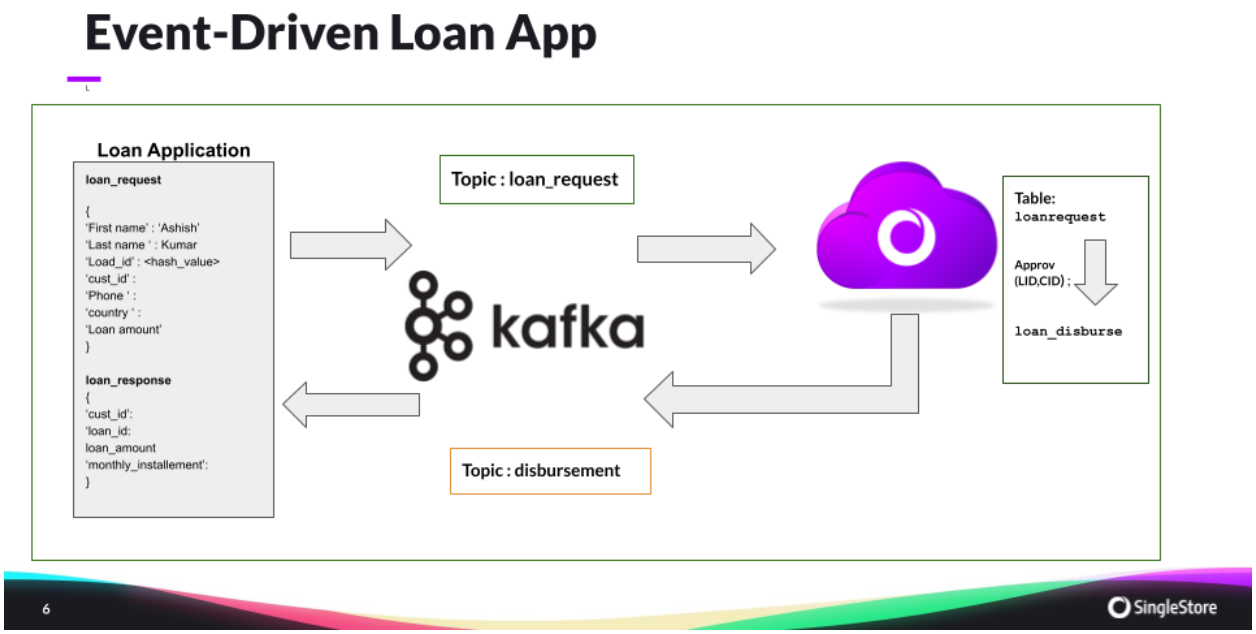
# SingleStore with Google Cloud - *Partner Dev-Day*

In this Hand-on Lab we will build Event-driven **LOAN PROCESSING APPLICATION.** Application will send the user data to the Kafka topic (loan_request) in JSON format The singlestore native pipeline will consume the JSON data from kafka topic and ingest it to a table (loanrequest) .The table will get populated with kafka-stream every second . We will create a procedure which will calculate the monthly emi for each loan which is processed and will store back in the database another table called loan_disburse. Once the entry of approval is created it will send back the JSON data to kafka topic - disbursement so that it can be consumed by Application giving back the response to the user.

## Architecture :



**High level steps :**

1. **Create a SingleStore DB cluster.**
2. **Build Database, table and procedure .**
3. **Create an application with the help of apache Kafka( open source ) & python.**
4. **Start the Singlestore pipeline  to stream data from kafka to SingleStore DB**
5. **Process the LOAN and send the response back to the application.**

# Let's start building the application

1. **Create a SingleStore DB.**

**Launch your own SingleStore Cluster**

Duration: 10:00

Go to portal.singlestore.com
Choose Cloud or install:  Cloud

Job role: Developer

Cluster Name: “Workshop”
Cloud provider: GCP
Region: Us East 4 (N. Virginia)
Compute: use default
Storage: use default.

# Create a Database Cluster

Learn more about pricing details and plan types

## 1 Cluster Name

Workshop

Create a unique name for your cluster

## 2 Cloud Provider

AWS | GCP | Azure

## 3 Region

Region

US East 4 (N. Virginia)

Want to try SingleStore in another region? Contact Us.

