Міністерство освіти і науки України

Національний технічний університет України

“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Лабораторна робота №1

Обробка надвеликих масивів даних

Тема: Розподілена обробка даних в Apache Hadoop та Apache Hive

Виконав
студент групи ІП-11:
Панченко С. В.

Перевірив:
Смілянець Ф. А.

Київ 2025

ЗМІСТ

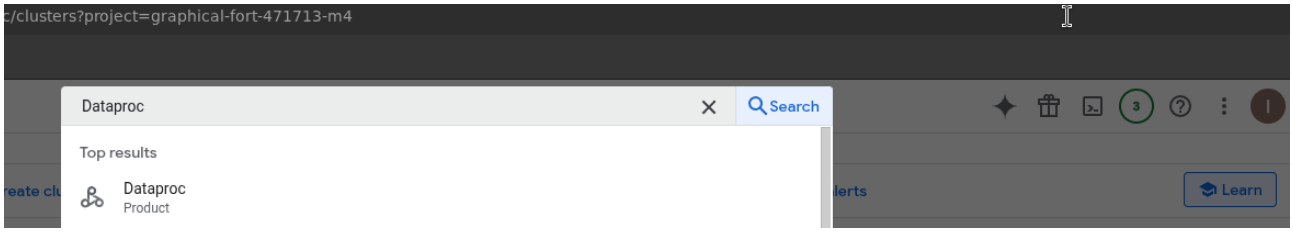
1 МЕТА

Відпрацювати повний цикл підготовки Big Data-проєкту: налаштувати компоненти Hadoop/Spark/Hive, реалізувати завантаження даних та ETL-процедури мовами Java/Python/Scala, спроєктувати архітектуру бази даних і підготувати короткий аналітичний звіт про результати обробки.

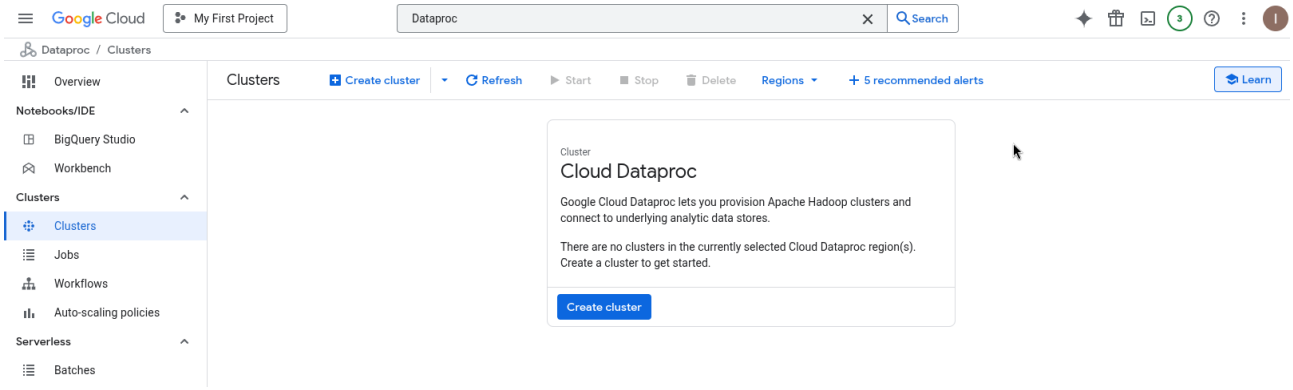
2 ВИКОНАННЯ

2.1 Підготовка даних

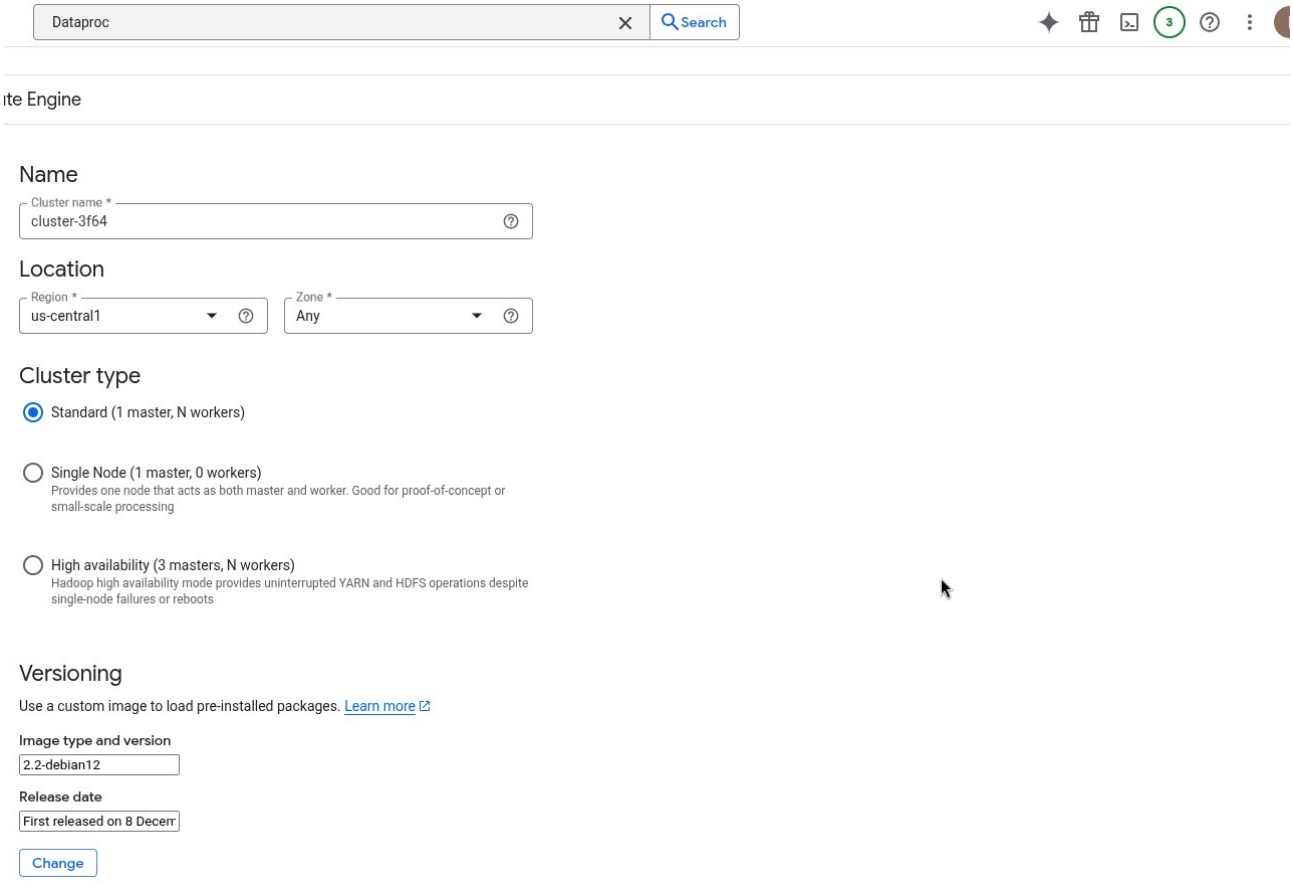
Обираю Dataproc.

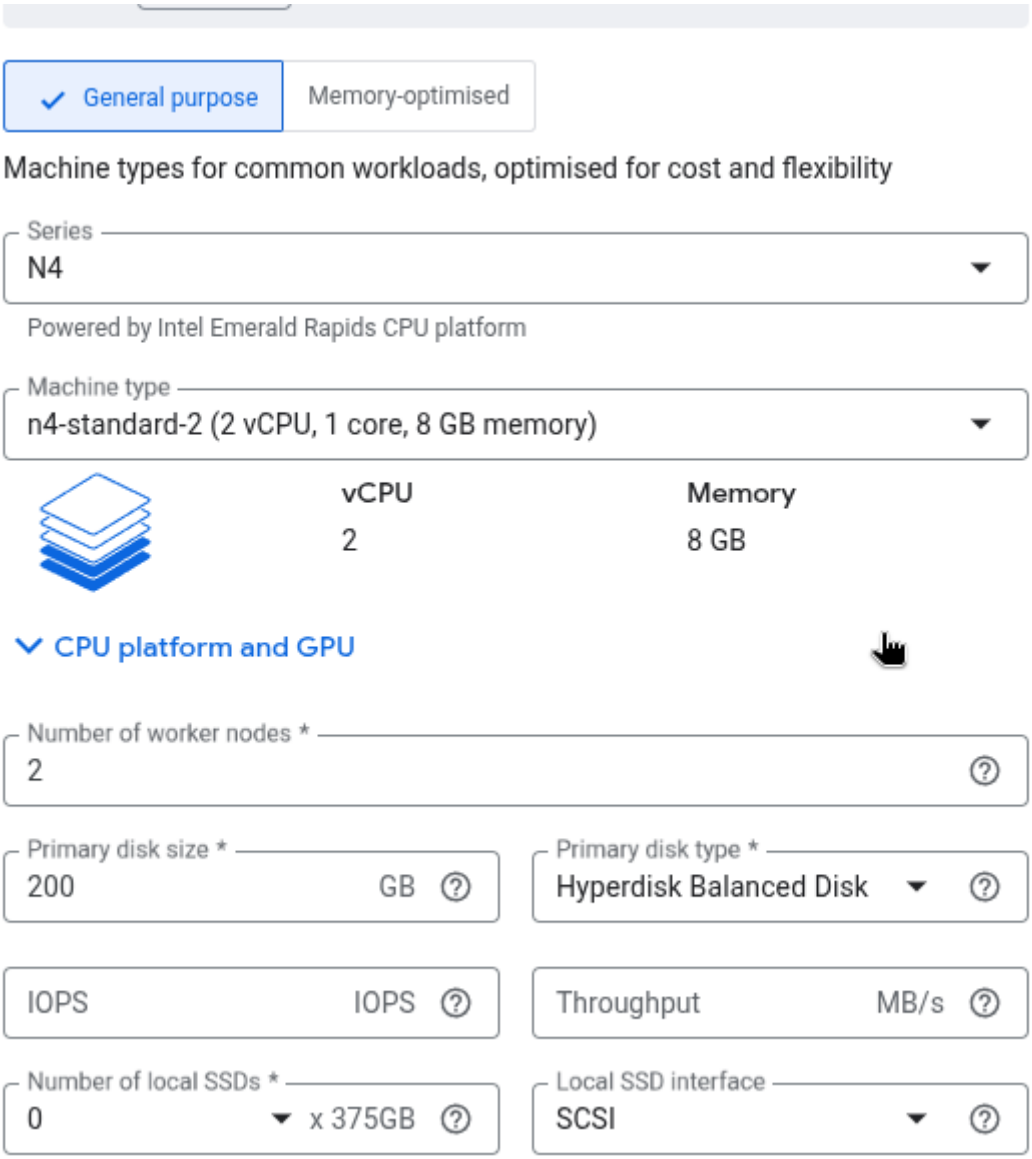


Обираю вкладку кластери.

