



Faculty of Engineering and Applied Science
Cloud Computing SOFE 4630U CRN: 74293
Lab 3

Due: March 18th 2022
Name: Iliya Karac
Student ID: 100703933

Github repository:

Videos :

Task1: Videos watched

Task 2:

Sink and source connectors are encapsulated by kafka connect which houses the connector, connect record, and the converter. The connector is a java class set up as an interface that can read data from external sources and start to process it for the converter. This is how it works for a source, it is the inverse for a sink. The data comes out of the converted and recorded. The connector then outputs the data in the proper format to the sink. Source is input, sink is output.

The advantage of using a service like confluent is that you can bypass most programming and set up the connections through a user interface most likely. But this only applies to a selection of mainstream services. If you want to use a private system as your source or system you will need kafka connectors. Fortunately they are also fairly simple to use. Instead of rewriting the code for every connection you can use a connector and declare all the connections you need. They also record data automatically which is useful in data storage.

A kafka connector is a worker object that runs in a java virtual machine. Many of these worker objects can be made to balance the load of the tasks. This also means that there are more connectors working which allows for availability. If a worker were to go down its specifications are stored in kafka and a new worker with the same specifications as the previous one exactly where the old one went down.

- Avro - each key is unique and the values can be vastly different, you can specify the type of values you need and store data in a KV database
- Protobuf - Integrates nicely with schema registry, serializes in protobuf format
- String - converts to string, good for document type NoSQL best option if the data is being processed immediately then dumped
- JSON - default format for serialization, compatible with most systems visually legible
- JSON schema - just like JSON but it adds more bytes to the file, this allows for a low level check of the file authenticity or it can be used for sorting JSON files that have a different function

Task 3:

A Key value database is a type of database that falls under the NoSQL family of query languages. As the name suggests it consists of 1 key (a unique identifier) and a value (any sort of data). All the data entries can follow a specific format but are allowed to have vastly different attributes that may be unrelated. The database is also non relational meaning that the key values pairs cannot be related to each other by a specified attribute.

Advantages:

- Simple
- Fast
- Logs all types of data
- Reliable
- Scalable
- Flexible

Disadvantages:

- Single key
- Single value
- Hard to look up values
- Collects lots of junk if not set up optimally
- Non-relational

Popular KV databases:

- MongoDB
- Hbase
- Redis
- Aerospike
- Amazon DynamoDB
- Oracle NoSQL database
- Oracle Berkeley DB
- Voldemort

Task 4:

```
Cloud Shell Editor

Welcome to Cloud Shell! Type "help" to get started.
Your Cloud Platform project in this session is set to firstkubert.
Use "gcloud config set project [PROJECT_ID]" to change to a different project.
karac_iliya@cloudshell:~ (firstkubert)$ kubectl apply -f mysql-app.yaml
error: the path "mysql-app.yaml" does not exist
karac_iliya@cloudshell:~ (firstkubert)$ ls
box index2.html index.html README-cloudshell.txt SOFE4630U-tut3
karac_iliya@cloudshell:~ (firstkubert)$ cd SOFE4630U-tut3/
karac_iliya@cloudshell:~/SOFE4630U-tut3 (firstkubert)$ ls
connectors GKE python
karac_iliya@cloudshell:~/SOFE4630U-tut3 (firstkubert)$ cd GKE
karac_iliya@cloudshell:~/SOFE4630U-tut3/GKE (firstkubert)$ kubectl apply -f mysql-app.yaml
service/mysql created
deployment.apps/mysql created
karac_iliya@cloudshell:~/SOFE4630U-tut3/GKE (firstkubert)$ kubectl get pods
NAME READY STATUS RESTARTS AGE
mysql-7dcb5fd764-thzhw 0/1 Pending 0 13m
karac_iliya@cloudshell:~/SOFE4630U-tut3/GKE (firstkubert)$ kubectl get deployment
NAME READY UP-TO-DATE AVAILABLE AGE
mysql 0/1 1 0 13m
karac_iliya@cloudshell:~/SOFE4630U-tut3/GKE (firstkubert)$ kubectl get services
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
kubernetes ClusterIP 10.28.0.1 <none> 443/TCP 42d
mysql LoadBalancer 10.28.11.53 34.71.231.64 3306:30095/TCP 14m
karac_iliya@cloudshell:~/SOFE4630U-tut3/GKE (firstkubert)$ ^C
karac_iliya@cloudshell:~/SOFE4630U-tut3/GKE (firstkubert)$
```