## 4 Compute

S-00 Edit

| vCPUs | RAM | Credits per hour |
|---|---|---|
| 2 | 16 GB | 0.25 |

Compute consumes a fixed number of credits per hour, and can be scaled up or down to increase or decrease performance.
On-demand credits cost $2.60 for this provider and region. Learn about credits or contact us to prepay for a discount.

## 5 Storage

| DISK | GB per Month |
|---|---|
| Storage | $0.023 |

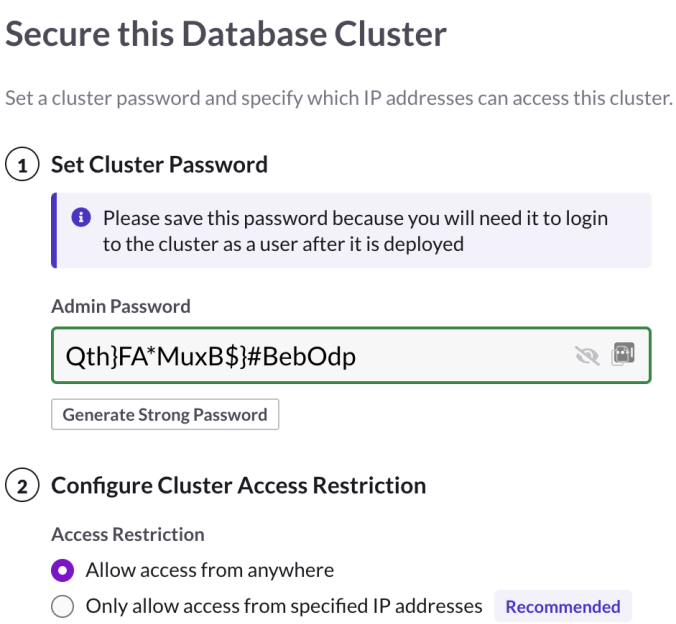
During your trial you get 4 TB per hour free.

## Click Next.

## Choose an admin password: (used generate password and write it down in Notepad): Qth}FA*MuxB$}#BebOdp

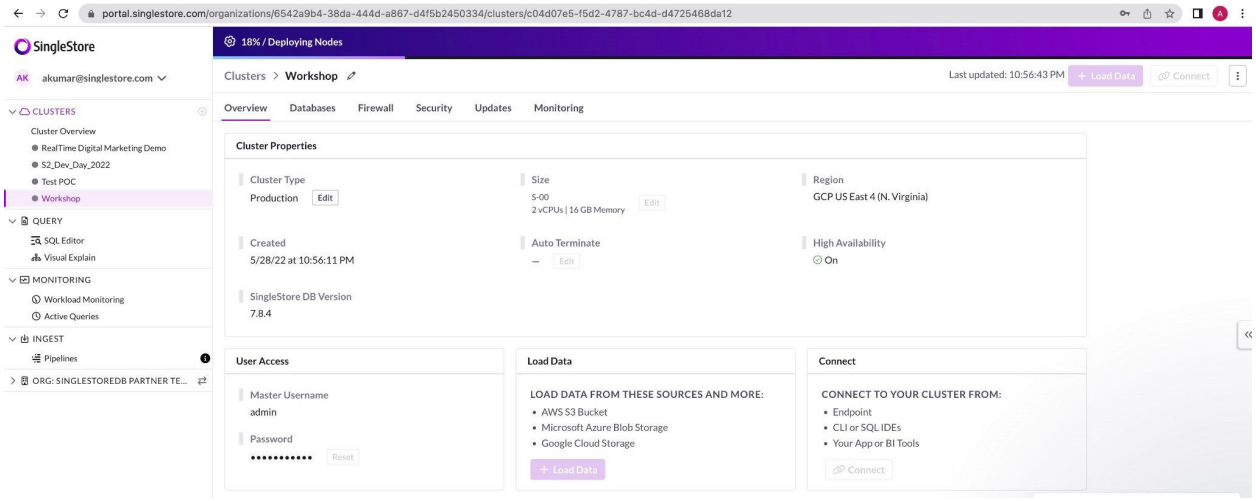
## Configure Cluster access restrictions