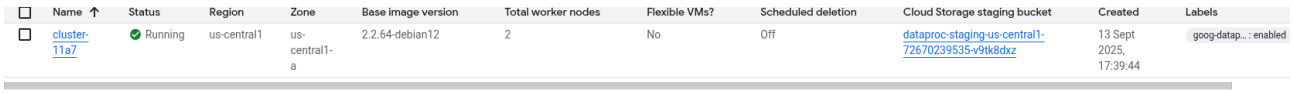


Створюю кластер з основною нодою та двома нодами-worker’ами.

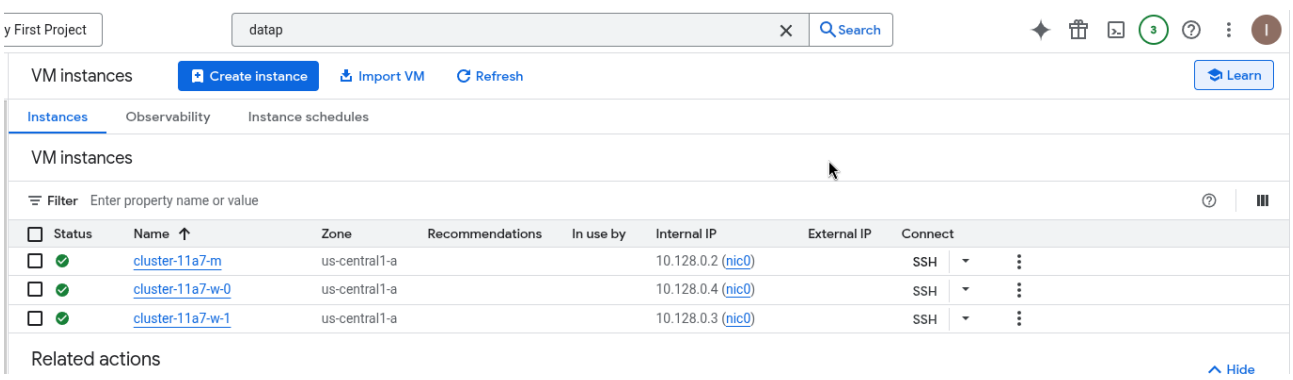




Створений кластер.



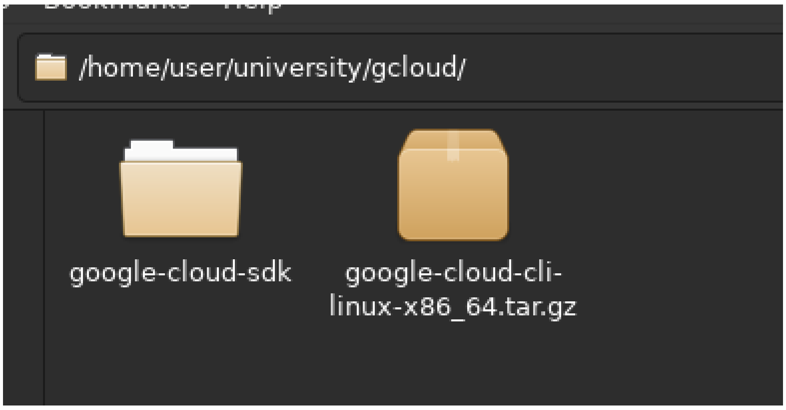
Відкриваємо SSH.



Але для початку встановимо gcloud на мою машину на Archlinux. Завантажимо архів з gcloud.

Platform	Package name	Size	SHA256 Checksum
Linux 64-bit (x86_64)	google-cloud-cli-linux-x86_64.tar.gz	150.2 MB	8ba7e746ca05f225e5a73952bbc03f4086a5f65fd94f3717df6f75f212587159

Розпакуємо його.



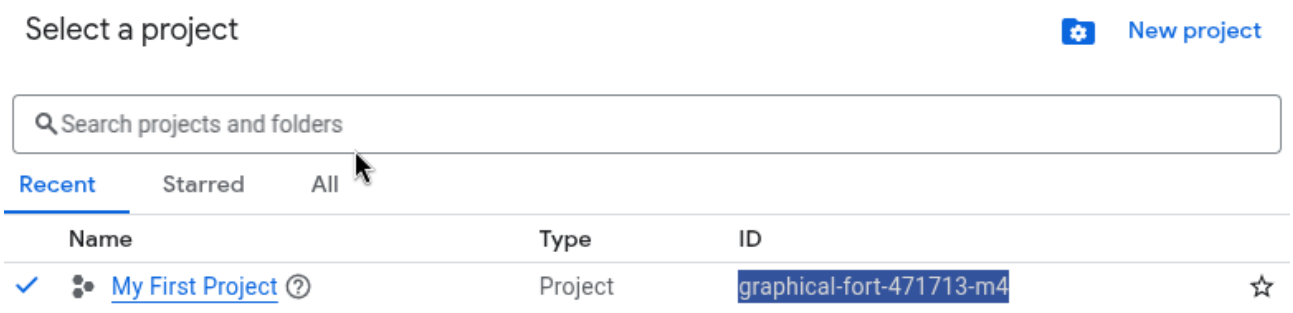
Встановимо gcloud.

```
user@archlinux:~/university/gcloud/google-cloud-sdk$ bash install.sh
Welcome to the Google Cloud CLI!

To help improve the quality of this product, we collect anonymized usage data
and anonymized stacktraces when crashes are encountered; additional information
is available at <https://cloud.google.com/sdk/usage-statistics>. This data is
handled in accordance with our privacy policy
<https://cloud.google.com/terms/cloud-privacy-notice>. You may choose to opt in this
collection now (by choosing 'Y' at the below prompt), or at any time in the
future by running the following command:

    gcloud config set disable_usage_reporting false

Do you want to help improve the Google Cloud CLI (y/N)?  y


Your current Google Cloud CLI version is: 538.0.0
The latest available version is: 538.0.0

┌──────────────────────────────────────────────────────────────────────────────────────────────────────────┐
│                                                Components                                                │
├───────────────┬────────────────────────────────────────────────────┬────────────────────────────┬──────────┤
│     Status    │                        Name                        │             ID             │   Size   │
├───────────────┼────────────────────────────────────────────────────┼────────────────────────────┼──────────┤
│ Not Installed │ App Engine Go Extensions                           │ app-engine-go              │  4.7 MiB │
│ Not Installed │ Artifact Registry Go Module Package Helper         │ package-go-module          │  < 1 MiB │
│ Not Installed │ Cloud Bigtable Command Line Tool                   │ cbt                        │ 20.5 MiB │
│ Not Installed │ Cloud Bigtable Emulator                            │ bigtable                   │  8.5 MiB │
│ Not Installed │ Cloud Datastore Emulator                           │ cloud-datastore-emulator   │ 36.2 MiB │
│ Not Installed │ Cloud Firestore Emulator                           │ cloud-firestore-emulator   │ 53.6 MiB │
│ Not Installed │ Cloud Pub/Sub Emulator                             │ pubsub-emulator            │ 49.8 MiB │
│ Not Installed │ Cloud Run Proxy                                    │ cloud-run-proxy            │ 13.3 MiB │
│ Not Installed │ Cloud SQL Proxy v2                                 │ cloud-sql-proxy            │ 15.7 MiB │
│ Not Installed │ Cloud Spanner Emulator                             │ cloud-spanner-emulator     │ 37.7 MiB │
│ Not Installed │ Google Container Registry's Docker credential helper │ docker-credential-gcr    │  1.8 MiB │
│ Not Installed │ Kustomize                                          │ kustomize                  │  4.3 MiB │
```

Залогінимося в gcloud.

```
user@archlinux:~/university/gcloud/google-cloud-sdk$ gcloud auth login
Your browser has been opened to visit:
```

Визначимо ID поточного проєкту та збережемо його в конфігу gcloud.




```
user@archlinux:~/university/gcloud/google-cloud-sdk$ gcloud config set project graphical-fort-471713-m4
Updated property [core/project].
```

Перекинемо файли FOP.zip та UO.zip на віртуальну машину.

```
user@archlinux:~/university/masters_first_semester_bigdata/lab_1$ gcloud compute scp FOP.zip user@cluster-11a7-m:~/FOP.zip
No zone specified. Using zone [us-central1-a] for instance: [cluster-11a7-m].
External IP address was not found; defaulting to using IAP tunneling.
WARNING:

To increase the performance of the tunnel, consider installing NumPy. For instructions,
please see https://cloud.google.com/iap/docs/using-tcp-forwarding#increasing_the_tcp_upload_bandwidth

FOP.zip                                                                                       100%  245MB   4.4MB/s   00:55
user@archlinux:~/university/masters_first_semester_bigdata/lab_1$ gcloud compute scp UO.zip user@cluster-11a7-m:~/UO.zip
No zone specified. Using zone [us-central1-a] for instance: [cluster-11a7-m].
External IP address was not found; defaulting to using IAP tunneling.
WARNING:

To increase the performance of the tunnel, consider installing NumPy. For instructions,
please see https://cloud.google.com/iap/docs/using-tcp-forwarding#increasing_the_tcp_upload_bandwidth

UO.zip                                                                                        100%  192MB   4.4MB/s   00:43
user@archlinux:~/university/masters_first_semester_bigdata/lab_1$
```

Під’єднаємося до віртульної машини.

```
user@archlinux:~/university/masters_first_semester_bigdata/lab_1$ gcloud compute ssh cluster-11a7-m
No zone specified. Using zone [us-central1-a] for instance: [cluster-11a7-m].
External IP address was not found; defaulting to using IAP tunneling.
WARNING:

To increase the performance of the tunnel, consider installing NumPy. For instructions,
please see https://cloud.google.com/iap/docs/using-tcp-forwarding#increasing_the_tcp_upload_bandwidth

Linux cluster-11a7-m 6.1.0-38-cloud-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.147-1 (2025-08-02) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sat Sep 13 15:14:37 2025 from 35.235.241.17
user@cluster-11a7-m:~$
```

