

DATU MASIBOEN PROZESAMENDURAKO AZPIEGITURAK

2.LANA

Andoni Sudupe eta Oier Ijurko

Aurkibidea

1.	Sarrera.....	3
2.	Datuak.....	4
3.	Erabili ditugun tresnak.....	4
3.1.	Kudu.....	4
3.2.	Impala.....	5
3.3.	Superset.....	6
4.	Lana garatzeko prozesua.....	6
4.1.	Compose-ak eta impyla.....	6
4.2.	Network aldaketak.....	7
4.3.	Parquet fitxategia sartu Supersetara.....	7
4.4.	Impala Bash.....	8
4.5.	SQL deiak.....	9
4.6.	Dashboard eta Chart-ak.....	10
5.	Ondorioak.....	12
6.	Erreferentziak.....	12

1. Sarrera

Lan honen helburu nagusia Docker Compose bidez Kudu eta Impala ingurune bat hedatzea da. Ingurune honen laguntzarekin biltegitratuko ditugun datuak Superset erabilia bistaratuko ditugu.

Lanaren funtzionamendua probatzeko pauso hauek emango ditugu:

1. New York-eko taxiek egindako bidaiak biltzen dituen Parquet datubasea jaitsi.
2. Fitxategi lokal hau Impala-n kanpo taula bat bezala montatu (Parquet fitxategi external bat bezala).
3. Taula osoaren kopia bat egin, Impala-n, Kudun biltegitratuz “taxik” izenarekin
4. Superset Impala-ra konektatu, bi taulen gainean kontsultak egin ahal izateko.
5. Bi tauletako informazioarekin grafikoak/taulak sortu. Hauekin panel bat diseinatu.

2. Datuak

Erabili ditugun datuak New Yorkeko taxiak egindako bidaiak gordetzen dituen datubase batetik hartu ditugu, zehazki, “Yellow Taxi” izeneko enpresako taxien bidaiak. Datu hauek Parquet formatuko fitxategiak dira, hau da, datu biltegitratze formatu bat. Formatu honen berezitasuna, kodigo irekikoa izateaz gain, datuak zutabetan biltegitratzen dituela da, horrela, datu multzo erraldoiak azkarrago kudeatuz, datuen konpresio hobeago bat lortzen baita.

Parquet fitxategi honetan 3 milioi tupla daude eta bakoitzean 19 kanpo; bidaiari kopurua, egindako distantzia, nola ordaindu duten, eta beste informazio gehiago gordetzen da tupla bakoitzean.

Hona hemen erabiliko dugun fitxategiak gordetzen dituen kanpo batzuk eta bertako tupla batzuen balioak:

vendorid	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count	trip_distance	ratecodeid	store_and_fwd_flag	pulocationid	dolocationid
2	2022-02-24T11:13:16	2022-02-24T11:31:18	1	2.83999999141693115	1	N	161	158
2	2022-02-24T11:34:32	2022-02-24T11:56:19	1	1.73000000190734863	1	N	68	164
1	2022-02-24T11:27:01	2022-02-24T11:34:49	1	0.8000000011920929	1	N	229	141
1	2022-02-24T11:53:08	2022-02-24T12:23:31	1	10.1999999809265137	1	N	75	200
1	2022-02-24T11:21:35	2022-02-24T11:37:02	3	2.09999999046325684	1	N	239	75
1	2022-02-24T11:55:27	2022-02-24T11:58:24	3	0.5	1	N	263	236
2	2022-02-24T11:05:42	2022-02-24T11:15:40	1	0.98000000190734863	1	N	48	162
2	2022-02-24T11:21:24	2022-02-24T11:45:02	1	10.4700000267028809	1	N	233	138
2	2022-02-24T11:12:28	2022-02-24T11:34:17	1	10.5200000457763672	1	N	229	138
1	2022-02-24T11:04:10	2022-02-24T11:15:16	2	1.8999999976158142	1	N	142	151
1	2022-02-24T11:17:39	2022-02-24T11:26:12	1	1.3999999976158142	1	N	238	43

3. Erabili ditugun tresnak

Lan hau garatzeko hainbat tresna erabili ditugu. Hauek, datuak lortu, gorde eta bistaratzeko erabiliko ditugu.