**Set to Allow Access from anywhere**



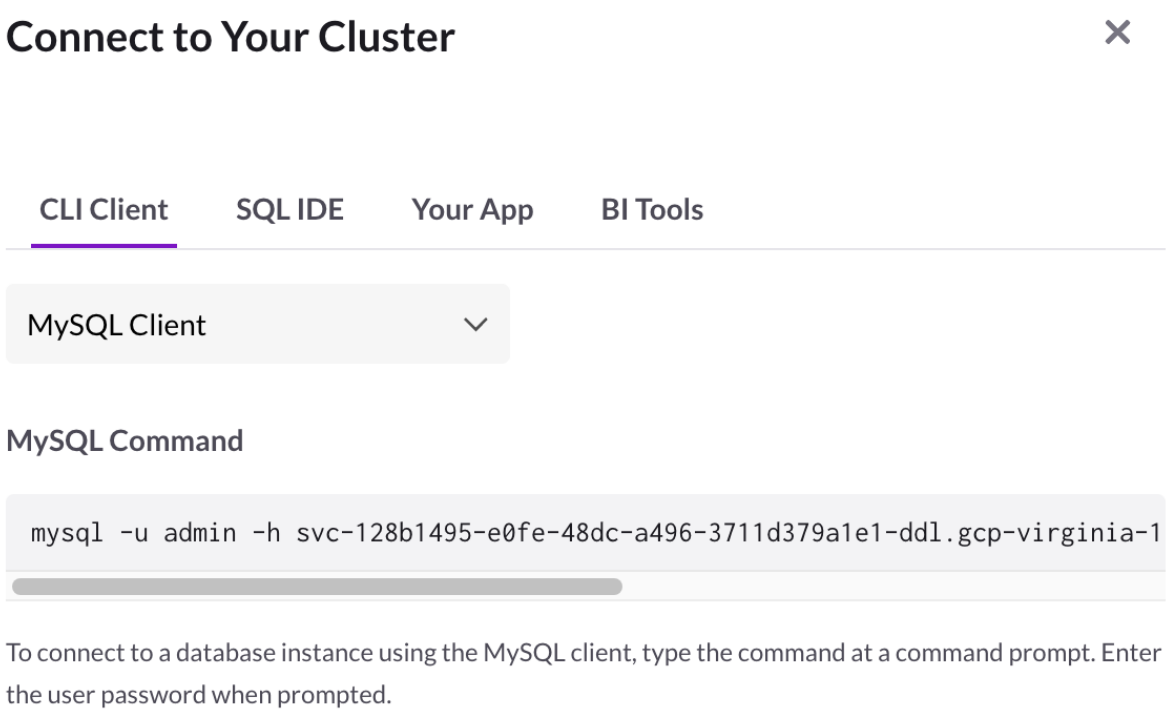

**Click Create Cluster**

**Resulting screen:**



**The banner at the top will indicate progress in connecting to the cluster. When this reaches 100% (about 3 minutes, the banner will turn green), you can continue to connect to the cluster.**

For Database Connection string , click on connect . Please note the string as we are going to use this in our lab


**At this point, we need to establish a Shell terminal session so that we can connect to our SingleStore Cluster.**

Connect to your linux terminal & Clone Github Repository to download the code for the workshop.

```
git clone https://github.com/shshkr09/SSGdevday.git
cd SSGdevday
```

```
[root@labdevday ~]#
[root@labdevday ~]#
[root@labdevday ~]# git clone https://github.com/shshkr09/SSGdevday.git
Cloning into 'SSGdevday'...
remote: Enumerating objects: 47, done.
remote: Counting objects: 100% (47/47), done.
remote: Compressing objects: 100% (35/35), done.
remote: Total 47 (delta 10), reused 47 (delta 10), pack-reused 0
Receiving objects: 100% (47/47), 180.65 KiB | 36.13 MiB/s, done.
Resolving deltas: 100% (10/10), done.
[root@labdevday ~]# ls
SSGdevday
[root@labdevday ~]# cd SSGdevday
[root@labdevday SSGdevday]# ls
__pycache__          create_pipeline.sql  deletetopic.sh          producer.py
approve.sql          create_table.sql     listtopic.sh            startkafka.sh
condisbursement.sh   createtopic.sh       proc_approve_loan.sql   stopkafka.sh
consumer.py          data_generator.py    proc_loanreq.sql
[root@labdevday SSGdevday]#
```

**Sample output :**

At this point we need to to modify 2 script with the ip address of our application vm

```
curl ifconfig.me
vi proc_approve_loan.sql
vi create_pipeline.sql
```