Створимо директорії в hadoop.

```
user@cluster-11a7-m:~$ hadoop fs -mkdir /tables_data
hadoop fs -mkdir /tables_data/UO
hadoop fs -mkdir /tables_data/FOP
```

Розпакуємо zip-файли даних та завантажимо їх в hadoop.

```
user@cluster-11a7-m:~$ unzip FOP.zip
Archive:  FOP.zip
  inflating: FOP.csv
user@cluster-11a7-m:~$ unzip UO.zip
Archive:  UO.zip
  inflating: UO.csv
user@cluster-11a7-m:~$ hadoop fs -put ./UO.csv /tables_data/UO/
hadoop fs -put ./FOP.csv /tables_data/FOP/
user@cluster-11a7-m:~$
```

Відкриємо Apache Hive.

```
user@cluster-11a7-m:~$ hive
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/tez/lib/slf4j-reload4j-1.7.36.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/hadoop/lib/slf4j-reload4j-1.7.36.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Reload4jLoggerFactory]
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/tez/lib/slf4j-reload4j-1.7.36.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/hadoop/lib/slf4j-reload4j-1.7.36.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Reload4jLoggerFactory]
Hive Session ID = c2e09510-c595-4527-87e0-57a2ae15ca07

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.properties Async: true
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.hadoop.hive.common.StringInternUtils (file:/usr/lib/hive/lib/hive-common-3.1.3.jar) to field java.net.URI.string
WARNING: Please consider reporting this to the maintainers of org.apache.hadoop.hive.common.StringInternUtils
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
Hive Session ID = 50fc9d72-ec1f-4ae5-b2b5-2507443454e1
hive>
```

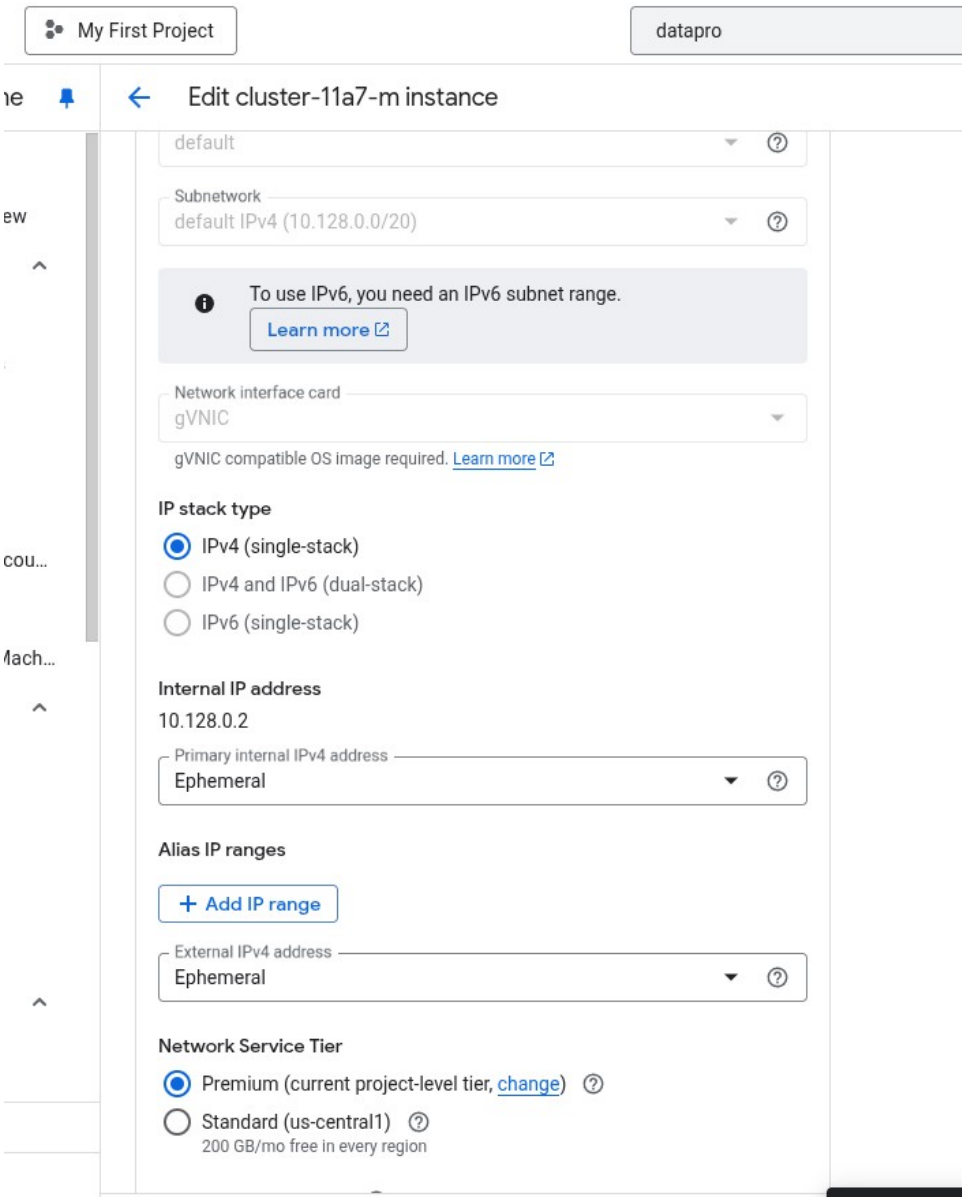
Створимо таблицю для UO.

```
hive> create external table UOtable(name string,EDRPOU string,ADDRESS string,BOSS string,founders string,fio string,KVED string,stan string) ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.O
penCSVSerde' STORED AS TEXTFILE LOCATION '/tables_data/UO/';
OK
Time taken: 1.205 seconds
hive>
```

Створимо таблицю для FOP.

```
hive> create external table FOP_table(fio string,address string,kved string,stan string) ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde' STORED AS TEXTFILE LOCATION '/tables_da
ta/FOP/';
OK
Time taken: 0.545 seconds
```

Встановимо для віртуальної машини зовнішній ефемерний ip для того щоб мати доступ до репозиторіїв apt та встановити postgresql.



Встановимо postgresql.

```
user@cluster-11a7-m:~$ sudo apt install postgresql
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libc-bin libc-dev-bin libc-devtools libc-l10n libc6 libc6-dbg libc6-dev libllvm14 libpq5 libxslt1.1 locales po
  ssl-cert sysstat
Suggested packages:
  glibc-doc libnss-nis libnss-nisplus postgresql-doc postgresql-doc-15 isag
The following NEW packages will be installed:
  libc-l10n libllvm14 libxslt1.1 locales postgresql postgresql-15 postgresql-client-15 postgresql-client-common
The following packages will be upgraded:
  libc-bin libc-dev-bin libc-devtools libc6 libc6-dbg libc6-dev libpq5
7 upgraded, 11 newly installed, 0 to remove and 77 not upgraded.
Need to get 59.1 MB of archives.
After this operation, 197 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 file:/etc/apt/mirrors/debian.list Mirrorlist [30 B]
Get:2 https://deb.debian.org/debian bookworm/main amd64 libc6-dbg amd64 2.36-9+deb12u13 [7375 kB]
Get:3 https://deb.debian.org/debian bookworm/main amd64 libc-devtools amd64 2.36-9+deb12u13 [55.0 kB]
Get:4 https://deb.debian.org/debian bookworm/main amd64 libc6-dev amd64 2.36-9+deb12u13 [1904 kB]
Get:5 https://deb.debian.org/debian bookworm/main amd64 libc-dev-bin amd64 2.36-9+deb12u13 [47.4 kB]
Get:6 https://deb.debian.org/debian bookworm/main amd64 libc6 amd64 2.36-9+deb12u13 [2758 kB]
Get:7 https://deb.debian.org/debian bookworm/main amd64 libc-bin amd64 2.36-9+deb12u13 [609 kB]
```

Підключимося до postgresql від sudo та створимо користувача user та базу даних.

```
user@cluster-11a7-m:~$ sudo -u postgres psql
psql (15.14 (Debian 15.14-0+deb12u1))
Type "help" for help.

postgres=#
```

```
postgres=# CREATE USER "user" WITH CREATEDB;
CREATE ROLE
postgres=# CREATE DATABASE mydb;
CREATE DATABASE
postgres=#
```

```
postgres=# ALTER USER "user" WITH PASSWORD '1111';
ALTER ROLE
postgres=#
```

Створимо дві таблиці UO_table та FOP_table.

```
mydb=> CREATE TABLE UO_table (
    name TEXT,
    EDRPOU TEXT,
    ADDRESS TEXT,
    BOSS TEXT,
    founders TEXT,
    fio TEXT,
    KVED TEXT,
    stan TEXT
);
CREATE TABLE
mydb=> CREATE TABLE FOP_table (
    fio TEXT,
    address TEXT,
    kved TEXT,
    stan TEXT
);
CREATE TABLE
```

UO.csv має JSON рядки, і в них ліпки не правильно заескейплині, тому заекспортуємо hive таблицю у csv.

```
hive> INSERT OVERWRITE LOCAL DIRECTORY '/home/user/hive_output'
    > ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
    > WITH SERDEPROPERTIES (
    >    "separatorChar" = ",",
    >    "quoteChar"     = "\"",
    >    "escapeChar"    = "\\"
    > )
    > STORED AS TEXTFILE
    > SELECT * FROM uotable;
Query ID = user_20250913174234_80ce4c61-5d06-41d6-bdff-341d4af34578
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1757774494954_0006)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 ..... container     SUCCEEDED      9          9        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 01/01  [==========================>>] 100%  ELAPSED TIME: 32.51 s
--------------------------------------------------------------------------------
Moving data to local directory /home/user/hive_output
OK
Time taken: 456.621 seconds
```

hive> INSERT OVERWRITE LOCAL DIRECTORY '/home/user/hive_output'

> ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'

> WITH SERDEPROPERTIES (

> "separatorChar" = ",",

> "quoteChar" = "\"",

> "escapeChar" = "\\"

>)