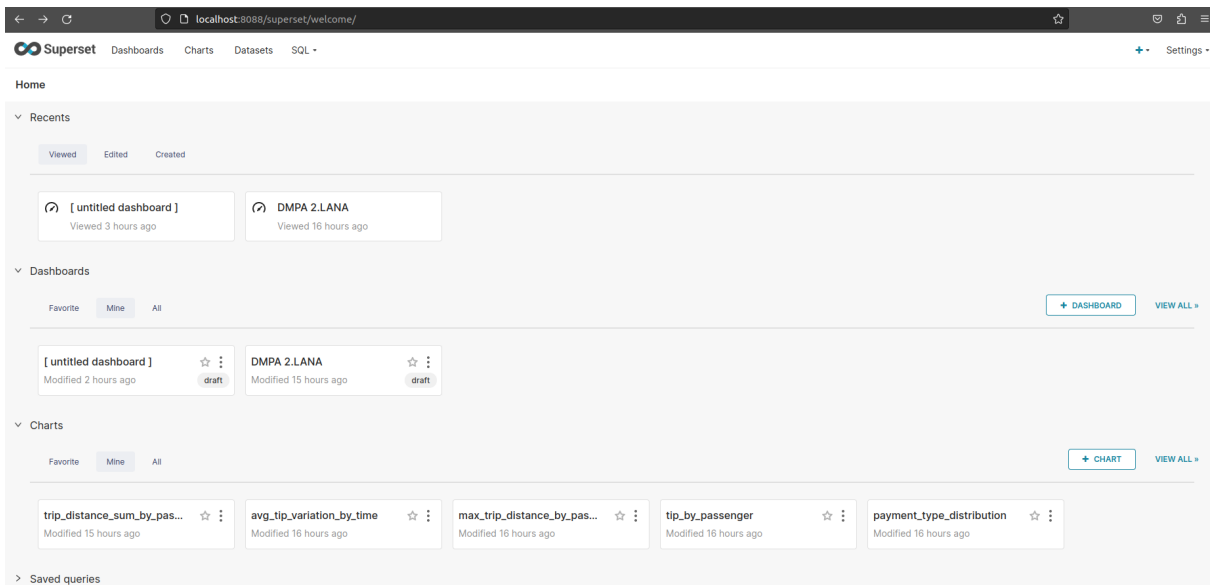
3.1. Kudu

Apache Kudu datuak biltegitratzeko erabiltzen den software libre eta irekia da. Hadoop-en biltegitratze-geruzari integritatea ematen dio, datu azkarretan analisi azkarrak egin ahal izateko. OLAP lan-kargetarako diseinatu eta optimizatuta dago. HBase-ren antzera, denbora errealeko biltegitratze-tresna bat da, kodean indexatutako erregistroak bilatzen laguntzen duena.

3.3. Superset

Apache Superset kode irekiko software-aplikazio bat da, datuak arakatzeko eta datuak bistaratzeko balio duena. Bere eginkizun nagusia, big dataekin lan egitea da.

Baliabideen artean, datasetak gorde, datu horiekin grafikoak sortu eta grafiko multzo batekin “dashboard”-ak sortzea daude. Dashboard hauekin, big-dataren bistaratze grafiko orokor bat egin ahal izango dugu.



