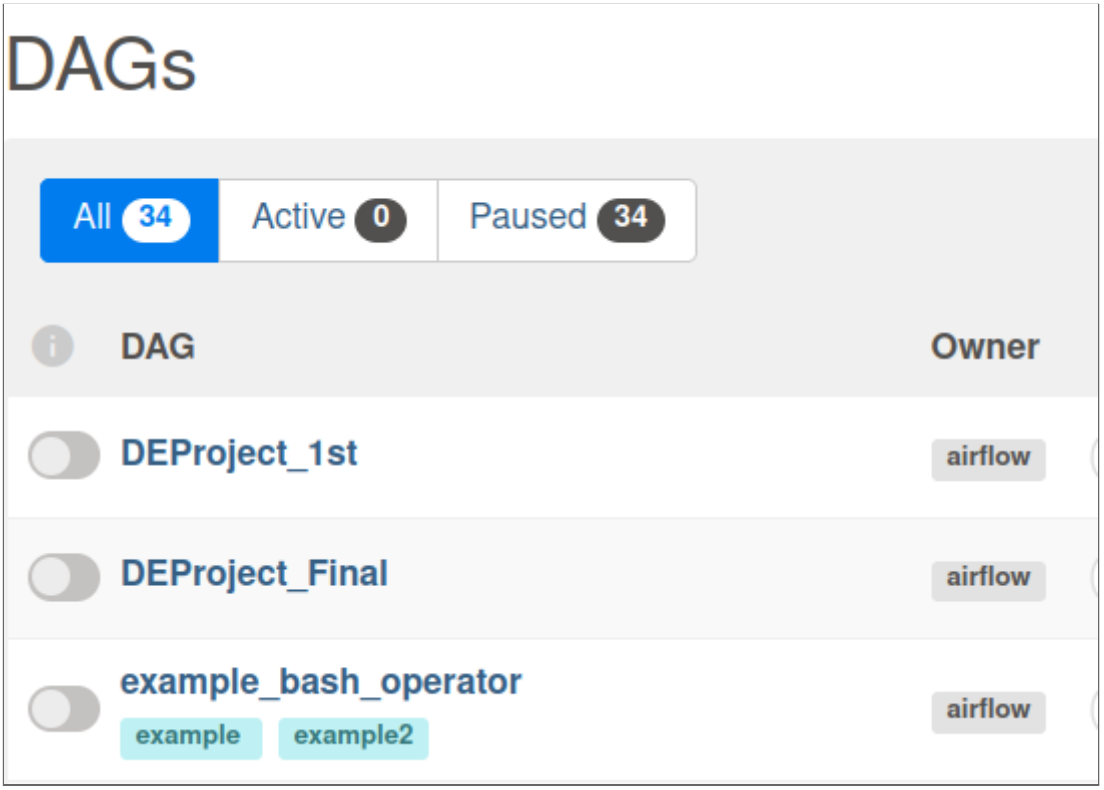
```
git clone https://github.com/bitnami/bitnami-docker-mysql.git
cd bitnami-docker-mysql
docker-compose up -d
```

Spark

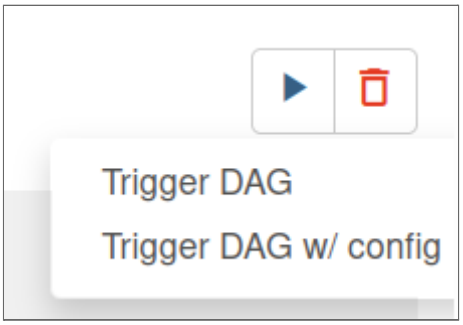
Follow https://phoenixnap.com/kb/install-spark-on-ubuntu to install spark and start master and workers nodes