> STORED AS TEXTFILE

> SELECT * FROM uotable;

Об’єднаємо результат експорту таблиці в один файл.

```
user@cluster-11a7-m:~$ less hive_output/000001_0
user@cluster-11a7-m:~$ cat hive_output/* > hive_output.csv
user@cluster-11a7-m:~$ less hive_output.csv
```

Далі заімпортуємо цю таблиці в PostgreSQL.

```
mydb=> \copy uo_table (name, edrpou, address, boss, founders, fio, kved, stan)
FROM '/home/user/hive_output.csv'
WITH (
    FORMAT csv,
    DELIMITER ',',
    QUOTE '"',
    ESCAPE '\',
    HEADER false
);
COPY 1659657
```

2.2 Завдання 1.1

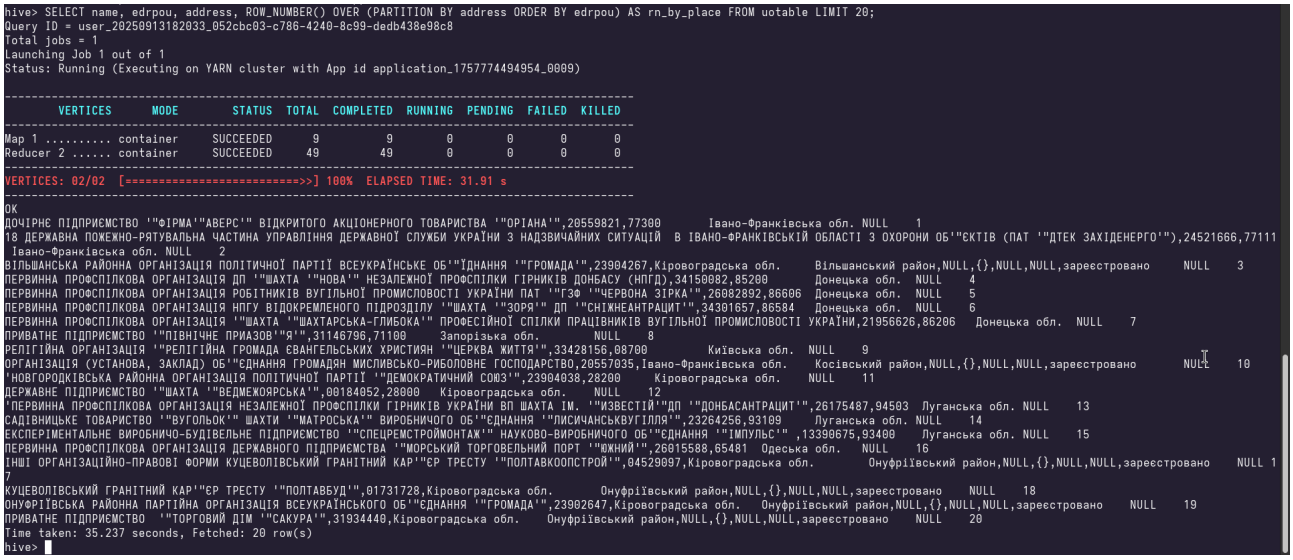
Визначимо кількість рядків та підрахуємо час виконання в hive та postgresql.

```
hive> select count(*) from uotable;
Query ID = user_20250913181308_1f69df39-224e-42bf-972d-23fe4f104a77
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1757774494954_0008)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      9          9        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 21.90 s
--------------------------------------------------------------------------------
OK
1659657
Time taken: 25.82 seconds, Fetched: 1 row(s)
hive>
```

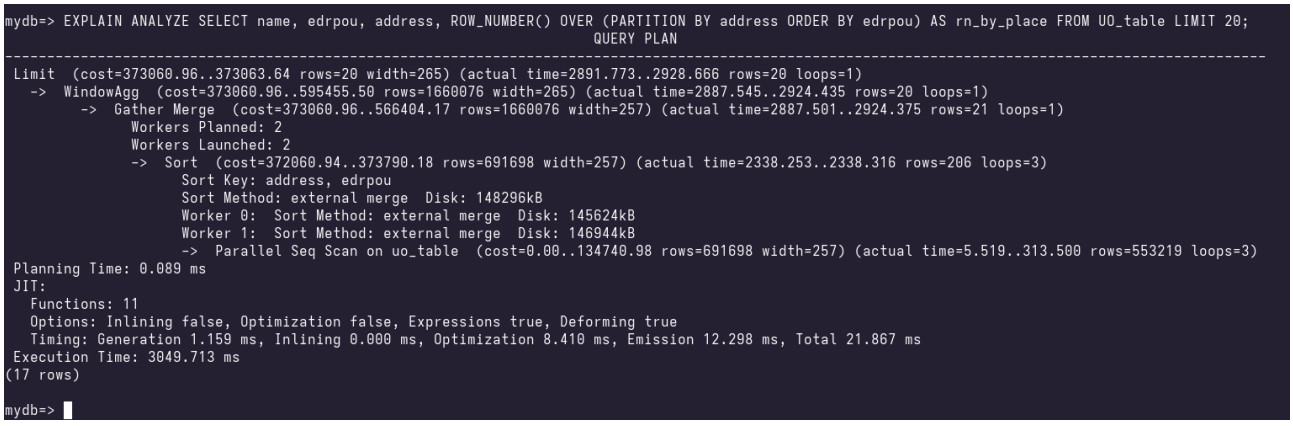
```
mydb=> EXPLAIN ANALYZE SELECT COUNT(*) FROM UO_table;
                                                                  QUERY PLAN
-----------------------------------------------------------------------------------------------------------------------------------------------
 Finalize Aggregate  (cost=137470.44..137470.45 rows=1 width=8) (actual time=350.365..355.825 rows=1 loops=1)
   ->  Gather  (cost=137470.23..137470.44 rows=2 width=8) (actual time=350.329..355.792 rows=3 loops=1)
         Workers Planned: 2
         Workers Launched: 2
         ->  Partial Aggregate  (cost=136470.23..136470.24 rows=1 width=8) (actual time=317.023..317.025 rows=1 loops=3)
               ->  Parallel Seq Scan on uo_table  (cost=0.00..134740.98 rows=691698 width=0) (actual time=0.031..250.194 rows=553219 loops=3)
 Planning Time: 0.058 ms
 JIT:
   Functions: 8
   Options: Inlining false, Optimization false, Expressions true, Deforming true
   Timing: Generation 0.615 ms, Inlining 0.000 ms, Optimization 1.112 ms, Emission 13.151 ms, Total 14.877 ms
 Execution Time: 356.122 ms
(12 rows)

mydb=>
```

2.3 Завдання 1.2

```
SELECT name, edrpou, address, ROW_NUMBER() OVER
(PARTITION BY address ORDER BY edrpou) AS rn_by_place
FROM uotable LIMIT 20;
```




```
EXPLAIN ANALYZE SELECT name, edrpou, address,
ROW_NUMBER() OVER (PARTITION BY address ORDER BY edrpou)
AS rn_by_place FROM UO_table LIMIT 20;
```



2.4 Завдання 1.3

```
EXPLAIN ANALYZE SELECT * FROM UO_table uo join FOP_table
fop on uo.address = fop.address join FOP_table fop1 on
uo.address = fop1.address;
```

```
mydb=> EXPLAIN ANALYZE SELECT * FROM UO_table uo join FOP_table fop on uo.address = fop.address join FOP_table fop1 on uo.address = fop1.address;
                                                                             QUERY PLAN
--------------------------------------------------------------------------------------------------------------------------------------------------------
 Gather  (cost=732420.59..4536304.41 rows=14573054 width=1276) (actual time=65118.282..136066.147 rows=283585188 loops=1)
   Workers Planned: 2
   Workers Launched: 2
   ->  Parallel Hash Join  (cost=731420.59..3077999.00 rows=6072106 width=1276) (actual time=65054.527..87665.205 rows=94528396 loops=3)
         Hash Cond: (uo.address = fop1.address)
         ->  Parallel Hash Join  (cost=365710.30..1060597.69 rows=2049406 width=937) (actual time=27895.150..46990.671 rows=529939 loops=3)
               Hash Cond: (uo.address = fop.address)
               ->  Parallel Seq Scan on uo_table uo  (cost=0.00..134740.98 rows=691698 width=598) (actual time=0.625..8969.723 rows=553219 loops=3)
               ->  Parallel Hash  (cost=248448.02..248448.02 rows=2042102 width=339) (actual time=18082.463..18082.465 rows=1633268 loops=3)
                     Buckets: 32768  Batches: 256  Memory Usage: 7776kB
                     ->  Parallel Seq Scan on fop_table fop  (cost=0.00..248448.02 rows=2042102 width=339) (actual time=88.732..16829.727 rows=1633268 loops=3)
         ->  Parallel Hash  (cost=248448.02..248448.02 rows=2042102 width=339) (actual time=17272.215..17272.216 rows=1633268 loops=3)
               Buckets: 32768  Batches: 256  Memory Usage: 7776kB
               ->  Parallel Seq Scan on fop_table fop1  (cost=0.00..248448.02 rows=2042102 width=339) (actual time=1.056..15486.627 rows=1633268 loops=3)
 Planning Time: 15.516 ms
 JIT:
   Functions: 51
   Options: Inlining true, Optimization true, Expressions true, Deforming true
   Timing: Generation 6.293 ms, Inlining 389.395 ms, Optimization 734.230 ms, Emission 431.705 ms, Total 1561.623 ms
 Execution Time: 151065.981 ms
(20 rows)

mydb=>
```

PostgreSQL запит виконався лише з 8-го разу, після того як почистив оперативу та повідключав сторонні сервіси на VM.