4. Lana garatzeko prozesua

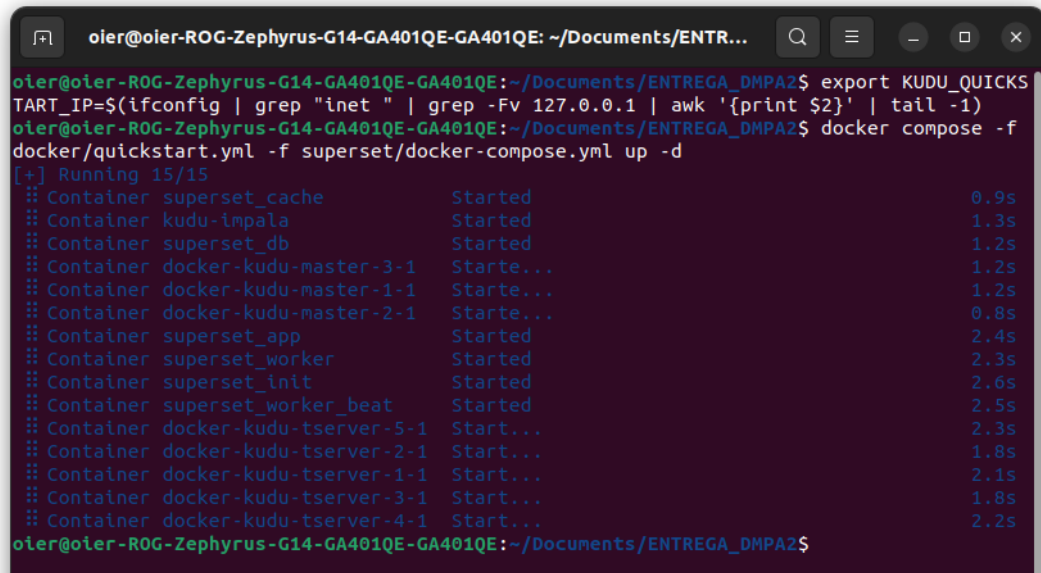
4.1. Compose-ak eta impyla

Aurreko lanean azaldu bezala, .yaml fitxategietan adierazten dira erabili nahi diren kontainerak. Kasu honetan, impala kontainer bat erabili nahi dugunez, superseteko quickstart.yaml fitxategian kudu-impala izeneko kontainer bat sortuko dugu apache/kudu:impala-latest iruditik. Eskubiko irudian agertzen den moduan.

```
202 impala:
203   image: apache/kudu:impala-latest
204   container_name: kudu-impala
205   ports:
206     - "21000:21000"
207     - "21050:21050"
208     - "25000:25000"
209     - "25010:25010"
210     - "25020:25020"
211   networks:
212     - net1
213   command: ["impala"]
```

Bestalde, superset kontainerra impalarekin konektatu nahi bada, driver batzuk behar ditugu, hauetako bat “Impyla” izanik, baina kasu honetan erabiltzen dugun superset-en irudiak ez ditu driver horiek barnean, beraz, requirements-local testu fitxategi bat sortu dugu superset/docker/ karpetaaren barruan instalatu behar diren driverrakin. Behar diren beste driverrak “thrift” eta “thrift_sasl” dira, baina

instalatzen dugun Impylan dagoeneko driver hauek badaude, beraz requirements fitxategi honetan adierazi beharrik ez daude.

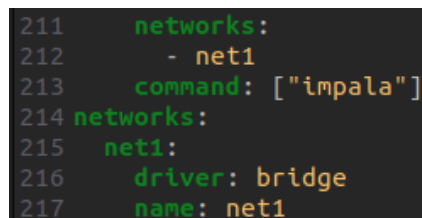


```
oier@oier-ROG-Zephyrus-G14-GA401QE-GA401QE: ~/Documents/ENTR...
oier@oier-ROG-Zephyrus-G14-GA401QE-GA401QE:~/Documents/ENTREGA_DMPA2$ export KUDU_QUICKSTART_IP=$(ifconfig | grep "inet " | grep -Fv 127.0.0.1 | awk '{print $2}' | tail -1)
oier@oier-ROG-Zephyrus-G14-GA401QE-GA401QE:~/Documents/ENTREGA_DMPA2$ docker compose -f docker/quickstart.yml -f superset/docker-compose.yml up -d
[+] Running 15/15
:: Container superset_cache Started 0.9s
:: Container kudu-impala Started 1.3s
:: Container superset_db Started 1.2s
:: Container docker-kudu-master-3-1 Starte... 1.2s
:: Container docker-kudu-master-1-1 Starte... 1.2s
:: Container docker-kudu-master-2-1 Starte... 0.8s
:: Container superset_app Started 2.4s
:: Container superset_worker Started 2.3s
:: Container superset_init Started 2.6s
:: Container superset_worker_beat Started 2.5s
:: Container docker-kudu-tserver-5-1 Start... 2.3s
:: Container docker-kudu-tserver-2-1 Start... 1.8s
:: Container docker-kudu-tserver-1-1 Start... 2.1s
:: Container docker-kudu-tserver-3-1 Start... 1.8s
:: Container docker-kudu-tserver-4-1 Start... 2.2s
oier@oier-ROG-Zephyrus-G14-GA401QE-GA401QE:~/Documents/ENTREGA_DMPA2$
```