Running

- On Airflow UI, select DEProject_Final DAG



- Choose option Trigger DAG to run pipeline



Results

After pushing to MySQL database, we have up-to-date tables, ready to be used by marketers

```
mysql> show tables;
+--------------------+
| Tables_in_project |
+--------------------+
| active             |
| payment            |
| userApp            |
| userPmc            |
| userPromotion      |
| user_info          |
+--------------------+
6 rows in set (0.01 sec)
```

```
mysql> select * from user_info limit 10;
+--------+------------+--------------+--------+---------------------+
| userid | birthdate  | profileLevel | gender | updatedTime         |
+--------+------------+--------------+--------+---------------------+
| 1      | 1978-03-11 |            1 |      2 | 2021-11-18 06:31:41 |
| 10     | 1960-01-12 |            3 |      1 | 2022-02-06 23:44:19 |
| 10001  | 2000-01-02 |            3 |      2 | 2022-01-04 23:23:56 |
| 10002  | 1970-01-13 |            1 |      1 | 2022-02-13 16:11:59 |
| 10003  | 1996-12-10 |            1 |      1 | 2022-01-02 20:53:16 |
| 10004  | 1960-07-28 |            2 |      1 | 2022-01-16 14:00:16 |
| 10005  | 1978-05-27 |            3 |      2 | 2022-01-30 09:06:40 |
| 10007  | 1997-02-12 |            2 |      2 | 2021-11-30 21:49:37 |
| 10010  | 1965-04-13 |            2 |      1 | 2021-11-01 19:23:37 |
| 10011  | 1960-01-15 |            2 |      2 | 2022-02-18 02:48:57 |
+--------+------------+--------------+--------+---------------------+
10 rows in set (0.00 sec)

mysql> select count(*) from user_info;
+----------+
| count(*) |
+----------+
|    26357 |
+----------+
1 row in set (0.01 sec)
```

```
mysql> select * from payment limit 10;
+--------+---------------------+---------------------+----------------+
| userid | fistPaymentDate     | lastPaymentDate     | lastPaymentApp |
+--------+---------------------+---------------------+----------------+
| 10     | 2021-11-30 14:39:26 | 2022-01-09 03:01:27 |             34 |
| 10002  | 2022-01-31 05:32:26 | 2022-01-31 05:32:26 |            100 |
| 10005  | 2022-01-10 02:36:09 | 2022-01-10 02:36:09 |             94 |
| 10007  | 2022-02-05 20:33:41 | 2022-02-05 20:33:41 |             89 |
| 1001   | 2021-11-30 06:04:31 | 2022-01-16 04:33:36 |             63 |
| 10010  | 2022-02-12 22:49:04 | 2022-02-18 06:40:11 |             82 |
| 10011  | 2021-12-17 00:59:36 | 2021-12-17 00:59:36 |             27 |
| 10012  | 2021-12-18 08:08:04 | 2022-01-16 16:41:59 |            102 |
| 10015  | 2021-11-30 20:16:58 | 2021-11-30 20:16:58 |             28 |
| 10021  | 2022-01-29 02:37:48 | 2022-01-29 02:37:48 |             68 |
+--------+---------------------+---------------------+----------------+
10 rows in set (0.00 sec)

mysql> select count(*) from payment;
+----------+
| count(*) |
+----------+
|    16375 |
+----------+
1 row in set (0.01 sec)
```

```
mysql> select * from active limit 10;
+--------+---------------------+---------------------+---------------+
| userId | fistActiveDate      | lastActiveDAte      | lastTransType |
+--------+---------------------+---------------------+---------------+
| 1      | 2021-11-02 04:18:52 | 2022-02-04 08:26:44 |             4 |
| 10     | 2021-11-30 14:39:26 | 2022-02-17 05:54:44 |             5 |
| 100    | 2021-11-01 09:25:52 | 2022-01-19 11:30:08 |             5 |
| 10000  | 2021-11-14 14:11:38 | 2021-12-30 04:44:45 |             2 |
| 10001  | 2022-02-02 18:04:57 | 2022-02-02 18:04:57 |             2 |
| 10002  | 2022-01-08 21:30:32 | 2022-01-31 05:32:26 |             3 |
| 10004  | 2022-01-12 11:20:48 | 2022-02-06 03:16:09 |             4 |
| 10005  | 2021-11-12 13:35:19 | 2022-01-10 02:36:09 |             3 |
| 10006  | 2021-12-18 15:29:52 | 2022-01-09 10:36:59 |             5 |
| 10007  | 2021-11-15 01:33:48 | 2022-02-05 20:33:41 |             3 |
+--------+---------------------+---------------------+---------------+
10 rows in set (0.00 sec)

mysql> select count(*) from active;
+----------+
| count(*) |
+----------+
|    37237 |
+----------+
1 row in set (0.01 sec)
```

```
mysql> select * from userApp limit 10;
+--------+-------+
| userId | appId |
+--------+-------+
| 1      | 1     |
| 1      | 4     |
| 1      | 5     |
| 10     | 1     |
| 10     | 2     |
| 10     | 34    |
| 10     | 45    |
| 10     | 5     |
| 100    | 2     |
| 100    | 5     |
+--------+-------+
10 rows in set (0.00 sec)

mysql> select count(*) from userApp;
+----------+
| count(*) |
+----------+
|    87167 |
+----------+
1 row in set (0.01 sec)
```

```
mysql> select * from userPmc limit 10;
+--------+-------+
| userId | pmcId |
+--------+-------+
| 1      | 1     |
| 1      | 2     |
| 1      | 4     |
| 10     | 1     |
| 10     | 2     |
| 10     | 3     |
| 10     | 4     |
| 100    | 1     |
| 100    | 2     |
| 100    | 3     |
+--------+-------+
10 rows in set (0.00 sec)

mysql> select count(*) from userPmc;
+----------+
| count(*) |
+----------+
|    77962 |
+----------+
1 row in set (0.00 sec)
```

```
mysql> select * from userPromotion limit 10;
+--------+------------+--------+---------------------+
| userid | campaignID | status | actualExpire        |
+--------+------------+--------+---------------------+
|      1 |       1000 | GIVEN  | 2022-01-01 00:00:00 |
|      2 |       1001 | GIVEN  | 2022-01-01 00:00:00 |
|      3 |       1004 | GIVEN  | 2022-01-01 00:00:00 |
|     14 |       1002 | GIVEN  | 2022-01-01 00:00:00 |
|     15 |       1005 | GIVEN  | 2022-01-01 00:00:00 |
|     20 |       1006 | GIVEN  | 2022-01-01 00:00:00 |
|     26 |       1005 | GIVEN  | 2022-01-01 00:00:00 |
|     27 |       1000 | GIVEN  | 2022-01-01 00:00:00 |
|     31 |       1000 | GIVEN  | 2022-01-01 00:00:00 |
|     36 |       1006 | GIVEN  | 2022-01-01 00:00:00 |
+--------+------------+--------+---------------------+
10 rows in set (0.00 sec)
```

```
mysql> select count(*) from userPromotion;
+----------+
| count(*) |
+----------+
|    63077 |
+----------+
1 row in set (0.01 sec)
```

Relevant problems

In development mode, Airflow database (SQLite) is overloaded due to large amount of data
Solution: Using MySQL or Postgres in production

Conclusion and Future work

Conclusion

In the scope of the project, a data processing system is built. It includes MySQL as a database, a Spark job to run all tasks and Airflow for scheduling and orchestration of data pipelines. Using data from sources, the system produces understandable tables at the end of the pipeline and could be used for analytical steps. It is an irreplaceable part before marketing process.

Future work

The system collects data from multiple sources using Kafka
Other databases (Postgres, MongoDB, etc) could be used to store data