На hive запит зайняв багато часу більше 1000 секунд.

```
hive> SELECT * FROM uotable uo join fop_table fop on uo.address = fop.address join fop_table fop1 on uo.address = fop1.address;
No Stats for default@uotable, Columns: edrpou, address, boss, name, kved, stan, founders, fio
No Stats for default@fop_table, Columns: address, kved, stan, fio
No Stats for default@fop_table, Columns: address, kved, stan, fio
Query ID = user_20250913185314_b5b496d7-e50b-4616-a473-f0106fd64180
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1757774494954_0010)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      6          6        0        0       0       0
Map 4 .......... container     SUCCEEDED      9          9        0        0       0       0
Reducer 2 ...... container     SUCCEEDED    137        137        0        0       0       0
Reducer 3 ..     container       RUNNING    137         54        2       81       3       0
----------------------------------------------------------------------------------------------
VERTICES: 03/04  [=================>>--------] 71%   ELAPSED TIME: 1007.99 s
----------------------------------------------------------------------------------------------
Status: Submitted
Interrupting... Be patient, this might take some time.
Press Ctrl+C again to kill JVM
Trying to shutdown DAG
Exiting the JVM
Trying to shutdown DAG
user@cluster-11a7-m:~$
```

Спробуємо виконати запит без останнього джоіна.

```
EXPLAIN ANALYZE SELECT * FROM uotable uo join fop_table
fop on uo.address = fop.address;
```

```
hive> EXPLAIN ANALYZE SELECT * FROM uotable uo join fop_table fop on uo.address = fop.address;
Query ID = user_20250913194036_99708c5e-1f42-464b-ab03-81f7c35284d8
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1757774494954_0013)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      9          9        0        0       0       0
Map 3 .......... container     SUCCEEDED      6          6        0        0       0       0
Reducer 2 ...... container     SUCCEEDED    114        114        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 94.28 s
--------------------------------------------------------------------------------
OK
OK
Plan optimized by CBO.

Vertex dependency in root stage
Reducer 2 <- Map 1 (SIMPLE_EDGE), Map 3 (SIMPLE_EDGE)

Stage-0
  Fetch Operator
    limit:-1
    Stage-1
      Reducer 2
      File Output Operator [FS_10]
        Merge Join Operator [MERGEJOIN_15] (rows=31344672/1589816 width=584)
          Conds:RS_18._col2=RS_21._col1(Inner),Output:["_col0","_col1","_col2","_col3","_col4","_col5","_col6","_col7","_col8","_col9","_col10","_col11"]
        <-Map 1 [SIMPLE_EDGE] vectorized
          SHUFFLE [RS_18]
            PartitionCols:_col2
            Select Operator [SEL_17] (rows=12585021/1659617 width=984)
              Output:["_col0","_col1","_col2","_col3","_col4","_col5","_col6","_col7"]
              Filter Operator [FIL_16] (rows=12585021/1659617 width=984)
                predicate:address is not null
                TableScan [TS_0] (rows=12585021/1659657 width=984)
                  default@uotable,uo,Tbl:COMPLETE,Col:NONE,Output:["name","edrpou","address","boss","founders","fio","kved","stan"]
        <-Map 3 [SIMPLE_EDGE] vectorized
          SHUFFLE [RS_21]
            PartitionCols:_col1
            Select Operator [SEL_20] (rows=28495156/4899804 width=584)
              Output:["_col0","_col1","_col2","_col3"]
              Filter Operator [FIL_19] (rows=28495156/4899804 width=584)
                predicate:address is not null
                TableScan [TS_3] (rows=28495156/4899804 width=584)
                  default@fop_table,fop,Tbl:COMPLETE,Col:NONE,Output:["fio","address","kved","stan"]

Time taken: 397.997 seconds, Fetched: 32 row(s)
```

2.5 Завдання 1.4

Завантажимо Lending Club Loans_synthetic1.csv на віртуальну машину.

```
gcloud compute scp Lending\ Club\ Loans_synthetic1.csv
user@cluster-11a7-m:~/lending.csv
```

```
user@archlinux:~/university/masters_first_semester_bigdata/lab_1$ gcloud compute scp Lending\ Club\ Loans_synthetic1.csv user@cluster-11a7-m:~/lending.csv
No zone specified. Using zone [us-central1-a] for instance: [cluster-11a7-m].
Lending Club Loans_synthetic1.csv                                                                                    100%  154MB  10.4MB/s   00:14
user@archlinux:~/university/masters_first_semester_bigdata/lab_1$ gcloud compute ssh cluster-11a7-m
No zone specified. Using zone [us-central1-a] for instance: [cluster-11a7-m].
Linux cluster-11a7-m 6.1.0-38-cloud-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.147-1 (2025-08-02) x86_64

5 updates could not be installed automatically. For more details,
see /var/log/unattended-upgrades/unattended-upgrades.log

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sat Sep 13 16:22:07 2025 from 35.235.241.17
user@cluster-11a7-m:~$ ls
lending.csv
user@cluster-11a7-m:~$
```

Завантажимо дані у hive.

```
hdfs dfs -mkdir -p /data/lending_club
hdfs dfs -put -f "Lending Club Loans_synthetic1.csv"
/data/lending_club/
```

```
user@cluster-11a7-m:~$ hdfs dfs -mkdir -p /data/lending_club
user@cluster-11a7-m:~$ hdfs dfs -put -f lending.csv /data/lending_club/
```

Створимо зовнішню таблицю для початкового завантаження даних.

```
CREATE EXTERNAL TABLE lending_club_raw (
    loan_amount           INT,
    payments_term         STRING,
    monthly_payment       DOUBLE,
    grade                 INT,
    working_years         INT,
    home                  STRING,
    annual_income         DOUBLE,
    verification          STRING,
    purpose               STRING,
    debt_to_income        DOUBLE,
    delinquency           INT,
    inquiries             INT,
    open_credit_lines     INT,
    derogatory_records    INT,
    revolving_balance     INT,
    revolving_rate        DOUBLE,
    total_accounts        INT,
    bankruptcies          INT,
    fico_average          INT,
    loan_risk             STRING
)
ROW FORMAT SERDE
'org.apache.hadoop.hive.serde2.OpenCSVSerde'
WITH SERDEPROPERTIES (
    "separatorChar" = ";",
    "quoteChar"     = "\"",
    "escapeChar"    = "\\"
)
STORED AS TEXTFILE
```

```
LOCATION '/data/lending_club'
TBLPROPERTIES ("skip.header.line.count"="1");
```

```
hive> CREATE EXTERNAL TABLE lending_club_raw (
    >     loan_amount           INT,
    >     payments_term         STRING,
    >     monthly_payment       DOUBLE,
    >     grade                 INT,
    >     working_years         INT,
    >     home                  STRING,
    >     annual_income         DOUBLE,
    >     verification          STRING,
    >     purpose               STRING,
    >     debt_to_income        DOUBLE,
    >     delinquency           INT,
    >     inquiries             INT,
    >     open_credit_lines     INT,
    >     derogatory_records    INT,
    >     revolving_balance     INT,
    >     revolving_rate        DOUBLE,
    >     total_accounts        INT,
    >     bankruptcies          INT,
    >     fico_average          INT,
    >     loan_risk             STRING
    > )
    > ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
    > WITH SERDEPROPERTIES (
    >     "separatorChar" = ";",
    >     "quoteChar"     = "\"",
    >     "escapeChar"    = "\\"
    > )
    > STORED AS TEXTFILE
    > LOCATION '/data/lending_club'
    > TBLPROPERTIES ("skip.header.line.count"="1");
OK
Time taken: 0.575 seconds
```

Створимо partitioned таблицю.

```
SET hive.exec.dynamic.partition=true;
SET hive.exec.dynamic.partition.mode=nonstrict;
```

```
hive> SET hive.exec.dynamic.partition=true;
hive> SET hive.exec.dynamic.partition.mode=nonstrict;
```

```
CREATE TABLE lending_club (
    loan_amount           INT,
    payments_term         STRING,
```

```
    monthly_payment       DOUBLE,
    grade                 INT,
    working_years         INT,
    home                  STRING,
    annual_income         DOUBLE,
    purpose               STRING,
    debt_to_income        DOUBLE,
    delinquency           INT,
    inquiries             INT,
    open_credit_lines     INT,
    derogatory_records    INT,
    revolving_balance     INT,
    revolving_rate        DOUBLE,
    total_accounts        INT,
    bankruptcies          INT,
    fico_average          INT,
    loan_risk             STRING
)
PARTITIONED BY (verification STRING)
STORED AS ORC;
```

```
hive> CREATE TABLE lending_club (
    >     loan_amount           INT,
    >     payments_term         STRING,
    >     monthly_payment       DOUBLE,
    >     grade                 INT,
    >     working_years         INT,
    >     home                  STRING,
    >     annual_income         DOUBLE,
    >     purpose               STRING,
    >     debt_to_income        DOUBLE,
    >     delinquency           INT,
    >     inquiries             INT,
    >     open_credit_lines     INT,
    >     derogatory_records    INT,
    >     revolving_balance     INT,
    >     revolving_rate        DOUBLE,
    >     total_accounts        INT,
    >     bankruptcies          INT,
    >     fico_average          INT,
    >     loan_risk             STRING
    > )
    > PARTITIONED BY (verification STRING)
    > STORED AS ORC;
OK
Time taken: 0.094 seconds
hive>
```

```
INSERT OVERWRITE TABLE lending_club PARTITION
(verification)
SELECT
    loan_amount,
    payments_term,
    monthly_payment,
    grade,
    working_years,
    home,
    annual_income,
    purpose,
    debt_to_income,
    delinquency,
    inquiries,
    open_credit_lines,
    derogatory_records,
```

```
    revolving_balance,
    revolving_rate,
    total_accounts,
    bankruptcies,
    fico_average,
    loan_risk,
    verification
FROM lending_club_raw;
```