Proiektua martxan jarri ahal izateko, goiko irudian agertzen den bezala, KUDU_QUICKSTART_IP-a ongi konfiguratu eta ondoren “docker compose -f docker/quickstart.yml -f superset/docker-compose.yml up -d” komandoa exekutatu behar da. Modu honetan, proiektua gauzatzeko beharrezko diren compose fitxategi guztiak hedatuko ditugu.

4.2. Network aldaketak

Ditugun kontainer guztiak batera lan egin dezaten, sare berean egon behar dira definituak, hau dela eta, .yml fitxategietan sortzen diren container guztiei lana egiteko sarea esleituko diegu, “net1” izenekoa. Eskubiko irudian ikus daiteke nola esleitzen diegun kontainer bakoitzari “net1” izeneko sarea.



```
211     networks:
212     - net1
213     command: ["impala"]
214 networks:
215     net1:
216         driver: bridge
217         name: net1
```

4.3. Parquet fitxategia sartu Supersetara

Kontainer guztiak exekutatu ondoren, datuekin lan egiten hasteko, gure datuak Superset-en sartu behar ditugu. Lehenik eta behin, datubasearekin konexioa esleitu behar dugu, impala://kudu-impala:21050 jarrita. “Impala” zerbitzari mota da eta “kudu-impala” impala zerbitzariaren izena. Honekin konexioa sortu dugu, baina ez digu utziko ezer egiten, baimenik ez duelako. Hau konpontzeko, konexioaren “security” sekzioan “Allow files uploads to database” baimenak eman behar zaizkio. SQL LAB-en SQL deiak egin ahal izateko, toki berdinean, baimen denak eman beharko ditugu. Honekin datubasea sartu ahalko dugu, baita aldaketak egin ere.

Azkenik, datuak sartuko ditugu supersetara, gure kasuan parquet fitxategi bat, beraz, “Upload columnar file to database” egin behar dugu, lehen esan bezala, zutabeetan orientatutako formatua baita. Datu asko daudenez, denbora pixkat ematen dio.

4.4. Impala Bash

“`docker exec -it kudu-impala impala-shell`” egiten badugu docker compose up egin ondoren, impala-shell bat irekitzeko gai izango gara. Bertan, gure datuen gainean SQL deiak egin ahal izango ditugu.

Lanaren zati bat, taxi taularen gainean egindako kopia bat egitea da, “taxik” izenakoa. Horretarako, impala-shell-ean ondorengo egingo dugu:

```
olier@olier-R0G-Zephyrus-G14-GA401QE-GA401QE:~/Documents/kudu$ docker exec -it kudu-impala impala-shell
Starting Impala Shell without Kerberos authentication
Opened TCP connection to 18e8bf7492e9:21000
Connected to 18e8bf7492e9:21000
Server version: impalad version 3.4.0-RELEASE RELEASE (build Could not obtain git hash)
*****
Welcome to the Impala shell.
(Impala Shell v3.4.0-RELEASE (9f1c31c) built on Fri Apr 24 14:10:19 PDT 2020)

The SET command shows the current value of all shell and query options.
*****
[18e8bf7492e9:21000] default> CREATE TABLE taxik LIKE taxi
> ;
Query: CREATE TABLE taxik LIKE taxi
+-----+
| summary |
+-----+
| Table has been created. |
+-----+
Fetched 1 row(s) in 0.06s
[18e8bf7492e9:21000] default> INSERT INTO TABLE taxik SELECT * FROM taxi
> ;
Query: INSERT INTO TABLE taxik SELECT * FROM taxi
Query submitted at: 2023-04-28 17:26:30 (Coordinator: http://18e8bf7492e9:25000)
Query progress can be monitored at: http://18e8bf7492e9:25000/query_plan?query_id=7f4b267c50089242:f1a0cbcd00000000
[
[
[
[
[
[#####
[#####
```