The mysql app has been deployed on <http://34.71.231.64/>

```
karac_iliya@cloudshell:~/SOFE4630U-tut3/GKE (firstkubert)$ kubectl -it mysql-7dcb5fd764-thzhw exec --mysql
Error: flags cannot be placed before plugin name: -it
karac_iliya@cloudshell:~/SOFE4630U-tut3/GKE (firstkubert)$ kubectl exec -it mysql-7dcb5fd764-thzhw --mysql
error: unknown flag: --mysql
See 'kubectl exec --help' for usage.
karac_iliya@cloudshell:~/SOFE4630U-tut3/GKE (firstkubert)$ kubectl exec -it mysql-7dcb5fd764-thzhw --mysql
Error from server (BadRequest): pod mysql-7dcb5fd764-thzhw does not have a host assigned
karac_iliya@cloudshell:~/SOFE4630U-tut3/GKE (firstkubert)$ kubectl get pod mysql-7dcb5fd764-thzhw
NAME READY STATUS RESTARTS AGE
mysql-7dcb5fd764-thzhw 0/1 Pending 0 27m
karac_iliya@cloudshell:~/SOFE4630U-tut3/GKE (firstkubert)$ kubectl apply -f mysql-app.yaml
service/mysql unchanged
deployment.apps/mysql unchanged
karac_iliya@cloudshell:~/SOFE4630U-tut3/GKE (firstkubert)$ kubectl get services
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
kubernetes ClusterIP 10.28.0.1 <none> 443/TCP 42d
mysql LoadBalancer 10.28.11.53 34.71.231.64 3306:30095/TCP 28m
karac_iliya@cloudshell:~/SOFE4630U-tut3/GKE (firstkubert)$ kubectl get pods
NAME READY STATUS RESTARTS AGE
mysql-7dcb5fd764-thzhw 0/1 Pending 0 28m
karac_iliya@cloudshell:~/SOFE4630U-tut3/GKE (firstkubert)$ kubectl get pods
NAME READY STATUS RESTARTS AGE
mysql-7dcb5fd764-thzhw 0/1 Pending 0 30m
karac_iliya@cloudshell:~/SOFE4630U-tut3/GKE (firstkubert)$
```

Pod is not running

```

DROP TABLE IF EXISTS test;

CREATE TABLE IF NOT EXISTS test (
  id serial NOT NULL PRIMARY KEY,
  name varchar(100),
  email varchar(200),
  department varchar(200),
  modified timestamp default CURRENT_TIMESTAMP NOT NULL,
  INDEX `modified_index` (`modified`)
);
USE myDB;
INSERT INTO test (name, email, department) VALUES ('alice', 'alice@abc.com', 'eng.');
```

id	name	email	department
1	alice	alice@abc.com	eng.
2	bob1	bob1@abc.com	sales
3	bob2	bob2@abc.com	sales
4	bob3	bob3@abc.com	sales
5	bob4	bob4@abc.com	sales
6	bob5	bob5@abc.com	sales
7	bob6	bob6@abc.com	sales
8	bob7	bob7@abc.com	sales
9	bob8	bob8@abc.com	sales
10	bob9	bob9@abc.com	sales

```

INSERT INTO test (name, email, department) VALUES ('bob1', 'bob1@abc.com', 'sales');
INSERT INTO test (name, email, department) VALUES ('bob2', 'bob2@abc.com', 'sales');
INSERT INTO test (name, email, department) VALUES ('bob3', 'bob3@abc.com', 'sales');
INSERT INTO test (name, email, department) VALUES ('bob4', 'bob4@abc.com', 'sales');
INSERT INTO test (name, email, department) VALUES ('bob5', 'bob5@abc.com', 'sales');
INSERT INTO test (name, email, department) VALUES ('bob6', 'bob6@abc.com', 'sales');
INSERT INTO test (name, email, department) VALUES ('bob7', 'bob7@abc.com', 'sales');
INSERT INTO test (name, email, department) VALUES ('bob8', 'bob8@abc.com', 'sales');
INSERT INTO test (name, email, department) VALUES ('bob9', 'bob9@abc.com', 'sales');
```