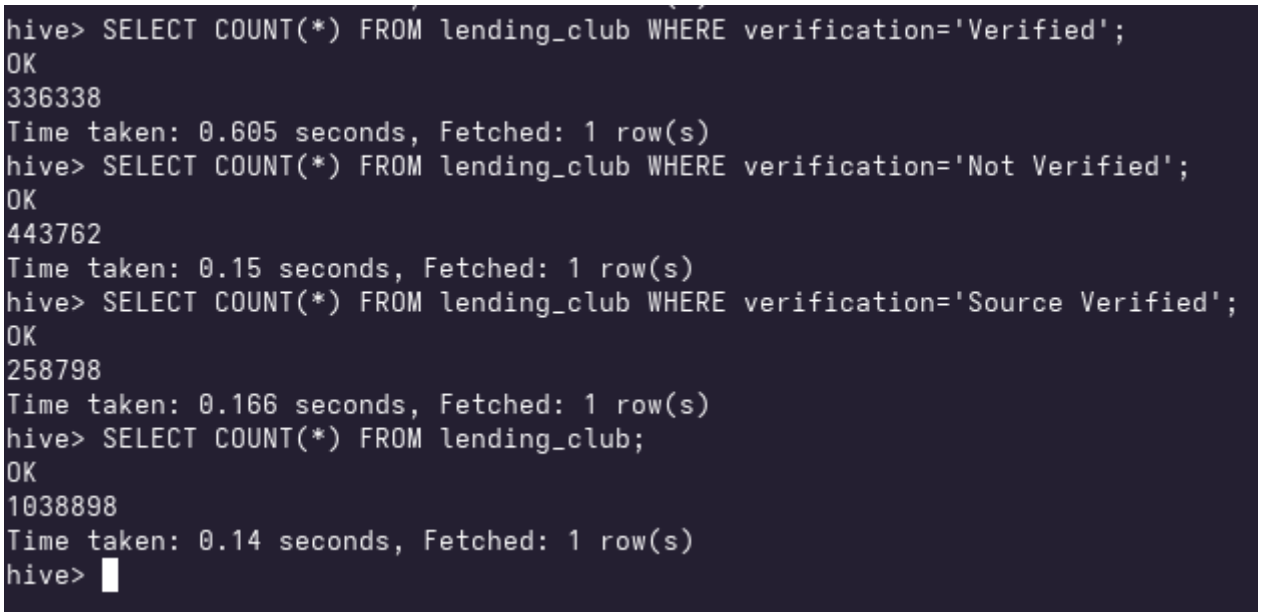
```
hive> INSERT OVERWRITE TABLE lending_club PARTITION (verification)
    > SELECT
    >     loan_amount,
    >     payments_term,
    >     monthly_payment,
    >     grade,
    >     working_years,
    >     home,
    >     annual_income,
    >     purpose,
    >     debt_to_income,
    >     delinquency,
    >     inquiries,
    >     open_credit_lines,
    >     derogatory_records,
    >     revolving_balance,
    >     revolving_rate,
    >     total_accounts,
    >     bankruptcies,
    >     fico_average,
    >     loan_risk,
    >     verification
    > FROM lending_club_raw;
Query ID = user_20250914074243_c0a0af2e-4929-4254-87ce-a2e22da3130d
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1757774494954_0028)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      7          7        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 28.35 s
--------------------------------------------------------------------------------
Loading data to table default.lending_club partition (verification=null)

Loaded : 3/3 partitions.
        Time taken to load dynamic partitions: 0.336 seconds
        Time taken for adding to write entity : 0.002 seconds
OK
Time taken: 37.857 seconds
hive>
```

Покажемо partitions.

```
SHOW PARTITIONS lending_club;
```

Виміряємо час для кожного partition.

```
SELECT COUNT(*) FROM lending_club WHERE
verification='Verified';
SELECT COUNT(*) FROM lending_club WHERE verification='Not
Verified';
SELECT COUNT(*) FROM lending_club WHERE
verification='Source Verified';
SELECT COUNT(*) FROM lending_club;
```



2.6 Завдання 1.5

Встановимо параметри.

```
SET hive.exec.dynamic.partition=true;
```

```
SET hive.exec.dynamic.partition.mode=nonstrict;
SET hive.enforce.bucketing=true;
SET hive.enforce.sorting=true;
```

```
hive> SET hive.exec.dynamic.partition=true;
hive> SET hive.exec.dynamic.partition.mode=nonstrict;
hive> SET hive.enforce.bucketing=true;
hive> SET hive.enforce.sorting=true;
```

Створимо таблицю.

```
CREATE TABLE lending_club_buckets (
    loan_amount           INT,
    payments_term         STRING,
    monthly_payment       DOUBLE,
    grade                 INT,
    working_years         INT,
    home                  STRING,
    annual_income         DOUBLE,
    purpose               STRING,
    debt_to_income        DOUBLE,
    delinquency           INT,
    inquiries             INT,
    open_credit_lines     INT,
    derogatory_records    INT,
    revolving_balance     INT,
    revolving_rate        DOUBLE,
    total_accounts        INT,
    bankruptcies          INT,
    fico_average          INT,
    loan_risk             STRING
)
PARTITIONED BY (verification STRING)
CLUSTERED BY (working_years) INTO 10 BUCKETS
STORED AS ORC
```

```
TBLPROPERTIES(
  "orc.compress"="SNAPPY",
  "bucketing_version"="2"
);
```

```
hive> CREATE TABLE lending_club_buckets (
    >     loan_amount           INT,
    >     payments_term         STRING,
    >     monthly_payment       DOUBLE,
    >     grade                 INT,
    >     working_years         INT,
    >     home                  STRING,
    >     annual_income         DOUBLE,
    >     purpose               STRING,
    >     debt_to_income        DOUBLE,
    >     delinquency           INT,
    >     inquiries             INT,
    >     open_credit_lines     INT,
    >     derogatory_records    INT,
    >     revolving_balance     INT,
    >     revolving_rate        DOUBLE,
    >     total_accounts        INT,
    >     bankruptcies          INT,
    >     fico_average          INT,
    >     loan_risk             STRING
    > )
    > PARTITIONED BY (verification STRING)
    > CLUSTERED BY (working_years) INTO 10 BUCKETS
    > STORED AS ORC
    > TBLPROPERTIES(
    >   "orc.compress"="SNAPPY",
    >   "bucketing_version"="2"
    > );
OK
Time taken: 0.067 seconds
hive>
```

Завантажимо дані.

```
INSERT OVERWRITE TABLE lending_club_buckets PARTITION
(verification)
SELECT
    loan_amount,
    payments_term,
    monthly_payment,
    grade,
    working_years,
```

```
    home,
    annual_income,
    purpose,
    debt_to_income,
    delinquency,
    inquiries,
    open_credit_lines,
    derogatory_records,
    revolving_balance,
    revolving_rate,
    total_accounts,
    bankruptcies,
    fico_average,
    loan_risk,
    verification
FROM lending_club_raw;
```

```
hive> INSERT OVERWRITE TABLE lending_club_buckets PARTITION (verification)
    > SELECT
    >     loan_amount,
    >     payments_term,
    >     monthly_payment,
    >     grade,
    >     working_years,
    >     home,
    >     annual_income,
    >     purpose,
    >     debt_to_income,
    >     delinquency,
    >     inquiries,
    >     open_credit_lines,
    >     derogatory_records,
    >     revolving_balance,
    >     revolving_rate,
    >     total_accounts,
    >     bankruptcies,
    >     fico_average,
    >     loan_risk,
    >     verification
    > FROM lending_club_raw;
Query ID = user_20250914081251_4b7f8731-d229-4997-b459-dccf514989cd
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1757774494954_0029)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
Reducer 2 ...... container     SUCCEEDED     10         10        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      7          7        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 32.51 s
--------------------------------------------------------------------------------
Loading data to table default.lending_club_buckets partition (verification=null)

Loaded : 3/3 partitions.
         Time taken to load dynamic partitions: 0.185 seconds
         Time taken for adding to write entity : 0.001 seconds
OK
Time taken: 41.493 seconds
hive>
```

Переглянемо partitions.

```
SHOW PARTITIONS lending_club_buckets;
```

```
hive> SHOW PARTITIONS lending_club_buckets;
OK
verification=Not Verified
verification=Source Verified
verification=Verified
Time taken: 0.075 seconds, Fetched: 3 row(s)
hive>
```

Переглянемо де зберігаються bucket-файли.

```
DESCRIBE FORMATTED lending_club_buckets;
```

```
hive> DESCRIBE FORMATTED lending_club_buckets;
OK
# col_name              data_type               comment
loan_amount             int
payments_term           string
monthly_payment         double
grade                   int
working_years           int
home                    string
annual_income           double
purpose                 string
debt_to_income          double
delinquency             int
inquiries               int
open_credit_lines       int
derogatory_records      int
revolving_balance       int
revolving_rate          double
total_accounts          int
bankruptcies            int
fico_average            int
loan_risk               string

# Partition Information
# col_name              data_type               comment
verification            string

# Detailed Table Information
Database:               default
OwnerType:              USER
Owner:                  user
CreateTime:             Sun Sep 14 08:09:18 UTC 2025
LastAccessTime:         UNKNOWN
Retention:              0
Location:               hdfs://cluster-11a7-m/user/hive/warehouse/lending_club_buckets
Table Type:             MANAGED_TABLE
Table Parameters:
        COLUMN_STATS_ACCURATE   {\"BASIC_STATS\":\"true\"}
        bucketing_version       2
        numFiles                18
        numPartitions           3
        numRows                 1038898
        orc.compress            SNAPPY
        rawDataSize             466270274
        totalSize               14605125
        transient_lastDdlTime   1757837358

# Storage Information
SerDe Library:          org.apache.hadoop.hive.ql.io.orc.OrcSerde
InputFormat:            org.apache.hadoop.hive.ql.io.orc.OrcInputFormat
OutputFormat:           org.apache.hadoop.hive.ql.io.orc.OrcOutputFormat
Compressed:             No
Num Buckets:            10
Bucket Columns:         [working_years]
Sort Columns:           []
Storage Desc Params:
        serialization.format    1
Time taken: 0.107 seconds, Fetched: 55 row(s)
hive>
```

Бачимо рядок Location:

Location: hdfs://cluster-11a7-m/user/hive/warehouse/lending_club_buckets

Переглянемо файли за цим шляхом.

```
hdfs dfs -ls
hdfs://cluster-11a7-m/user/hive/warehouse/lending_club_buckets/
```

```
user@cluster-11a7-m:~$ hdfs dfs -ls hdfs://cluster-11a7-m/user/hive/warehouse/lending_club_buckets/
Found 3 items
drwxr-xr-x   - user hadoop          0 2025-09-14 08:13 hdfs://cluster-11a7-m/user/hive/warehouse/lending_club_buckets/verification=Not Verified
drwxr-xr-x   - user hadoop          0 2025-09-14 08:13 hdfs://cluster-11a7-m/user/hive/warehouse/lending_club_buckets/verification=Source Verified
drwxr-xr-x   - user hadoop          0 2025-09-14 08:13 hdfs://cluster-11a7-m/user/hive/warehouse/lending_club_buckets/verification=Verified
user@cluster-11a7-m:~$
```