`CREATE TABLE taxik LIKE taxi;` -k Superset-en gordeta dugun taxi taularen gorpuzkera (kanpo eta datu-motak) kopiatuko ditu “taxik” taula berri batean.

`INSERT INTO TABLE taxi SELECT * FROM taxi;` -k taxi taularen eduki dena taxik taulan sartuko du, bi taula berdin lortuz.

Hau egin ondoren, impala-shell-aren bitartez bi taulen (taxi eta taxik) gainean SQL deiak egin ahal izango ditugu. Hona hemen SQL dei baten emaitza:


```
[e06d3989fbc7:21000] default> SELECT MAX(passenger_count) FROM taxi;
Query: SELECT MAX(passenger_count) FROM taxi
Query submitted at: 2023-04-30 19:15:36 (Coordinator: http://e06d3989fbc7:25000)
Query progress can be monitored at: http://e06d3989fbc7:25000/query_plan?query_id=b34b3c382b8264a3:42d1c03a00000000
+-----+
| max(passenger_count) |
+-----+
| 9                     |
+-----+
Fetched 1 row(s) in 0.41s
[e06d3989fbc7:21000] default> SELECT MAX(passenger_count) FROM taxik;
Query: SELECT MAX(passenger_count) FROM taxik
Query submitted at: 2023-04-30 19:15:38 (Coordinator: http://e06d3989fbc7:25000)
Query progress can be monitored at: http://e06d3989fbc7:25000/query_plan?query_id=6c4f884dd69930e7:e6124b8100000000
+-----+
| max(passenger_count) |
+-----+
| 9                     |
+-----+
Fetched 1 row(s) in 0.41s
[e06d3989fbc7:21000] default>
```

Kasu honetan, taxi batek aldi berean izan dituen bidaiari kopuru maximoa bueltatzeko eskatu diogu, eta bi taulak berdinak direnez, erantzuna berdina da: 9 bidaiari.

4.5. SQL deiak

Konexioa ondo egin bada, Superset-en taxi eta taxik taulen gainean SQL deiak egin ahal izango dugu. Chart-ak eta Dashboard-a sortu baino lehen, SQL LAB-en dei batzuk egin ditugu dena ondo egin dagoela ziurtatzeko. Hona hemen adibide batzuk:

The screenshot shows the Apache Superset SQL Lab interface. On the left, the 'DATABASE' dropdown is set to 'Impala' and 'Apache Impala'. The 'SCHEMA' dropdown is set to 'default'. The 'SEE TABLE SCHEMA' dropdown is set to 'Select table or type table name'. The main area shows a SQL query:

```
1 SELECT tpep_pickup_datetime AS PICKUP_TIME, tpep_dropoff_datetime AS DROPOFF_TIME, passenger_count AS PASSENGERS
2 FROM taxi
3 WHERE passenger_count > 3 AND trip_distance > 2
4 LIMIT 100
```

Below the query, there are buttons for 'RUN', 'LIMIT: 1000', 'SAVE', and 'COPY LINK'. The 'RESULTS' tab is active, showing a table with 100 rows returned. The table has three columns: 'pickup_time', 'dropoff_time', and 'passengers'.