karac_iliya@cloudshell:~/SOFE4630U-tut3/GKE (firstkubert)\$

Viewing the database scripts

```

Database changed
mysql> select * from test;
```

id	name	email	department
1	alice	alice@abc.com	eng.
2	bob1	bob1@abc.com	sales
3	bob2	bob2@abc.com	sales
4	bob3	bob3@abc.com	sales
5	bob4	bob4@abc.com	sales
6	bob5	bob5@abc.com	sales
7	bob6	bob6@abc.com	sales
8	bob7	bob7@abc.com	sales
9	bob8	bob8@abc.com	sales
10	bob9	bob9@abc.com	sales

```

karac_iliya@cloudshell:~/SOFE4630U-tut3/GKE (firstkubert)$ cat scli.sql | kubectl exec -i mysql-7dcb5fd764-2g9h2 -- mysql -user -pSOFE4630U
cat: scli.sql: No such file or directory
mysql: [Warning] Using a password on the command line interface can be insecure.
karac_iliya@cloudshell:~/SOFE4630U-tut3/GKE (firstkubert)$ ls
mysql-app.yaml mysql-pvc.yaml redis-app.yaml redis-pvc.yaml scli.sql
karac_iliya@cloudshell:~/SOFE4630U-tut3/GKE (firstkubert)$ cat scli.sql | kubectl exec -i mysql-7dcb5fd764-2g9h2 -- mysql -user -pSOFE4630U
mysql: [Warning] Using a password on the command line interface can be insecure.
karac_iliya@cloudshell:~/SOFE4630U-tut3/GKE (firstkubert)$ mysql -user -pSOFE4630U -h34.71.231.64
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.0.28 MySQL Community Server - GPL

Copyright (c) 2000, 2022, 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>
```

Populated db and mysql command

KV with redis:

```

karac_iliya@cloudshell:~/SOFE4630U-tut3/GKE (firstkubert)$ kubectl apply -f redis-pvc.yaml
persistentvolumeclaim/redis-volumeclaim created
karac_iliya@cloudshell:~/SOFE4630U-tut3/GKE (firstkubert)$ kubectl apply -f mysql-app.yaml
service/mysql created
deployment.apps/mysql created
karac_iliya@cloudshell:~/SOFE4630U-tut3/GKE (firstkubert)$ kubectl get pods
-bash: kubectl: command not found
karac_iliya@cloudshell:~/SOFE4630U-tut3/GKE (firstkubert)$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
mysql-7dcb5fd764-p29qx              1/1     Running   0          53s
karac_iliya@cloudshell:~/SOFE4630U-tut3/GKE (firstkubert)$
```

Redis deployed

```
deployment.apps/redis created
karac_iliya@cloudshell:~/SOPE4630U-tut3/GKE (firstkubert)$ kubectl exec -it mysql-7dcb5fd764-p29qx -- redis -cli
Error from server (NotFound): pods "mysql-7dcb5fd764-p29qx" not found
karac_iliya@cloudshell:~/SOPE4630U-tut3/GKE (firstkubert)$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
redis-56644c686c-lnbw1             1/1     Running   0           52s
karac_iliya@cloudshell:~/SOPE4630U-tut3/GKE (firstkubert)$ kubectl exec -it redis-56644c686c-lnbw1 -- redis -cli
error: Internal error occurred: error executing command in container: failed to exec in container: failed to start exec "ca3cae62c785c66e6087286df5fe0b414b809452ac30bbb2439154eb08a569b2": OCI runtime exec failed: exec failed: container_linux.go:380: starting container process caused: exec: "redis": executable file not found in $PATH: unknown
karac_iliya@cloudshell:~/SOPE4630U-tut3/GKE (firstkubert)$ kubectl exec -it redis-56644c686c-lnbw1 -- redis -cli
127.0.0.1:6379> auth SOPE4630U
(error): WRONGPASS invalid username-password pair or user is disabled.
127.0.0.1:6379> auth SOPE4630U
OK
127.0.0.1:6379> set k1 test
OK
127.0.0.1:6379> get k1
"test"
127.0.0.1:6379> █
```

KV db running

It can also be deployed locally or globally with an IP.

You can access redis with a client using your local machine code and libraries, the provided code takes key1's value in my case asdf and changes it to 30, but you can make it whatever you wish.

Task 8:

- open cv
- tensor flow
- image processing
- image manipulation
- neural network training