Виміряємо час для різних значень.

```
SELECT * FROM lending_club_buckets WHERE working_years =
0;
```

```
hive> EXPLAIN ANALYZE SELECT * FROM lending_club_buckets WHERE working_years = 0;
OK
OK
Plan optimized by CBO.

Stage-0
  Fetch Operator
    limit:-1
    Select Operator [SEL_2]
      Output:["_col0","_col1","_col2","_col3","_col4","_col5","_col6","_col7","_col8","_col9","_col10","_col11","_col12","_col13","_col14","_col15","_col16","_col17","_col18","_col19"]
      Filter Operator [FIL_4]
        predicate:(working_years = 0)
        TableScan [TS_0]
          Output:["loan_amount","payments_term","monthly_payment","grade","working_years","home","annual_income","purpose","debt_to_income","delinquency","inquiries","open_credit_lines","derogatory_records","revolving_balance","revolving
_rate","total_accounts","bankruptcies","fico_average","loan_risk"]

Time taken: 3.698 seconds, Fetched: 12 row(s)
hive>
```

```
SELECT * FROM lending_club_buckets WHERE working_years =
1;
```

```
hive> EXPLAIN ANALYZE SELECT * FROM lending_club_buckets WHERE working_years = 1;
OK
OK
Plan optimized by CBO.

Stage-0
  Fetch Operator
    limit:-1
    Select Operator [SEL_2]
      Output:["_col0","_col1","_col2","_col3","_col4","_col5","_col6","_col7","_col8","_col9","_col10","_col11","_col12","_col13","_col14","_col15","_col16","_col17","_col18","_col19"]
      Filter Operator [FIL_4]
        predicate:(working_years = 1)
        TableScan [TS_0]
          Output:["loan_amount","payments_term","monthly_payment","grade","working_years","home","annual_income","purpose","debt_to_income","delinquency","inquiries","open_credit_lines","derogatory_records","revolving_balance","revolving
_rate","total_accounts","bankruptcies","fico_average","loan_risk"]

Time taken: 3.697 seconds, Fetched: 12 row(s)
hive>
```

```
SELECT * FROM lending_club_buckets WHERE working_years =
10;
```

```
hive> EXPLAIN ANALYZE SELECT * FROM lending_club_buckets WHERE working_years = 10;
OK
OK
Plan optimized by CBO.

Stage-0
  Fetch Operator
    limit:-1
    Select Operator [SEL_2]
      Output:["_col0","_col1","_col2","_col3","_col4","_col5","_col6","_col7","_col8","_col9","_col10","_col11","_col12","_col13","_col14","_col15","_col16","_col17","_col18","_col19"]
      Filter Operator [FIL_4]
        predicate:(working_years = 10)
        TableScan [TS_0]
          Output:["loan_amount","payments_term","monthly_payment","grade","working_years","home","annual_income","purpose","debt_to_income","delinquency","inquiries","open_credit_lines","derogatory_records","revolving_balance","revolving
_rate","total_accounts","bankruptcies","fico_average","loan_risk"]

Time taken: 1.914 seconds, Fetched: 12 row(s)
hive>
```

2.7 Завдання 1.6

Створюємо зовнішню таблицю за базовою директорією для експорту.

```
SET hive.exec.dynamic.partition=true;
SET hive.exec.dynamic.partition.mode=nonstrict;
CREATE EXTERNAL TABLE lending_club_export_by_years (
    loan_amount           INT,
    payments_term         STRING,
    monthly_payment       DOUBLE,
    grade                 INT,
    home                  STRING,
    annual_income         DOUBLE,
    verification          STRING,
    purpose               STRING,
    debt_to_income        DOUBLE,
    delinquency           INT,
    inquiries             INT,
    open_credit_lines     INT,
    derogatory_records    INT,
    revolving_balance     INT,
    revolving_rate        DOUBLE,
    total_accounts        INT,
    bankruptcies          INT,
    fico_average          INT,
    loan_risk             STRING
)
PARTITIONED BY (working_years INT)
ROW FORMAT DELIMITED
  FIELDS TERMINATED BY ';'
STORED AS TEXTFILE
LOCATION
'hdfs://cluster-11a7-m/user/hive/warehouse/exports/lendin
```

```
g_club_by_working_years';
```

```
    > CREATE EXTERNAL TABLE lending_club_export_by_years (
    >     loan_amount           INT,
    >     payments_term         STRING,
    >     monthly_payment       DOUBLE,
    >     grade                 INT,
    >     home                  STRING,
    >     annual_income         DOUBLE,
    >     verification          STRING,
    >     purpose               STRING,
    >     debt_to_income        DOUBLE,
    >     delinquency           INT,
    >     inquiries             INT,
    >     open_credit_lines     INT,
    >     derogatory_records    INT,
    >     revolving_balance     INT,
    >     revolving_rate        DOUBLE,
    >     total_accounts        INT,
    >     bankruptcies          INT,
    >     fico_average          INT,
    >     loan_risk             STRING
    > )
    > PARTITIONED BY (working_years INT)
    > ROW FORMAT DELIMITED
    >   FIELDS TERMINATED BY ';'
    > STORED AS TEXTFILE
    > LOCATION 'hdfs://cluster-11a7-m/user/hive/warehouse/exports/lending_club_by_working_years';
OK
Time taken: 0.118 seconds
```

```
INSERT OVERWRITE TABLE lending_club_export_by_years
PARTITION (working_years)
SELECT
    loan_amount,
    payments_term,
    monthly_payment,
    grade,
    home,
    annual_income,
    verification,
    purpose,
    debt_to_income,
```

```
    delinquency,
    inquiries,
    open_credit_lines,
    derogatory_records,
    revolving_balance,
    revolving_rate,
    total_accounts,
    bankruptcies,
    fico_average,
    loan_risk,
    working_years
FROM lending_club_buckets;
SHOW PARTITIONS lending_club_export_by_years;
```

```
hive>
    > -- 2) Записати дані з bucket-таблиці, розклавши по партиціях working_years
    > INSERT OVERWRITE TABLE lending_club_export_by_years PARTITION (working_years)
    > SELECT
    >     loan_amount,
    >     payments_term,
    >     monthly_payment,
    >     grade,
    >     home,
    >     annual_income,
    >     verification,
    >     purpose,
    >     debt_to_income,
    >     delinquency,
    >     inquiries,
    >     open_credit_lines,
    >     derogatory_records,
    >     revolving_balance,
    >     revolving_rate,
    >     total_accounts,
    >     bankruptcies,
    >     fico_average,
    >     loan_risk,
    >     working_years
    > FROM lending_club_buckets;
Query ID = user_20250914083335_27248fa1-9767-4282-a7a7-34ce21862e21
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1757774494954_0033)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      2          2        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 15.69 s
--------------------------------------------------------------------------------
Loading data to table default.lending_club_export_by_years partition (working_years=null)

Loaded : 11/11 partitions.
         Time taken to load dynamic partitions: 0.669 seconds
         Time taken for adding to write entity : 0.003 seconds
OK
Time taken: 79.901 seconds
```

Переглянемо створені partitions.

```
SHOW PARTITIONS lending_club_export_by_years;
```

```
    > SHOW PARTITIONS lending_club_export_by_years;
OK
working_years=0
working_years=1
working_years=10
working_years=2
working_years=3
working_years=4
working_years=5
working_years=6
working_years=7
working_years=8
working_years=9
Time taken: 0.067 seconds, Fetched: 11 row(s)
```

Переглянемо каталоги і файли.

```
hdfs dfs -ls
hdfs://cluster-11a7-m/user/hive/warehouse/exports/lending
_club_by_working_years/
```

```
user@cluster-11a7-m:~$ hdfs dfs -ls hdfs://cluster-11a7-m/user/hive/warehouse/exports/lending_club_by_working_years/
Found 11 items
drwxr-xr-x   - user hadoop          0 2025-09-14 08:34 hdfs://cluster-11a7-m/user/hive/warehouse/exports/lending_club_by_working_years/working_years=0
drwxr-xr-x   - user hadoop          0 2025-09-14 08:34 hdfs://cluster-11a7-m/user/hive/warehouse/exports/lending_club_by_working_years/working_years=1
drwxr-xr-x   - user hadoop          0 2025-09-14 08:34 hdfs://cluster-11a7-m/user/hive/warehouse/exports/lending_club_by_working_years/working_years=10
drwxr-xr-x   - user hadoop          0 2025-09-14 08:34 hdfs://cluster-11a7-m/user/hive/warehouse/exports/lending_club_by_working_years/working_years=2
drwxr-xr-x   - user hadoop          0 2025-09-14 08:34 hdfs://cluster-11a7-m/user/hive/warehouse/exports/lending_club_by_working_years/working_years=3
drwxr-xr-x   - user hadoop          0 2025-09-14 08:34 hdfs://cluster-11a7-m/user/hive/warehouse/exports/lending_club_by_working_years/working_years=4
drwxr-xr-x   - user hadoop          0 2025-09-14 08:34 hdfs://cluster-11a7-m/user/hive/warehouse/exports/lending_club_by_working_years/working_years=5
drwxr-xr-x   - user hadoop          0 2025-09-14 08:34 hdfs://cluster-11a7-m/user/hive/warehouse/exports/lending_club_by_working_years/working_years=6
drwxr-xr-x   - user hadoop          0 2025-09-14 08:34 hdfs://cluster-11a7-m/user/hive/warehouse/exports/lending_club_by_working_years/working_years=7
drwxr-xr-x   - user hadoop          0 2025-09-14 08:34 hdfs://cluster-11a7-m/user/hive/warehouse/exports/lending_club_by_working_years/working_years=8
drwxr-xr-x   - user hadoop          0 2025-09-14 08:34 hdfs://cluster-11a7-m/user/hive/warehouse/exports/lending_club_by_working_years/working_years=9
user@cluster-11a7-m:~$
```

2.8 Завдання 1.7

Завантажимо дані на віртуальну машину.

```
gcloud compute scp articles.csv
user@cluster-11a7-m:~/articles.csv
```

```
user@archlinux:~/university/masters_first_semester_bigdata/lab_1$ gcloud compute scp articles.csv user@cluster-11a7-m:~/articles.csv
No zone specified. Using zone [us-central1-a] for instance: [cluster-11a7-m].
articles.csv
user@archlinux:~/university/masters_first_semester_bigdata/lab_1$
```