pickup_time	dropoff_time	passengers
2022-02-11T20:37:30	2022-02-11T21:02:03	4
2022-02-11T20:45:51	2022-02-11T21:03:09	5
2022-02-11T20:02:00	2022-02-11T20:20:14	5
2022-02-11T20:01:48	2022-02-11T20:18:07	4
2022-02-11T20:37:27	2022-02-11T20:57:25	6
2022-02-11T20:18:42	2022-02-11T20:48:12	4
2022-02-11T20:59:08	2022-02-11T21:20:47	6
2022-02-11T20:08:09	2022-02-11T20:19:54	4
2022-02-11T20:00:53	2022-02-11T20:27:08	5
2022-02-11T20:03:14	2022-02-11T20:14:04	5
2022-02-11T20:31:03	2022-02-11T20:47:30	5
2022-02-11T20:02:00	2022-02-11T20:17:44	4
2022-02-11T20:28:46	2022-02-11T20:39:24	6

Lehenengo adibide honetan, bidaia hasiera eta bukaeraren denborak bueltatzen ditugu, bidaiari kopuruarekin batera, honako baldintza betetzen badute: 3 bidaiari baino gehiago izan dira eta 2km baino gehiagoko bidaia egin dute.

Superset Dashboards Charts Datasets SQL

Untitled Query 1 x Query undefined x

DATABASE: impala Apache Impala

SCHEMA: default

SEE TABLE SCHEMA: Select table or type table name

```

1 SELECT passenger_count, MAX(tip_amount)
2 FROM Taxi
3 GROUP BY passenger_count
4 LIMIT 100

```

RUN LIMIT: 1 000 00:00:55 SAVE COPY LINK

RESULTS QUERY HISTORY

CREATE CHART DOWNLOAD TO CSV COPY TO CLIPBOARD Filter results

11 rows returned

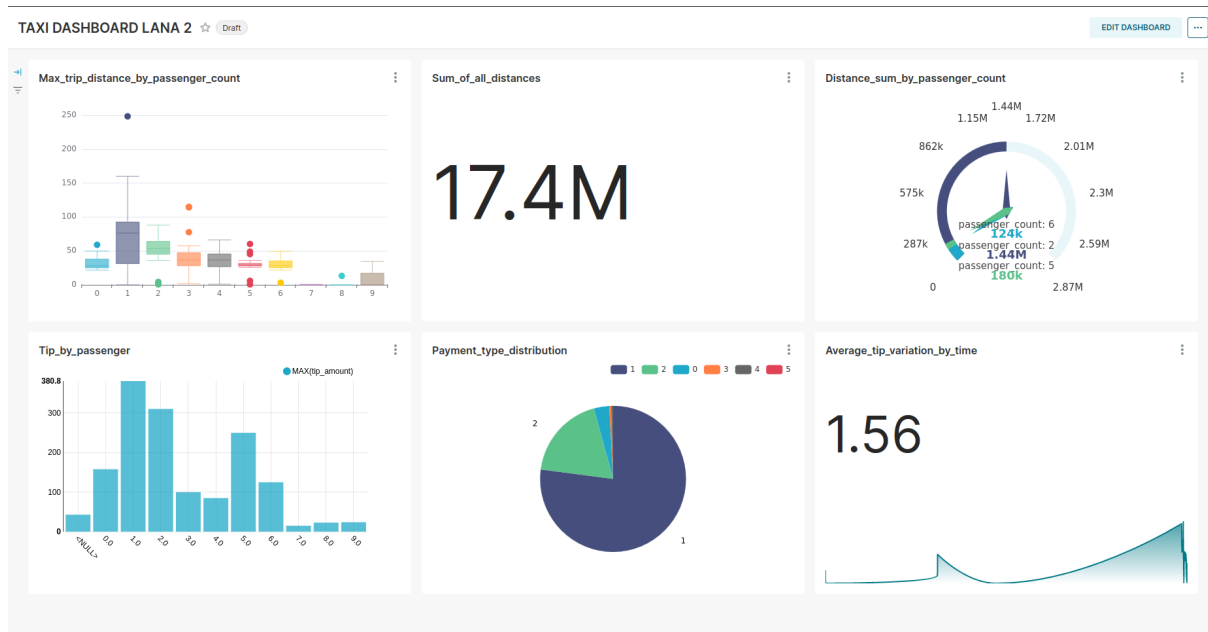
passenger_count	max(tip_amount)
6	125
2	310.32000732421875
5	250
0	158
9	24.059999465942383
8	23.010000228881836
1	380.79998779296875
NULL	43.11000061035156
4	85.0199966430664
3	100
7	15.15999884741211

Beste adibide honetan, propina kopuru handienaren bila gaude, bidaiari kopuru bakoitzerako. Bertan, egindako propina handiena 380 dolarrekoa dela ikusi daiteke, eta pertsona bakar bateko bidaia baten egin zela jakin dezakegu.