## 2. Build Database table and procedure

**Copy the MySQL Command that looks like this and run the command in the Terminal:**

*>> it will prompt for Singlestore Database password generated in step 1*

```
mysql -u admin -h <SingleStore DDL Service> -P
3306 --default-auth=mysql_native_password -p
```

### Sample output:

```
[root@labdevday SSGdevday]# mysql -u admin -h svc-c04d07e5-f5d2-4787-bc4d-d47254
68da12-ddl.gcp-virginia-1.svc.singlestore.com -P 3306 --default-auth=mysql_nativ
e_password -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 93014
Server version: 5.7.32 MemSQL source distribution (compatible; MySQL Enterprise
& MySQL Commercial)

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

So far we have been able to launch the cloudshell and connect to SingleStoreDB. Lets move to next step:

Execute the below in the mysql prompt

```
source create_table.sql
source proc_loanreq.sql
source proc_approve_loan.sql
```

**Sample Output:**

```
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> source create_table.sql
Query OK, 1 row affected (3.86 sec)

Database changed
Query OK, 0 rows affected (0.04 sec)

Query OK, 0 rows affected (0.01 sec)

mysql> source proc_loanreq.sql
Database changed
Query OK, 1 row affected (0.01 sec)

mysql> source proc_approve_loan.sql
Database changed
Query OK, 1 row affected (0.01 sec)

mysql>
```

**Well Done !!! Let’s move to the next step.**

3. **Create an application with the help of apache Kafka( open source ) & python.**

In this step we will install kafka and execute our python script to write JSON data to kafka topic

Execute the below to download the kafka and untar this.

```
wget https://dlcdn.apache.org/kafka/3.2.0/kafka_2.13-3.2.0.tgz
tar -xzf kafka_2.13-3.2.0.tgz
mv kafka_2.13-3.2.0 kafka
echo "advertised.listeners=PLAINTEXT://$(curl ifconfig.me):9092" >
/root/test>>$HOME/kafka/config/server.properties
```

**Sample output:**

```
[root@labdevday ~]# wget https://dlcdn.apache.org/kafka/3.2.0/kafka_2.13-3.2.0.tgz
--2022-05-30 09:22:05--  https://dlcdn.apache.org/kafka/3.2.0/kafka_2.13-3.2.0.tgz
Resolving dlcdn.apache.org (dlcdn.apache.org)... 151.101.2.132, 2a04:4e42::644
Connecting to dlcdn.apache.org (dlcdn.apache.org)|151.101.2.132|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 103955943 (99M) [application/x-gzip]
Saving to: 'kafka_2.13-3.2.0.tgz'

kafka_2.13-3.2.0.tg 100%[===================>]  99.14M   112MB/s    in 0.9s

2022-05-30 09:22:06 (112 MB/s) - 'kafka_2.13-3.2.0.tgz' saved [103955943/103955943]

[root@labdevday ~]# tar -xzf kafka_2.13-3.2.0.tgz
[root@labdevday ~]# mv kafka_2.13-3.2.0 kafka
[root@labdevday ~]#
[root@labdevday ~]#
[root@labdevday ~]# ls
SSGdevday  kafka  kafka_2.13-3.2.0.tgz
[root@labdevday ~]#
[root@labdevday ~]#
[root@labdevday ~]#
[root@labdevday ~]#
[root@labdevday ~]#
[root@labdevday ~]#
```

## Start the kafka by running the below script

```
cd SSGdevday
./startkafka.sh
ps -ef | grep kafka
```

### Sample output:

```
[root@labdevday SSGdevday]# ./startkafka.sh
starting  zookeeper
nohup: appending output to 'nohup.out'
starting kafka server
[root@labdevday SSGdevday]# nohup: appending output to 'nohup.out'