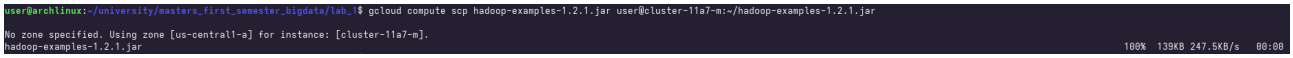
Завантажимо дані в hdfs.

```
hdfs dfs -mkdir -p /data/wordcount/input
hdfs dfs -put -f articles.csv /data/wordcount/input/
```

```
user@cluster-11a7-m:~$ hdfs dfs -mkdir -p /data/wordcount/input
user@cluster-11a7-m:~$ hdfs dfs -put -f articles.csv /data/wordcount/input/
user@cluster-11a7-m:~$
```

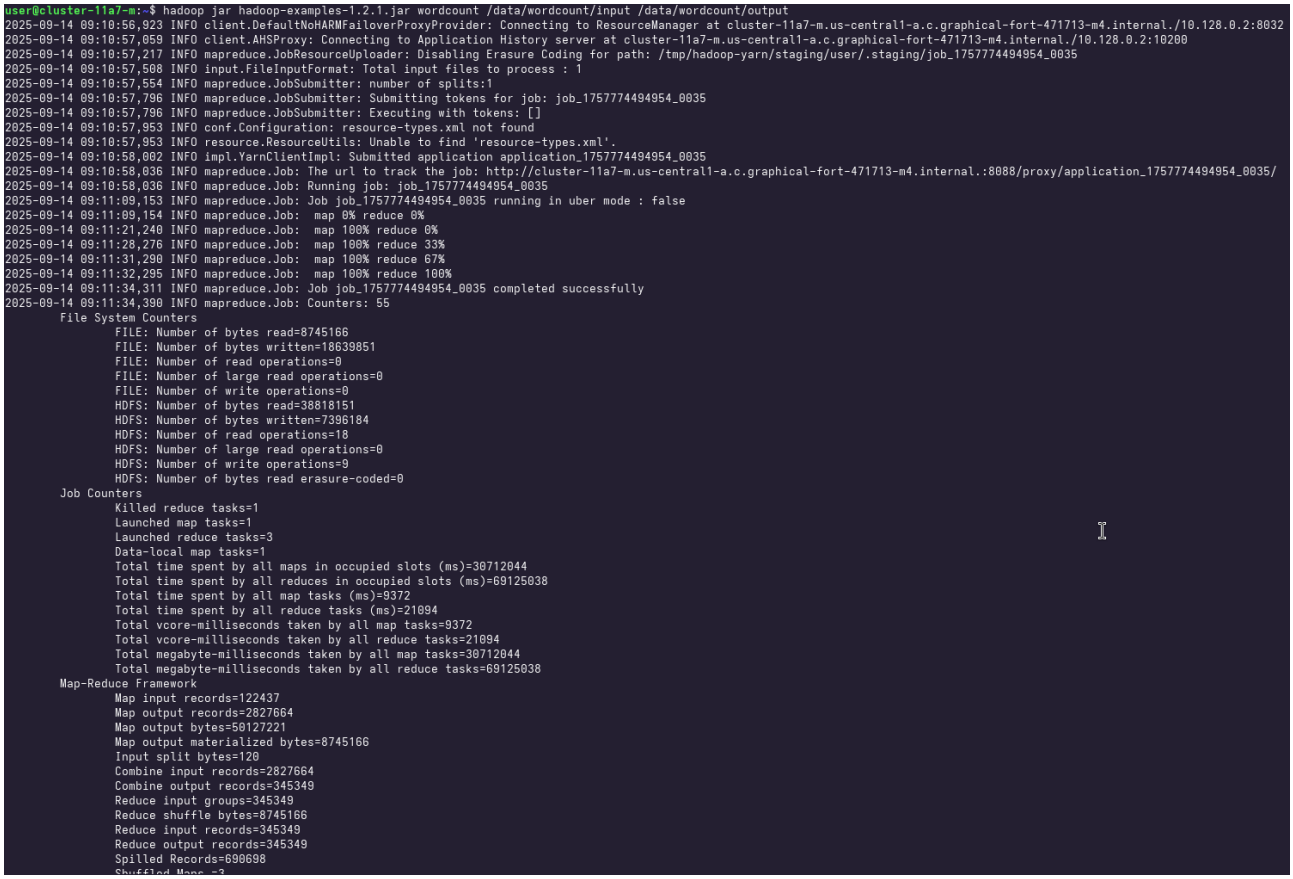
Завантажимо hadoop-examples-1.2.1 за посиланням https://repo1.maven.org/maven2/org/apache/hadoop/hadoop-examples/1.2.1/hadoop-examples-1.2.1.jar (ПРИМІТКА: Чому я сам маю шукати це посилання в інтернеті? Чому це посилання не вказано у лабораторній?). Завантажимо його на віртуальну машину.

```
gcloud compute scp hadoop-examples-1.2.1.jar
user@cluster-11a7-m:~/hadoop-examples-1.2.1.jar
```



Запустимо wordcount.

```
hdfs dfs -rm -r -f /data/wordcount/output
hadoop jar hadoop-examples-1.2.1.jar wordcount
/data/wordcount/input /data/wordcount/output
```



Подивимося результати частоти слів.

```
hdfs dfs -ls /data/wordcount/output
```

```
user@cluster-11a7-m:~$ hdfs dfs -ls /data/wordcount/output
Found 4 items
-rw-r--r--   2 user hadoop          0 2025-09-14 09:11 /data/wordcount/output/_SUCCESS
-rw-r--r--   2 user hadoop    2466624 2025-09-14 09:11 /data/wordcount/output/part-r-00000
-rw-r--r--   2 user hadoop    2468934 2025-09-14 09:11 /data/wordcount/output/part-r-00001
-rw-r--r--   2 user hadoop    2460626 2025-09-14 09:11 /data/wordcount/output/part-r-00002
user@cluster-11a7-m:~$
```

Переглянемо 20 найчастіших слів.

```
hdfs dfs -cat /data/wordcount/output/part-* | sort
-k2,2nr | head -20
```

```
user@cluster-11a7-m:~$ hdfs dfs -cat /data/wordcount/output/part-* | sort -k2,2nr | head -20
в       77626
и       54933
на      40415
что     33981
не      29032
с       23839
по      15781
-       13255
—       12067
В       11732
это     10399
Украины 10307
о       10192
для     9808
из      9621
как     9449
к       8845
–       8503
а       8470
за      8022
user@cluster-11a7-m:~$
```

2.9 Відповіді на запитання

2.9.1 Опис обробки запиту до завдання 1.2

```
SELECT name, edrpou, address, ROW_NUMBER() OVER
(PARTITION BY address ORDER BY edrpou) AS rn_by_place
FROM uotable LIMIT 20;
```

Вивід запиту:

```
hive> SELECT name, edrpou, address, ROW_NUMBER() OVER (PARTITION BY address ORDER BY edrpou) AS rn_by_place FROM
uotable LIMIT 20;
Query ID = user_20250917185201_5cea007b-f5db-439f-ad79-003d61f1a5b2
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1758134565428_0002)
```

```
--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      9          9        0        0       0       0
Reducer 2 ...... container     SUCCEEDED     49         49        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 30.92 s
--------------------------------------------------------------------------------
OK
```

На стороні Map виконується скан uotable по сплітах, тобто читаються лише потрібні колонки (name, edrpou, address). Map тут не рахує ROW_NUMBER(). Він лише розподіляє, тобто PARTITION BY, і забезпечує порядок, тобто ORDER BY.

На стороні Reducer відбувається потокове обчислення віконної функції, тобто для кожної групи address reducer проходить рядки у відсортованому порядку і лічильником виставляє ROW_NUMBER(). Далі виводяться готові рядки, а LIMIT 20 застосовується вже після обчислення.

У колонказ TOTAL, COMPLETED, RUNNING, PENDING, FAILED, KILLED виводиться кількість тасків для кожного Map та Reducer. У даному випадку для Map 1 ми маємо 9 завершиних завдань, тобто 9 сплітів. Також маємо одну Reducer вершину, тобто 49 тасків, визначений налаштуваннями hive.exec.reducers.bytes.per.reducer.

2.10 Визначення кількості Map та Reducer в залежності від запиту

Розглянемо запит та вивід до нього нижче.

```
EXPLAIN ANALYZE SELECT * FROM UO_table uo join FOP_table fop on uo.address = fop.address join FOP_table fop1 on uo.address = fop1.address;
```

```
hive> EXPLAIN ANALYZE SELECT * FROM uotable uo join fop_table fop on uo.address = fop.address join fop_table fop1 on uo.address = fop1.address;
No Stats for default@uotable, Columns: edrpou, address, boss, name, kved, stan, founders, fio
No Stats for default@fop_table, Columns: address, kved, stan, fio
No Stats for default@fop_table, Columns: address, kved, stan, fio
Query ID = user_20250917184411_49df2f53-e476-4c0b-9387-ee6fbedf166e
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1758134565428_0001)

--------------------------------------------------------------------------------
```

```
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------------------
Map 1            container        KILLED      6          0        0        6       0       0
Map 4 .......    container        KILLED      9          7        0        2       0       0
Reducer 2        container        KILLED    137          0        0      137       0       0
Reducer 3        container        KILLED    137          0        0      137       0       0
--------------------------------------------------------------------------------------------
VERTICES: 00/04  [>>-------------------------] 2%    ELAPSED TIME: 30.38 s
--------------------------------------------------------------------------------------------
```

Виникає питання чому саме створюються лише дві Map вершини: Map 1, Map 4? Чому не одна, не три, не N-на кількість? Аналогічно й для Reducer 2 та Reducer 3.

Map 1 та Map 4 створені для скану різних таблиць — uotable та fop_table. Reducer 2 та Reducer 3 — створені для двох послідовних джойнів.

Спробуємо заджойнити таблицю саму з собою, тобто виконається два скани таблиці, тобто 2 Map, і один Reducer.

```
VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      9          9        0        0       0       0
Map 3 .......... container     SUCCEEDED      9          9        0        0       0       0
Reducer 2 ..     container       RUNNING     97         37        3       57       0       0
--------------------------------------------------------------------------------------------
VERTICES: 02/03  [============>>--------------] 47%   ELAPSED TIME: 721.31 s
--------------------------------------------------------------------------------------------
```

3 ВИСНОВОК

У підсумку розгорнули та перевірили працездатність середовища, реалізували та запустили ETL-конвеєр на тестових даних, спроєктували схему БД і задокументували результати у звіті.