4.6. Dashboard eta Chart-ak

Enuntziatuan eskatzen zen bezala, .parquet fitxategiko datuekin grafiko desberdinak egin ditugu Superset erabiliz. Gure lanerako gehien erabili ditugun kanpoak honakoak izan dira: trip_distance (bidaiaren distantzia), passenger_count (bidaiari kopurua), tip_amount (propina kopurua) eta payment_type (ordainketa mota).

Hona hemen nola geratu den gure Dashboard-a:



Dashboardeko chart desberdinak laburki azalduko ditugu orain:

- **Max_trip_distance_by_passenger_count (goian eskubian):** Bidaiaren distantzia maximoak gordetzen dituen box-plot bat da, bidaiari kopuru desberdinekin banatuta dagoena.
- **Sum_of_all_distances (goian erdian):** Zenbaki bat da, kasu honetan, taxi databasean dugun distantzia guztien batura adierazten duena. SQL-n `SUM(trip_distance)` egitearen baliokidea izango zen.
- **Distance_sum_by_passenger_count (goian ezkerrean):** Kasu honetan, distantziaren batura adierazten duen grafiko bat dugu. Bidaiari kopuruarekiko banatuta dago eta 2, 5 eta 6 bidaiariko bidaien distantziak agertzen dira soilik.
- **Tip_by_passenger (behean eskubian):** Berriz ere bidaiari kopuruarekiko banatuko ditugu balioak bar-plot honetan. Y ardatzak adieraziko duena propina kopuru maximoa izango da.
- **Payment_type_distribution (behean erdian):** Ordainketa mota bakoitza zenbat aldiz egin den adierazten duen pie-plot bat da. SQL dei batean adierazi beharko bagenu, `COUNT(payment_type)` izango zen.
- **Average_tip_variation_by_time (behean ezkerrean):** Propinaren aldaketa denborarekiko adierazten duen grafikoa. Bertan agertzen den zenbakia propinaren batazbesteko orokorra da.

5. Ondorioak

Lan honetan, big data nola atzitu, gorde eta bertako informazioari balioa ematea ikasi dugu. Software libreko aplikazioak erabiliz, milioika datu dituzten datubaseekin lan egitea ere posible dela ikusi dugu, baita aplikazio hauen potentzia nolako den aztertu ere.

Aipatzekoa da, compose.yml fitxategietatik bolumenak kenduta ditugunez, gure proiektuan docker up egiten den bakoitzean, Superset-er taxi taula igo behar dela, hauek ez baitira gordetzen inon. Hau dela eta, “down” egin eta berriro “up” egiterakoan errore bat egon daiteke lehenagotik sartu ditugun taulak erabiltzerakoan, hauek ez baitira existitzen.

6. Erreferentziak

- Gure proiektuaren github biltegia:
https://github.com/oierIM/DMPA_kudu-impala-superset
- Taxi taularen kopia impala-shell-en:
<https://stackoverflow.com/questions/26636476/how-to-duplicate-cloud-impala-table-with-impala-shell-or-other-means>
- Hainbat docker up batera egin:
<https://learn.microsoft.com/en-us/dotnet/architecture/microservices/multi-container-microservice-net-applications/multi-container-applications-docker-compose>
- Taxi datubasearen informazioa:
https://github.com/anurag3290/NYC-Taxi-Hive/blob/master/data_dictionary_trip_records_yellow.pdf
- Sareak erabili compose-an:
<https://docs.docker.com/compose/networking/>