[root@labdevday SSGdevday]# ps -ef|grep kafka
root       38122       1  2 09:34 pts/0    00:00:01 java -Xmx512M -Xms512M -server -XX:+U
seG1GC -XX:MaxGCPauseMillis=20 -XX:InitiatingHeapOccupancyPercent=35 -XX:+ExplicitGCInvok
esConcurrent -XX:MaxInlineLevel=15 -Djava.awt.headless=true -Xloggc:/root/kafka/bin/../lo
gs/zookeeper-gc.log -verbose:gc -XX:+PrintGCDetails -XX:+PrintGCDateStamps -XX:+PrintGCTi
meStamps -XX:+UseGCLogFileRotation -XX:NumberOfGCLogFiles=10 -XX:GCLogFileSize=100M -Dcom
.sun.management.jmxremote -Dcom.sun.management.jmxremote.authenticate=false -Dcom.sun.man
agement.jmxremote.ssl=false -Dkafka.logs.dir=/root/kafka/bin/../logs -Dlog4j.configuratio
n=file:/root/kafka/bin/../config/log4j.properties -cp /root/kafka/bin/../libs/activation-
1.1.1.jar:/root/kafka/bin/../libs/aopalliance-repackaged-2.6.1.jar:/root/kafka/bin/../lib
s/argparse4j-0.7.0.jar:/root/kafka/bin/../libs/audience-annotations-0.5.0.jar:/root/kafka
/bin/../libs/commons-cli-1.4.jar:/root/kafka/bin/../libs/commons-lang3-3.8.1.jar:/root/ka
fka/bin/../libs/connect-api-3.2.0.jar:/root/kafka/bin/../libs/connect-basic-auth-extensio
n-3.2.0.jar:/root/kafka/bin/../libs/connect-json-3.2.0.jar:/root/kafka/bin/../libs/connec
t-mirror-3.2.0.jar:/root/kafka/bin/../libs/connect-mirror-client-3.2.0.jar:/root/kafka/bi
n/../libs/connect-runtime-3.2.0.jar:/root/kafka/bin/../libs/connect-transforms-3.2.0.jar:
```

## Create topic by executing below

```
./createtopic.sh
./listtopic.sh
```

### Sample output:

```
[root@labdevday SSGdevday]# ./createtopic.sh
WARNING: Due to limitations in metric names, topics with a period ('.') or underscore ('_
') could collide. To avoid issues it is best to use either, but not both.
Created topic loan_request.
Created topic disbursement.
[root@labdevday SSGdevday]# ./listtopic.sh
__consumer_offsets
disbursement
loan_request
[root@labdevday SSGdevday]#
```

### 4. Start the Singlestore pipeline to stream data from kafka to SingleStore DB

Goto to the previous terminal and start the pipeline by running below command in mysql prompt.

```
source create_pipline.sql
```

**Sample output:**

```
mysql> source create_pipeline.sql
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
Query OK, 0 rows affected (0.02 sec)

Query OK, 0 rows affected (0.01 sec)

mysql>
```

Execute the python script to generate the user data

```
python3 producer.py
```

**Sample output:**

```
[root@labdevday SSGdevday]# python3 producer.py
Producing message @ 2022-05-31 10:12:16.073615 | Message = {'cust_ids': 541, 'loan_amount': 1
929, 'loan_id': 'hSrBGpFSzdXdgkaYFSILzeAcOsGsMJpZ', 'first_name': 'Tammy', 'last_name': 'Glov
er', 'country': 'Mongolia', 'phone': '953.103.8785x8134'}
Producing message @ 2022-05-31 10:12:22.093323 | Message = {'cust_ids': 2, 'loan_amount': 648
79, 'loan_id': 'XBZYJCtJMByAAkgUDvUztmuplKBDOwFP', 'first_name': 'Christine', 'last_name': 'D
aniel', 'country': 'Taiwan', 'phone': '(252)428-1018x81745'}
Producing message @ 2022-05-31 10:12:23.113397 | Message = {'cust_ids': 451, 'loan_amount': 6
8240, 'loan_id': 'TRQrEmjEXoADjmUCWLfcUBFBOfQSQDcO', 'first_name': 'Andrea', 'last_name': 'Wo
ods', 'country': 'Madagascar', 'phone': '(067)162-3126x305'}
Producing message @ 2022-05-31 10:12:31.140100 | Message = {'cust_ids': 839, 'loan_amount': 1
3052, 'loan_id': 'BhzDehORmRSGuUHMpgySgGfmJVeSSunS', 'first_name': 'Shannon', 'last_name': 'G
allagher', 'country': 'Marshall Islands', 'phone': '+1-474-409-2202x88450'}
```

At this point of time  you should be seeing the data getting ingested to the table in the database

Execute the below to verify

```
use appws;
select count(*) from loanrequest;
```

**Sample output:**

```
mysql> use appws;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select count(*) from loanrequest;
+----------+
| count(*) |
+----------+
|      137 |
+----------+
1 row in set (0.03 sec)

mysql>
mysql>
mysql>
mysql>
mysql>
mysql>
mysql>
mysql>
mysql>
mysql>
```

## 5. Process the LOAN and send the response back to the application

In this step we will collect a user information from loanrequest and will approve his loan.

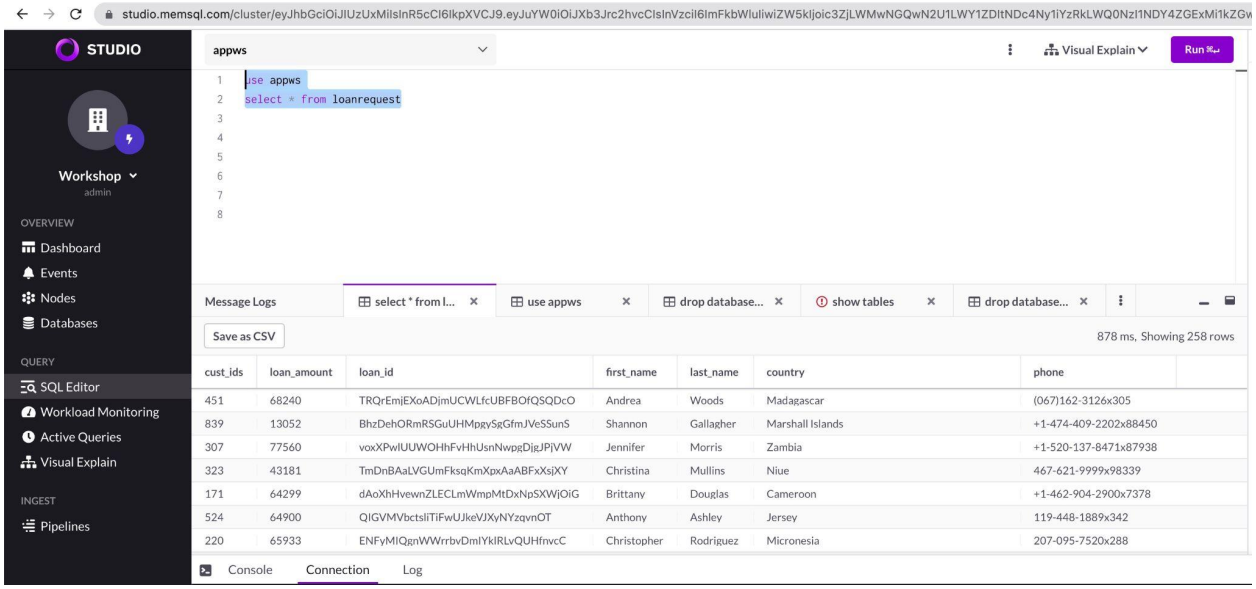
Launch a duplicate session of your terminal and execute the below

```
cd SSGdevday
./condisbursement.sh
```

Pick a customer id and loan id  and call the approve procedure.We will do this at singlestoreStudio.

```
use appws
select * from loanrequest
#please choose cust_ids & loan_id from result of above sql
CALL approve_loan(451,'TRQrEmjEXoADjmUCWLfcUBFBOfQSQDcO')
```

### Sample output:




As a result of above approval process you will find a customer detailed sent kafka topic and a entry will populate in ./condisbursement.sh  session

```
[root@labdevday ~]# cd SSGdevday/
[root@labdevday SSGdevday]# ls
__pycache__          createtopic.sh        proc_loanreq.sql
approve.sql          data_generator.py     producer.py
condisbursement.sh   deletetopic.sh        startkafka.sh
consumer.py          listtopic.sh          stopkafka.sh
create_pipeline.sql  nohup.out
create_table.sql     proc_approve_loan.sql
[root@labdevday SSGdevday]# ./condisbursement.sh
{"cust_ids":451,"loan_amount":68240,"loan_id":"TRQrEmjEXoADjmUCWLfcUBFBOfQSQDcO"
,"monthly_inst":6653}
```

There will be an entry to the load_disbursement table as well. Let's validate this with the below command.

```
use appws
select * from loan_disburse
```

appws

```
use appws
select * from loan_disburse
```

Message Logs | select * from l... | select * from l... | use appws sel... | CALL approve...

Save as CSV

| cust_ids | loan_id | loan_amount | monthly_inst |
|---|---|---|---|
| 451 | TRQrEmjEXoADjmUCWLfcUBFBOfQSQDcO | 68240 | 6653 |

# That’s it

# Congratulations!!!