Kafka Connect

_

1 Goto the folder kafkaConnect_01
2 Open cmd from this folder
3 Once cmd is opened, type **docker login** in cmd and press enter. now you are logged into the docker
4 Then type **docker-compose up -d** this command will download all the images and will create a container.
5 Once your download is done check for the status of all the containers by typing **docker ps**

```
F:\11111\KafkaConnect_01>docker-compose up -d
[+] Running 7/7
 - Network kafkaconnect_01_default              Created                                    0.7s
 - Container kafkaconnect_01-postgres-1         Started                                    4.7s
 - Container kafkaconnect_01-zookeeper-1        Started                                    5.1s
 - Container kafkaconnect_01-mysql-1            Started                                    5.0s
 - Container kafkaconnect_01-kafka-1            Started                                    5.9s
 - Container kafkaconnect_01-schema-registry-1  Started                                    8.1s
 - Container kafkaconnect_01-debezium-1         Started                                    7.8s
```

.You can also goto docker desktop then
containers and check the status of your container

6 Now we have to create a postgresql database. Open postgresql CLI from docker desktop

Type below command
psql -U docker -d exampledb -W
Password docker → given in yml file
create table student(id integer primary key,name varchar);
Alter table student REPLICA IDENTITY FULL; --> it means we have to replicate the whole table
insert into student values(1,'asd');

==========

psql -U postgresuser inventory

Dont use this command just for reference **goto 7 point**
```
psql --username postgres
\l
```

```
GRANT CONNECT ON DATABASE inventory TO public;
\connect inventory
Alter table student REPLICA IDENTITY FULL;
```
==========

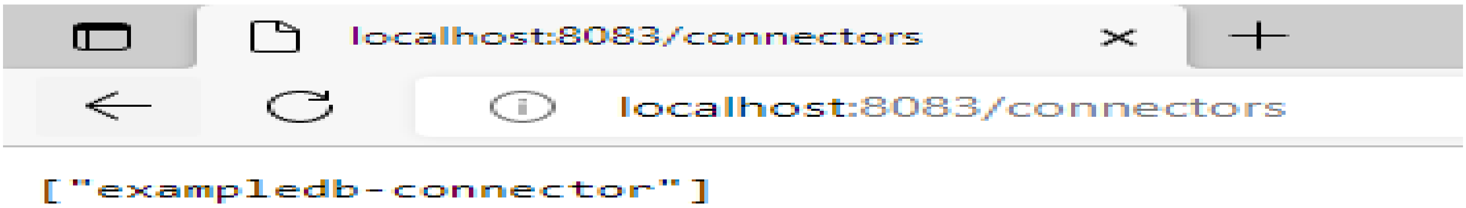
7 Now we have to create the source connector for postgresql. The configuration is given in debezium.json file
To create a source connector we have to type

curl -i -X POST -H "Accept:application/json" -H "Content-Type:application/json" http://localhost:8083/connectors/ --data "@debezium.json"

The above command will create a source connector.You will see a message as Created on the cmd.

```
F:\11111\KafkaConnect_01>curl -i -X POST -H "Accept:application/json" -H "Content-Type:application/json" http://localhos
t:8083/connectors/ --data "@debezium.json"
HTTP/1.1 201 Created
Date: Thu, 21 Jul 2022 14:18:49 GMT
Location: http://localhost:8083/connectors/exampledb-connector
Content-Type: application/json
Content-Length: 412
Server: Jetty(9.4.33.v20201020)
```

You could verify it by visiting http://localhost:8083/connectors
on this port we are running the debezium connector. You will see ["exampledb-connector"] created on your browser.



8 Now we have to login to our postgres database
Before doing that type **docker ps** command and check the container id and status of all the containers

```
F:\11111\KafkaConnect_01>docker ps
CONTAINER ID   IMAGE                                      COMMAND                  CREATED          STATUS          PORTS
                                                 NAMES
176281e15ecc   debezium/connect:1.4                       "/docker-entrypoint.…"   28 minutes ago   Up 28 minutes   8778/t
cp, 9092/tcp, 0.0.0.0:8083->8083/tcp, 9779/tcp   kafkaconnect_01-debezium-1
667b51f542a7   confluentinc/cp-schema-registry:5.5.3      "/etc/confluent/dock…"   28 minutes ago   Up 28 minutes   0.0.0.
0:8081->8081/tcp                                 kafkaconnect_01-schema-registry-1
9e083092d40e   confluentinc/cp-enterprise-kafka:5.5.3     "/etc/confluent/dock…"   28 minutes ago   Up 28 minutes   0.0.0.
0:9092->9092/tcp                                 kafkaconnect_01-kafka-1
56ef7f350b08   confluentinc/cp-zookeeper:5.5.3            "/etc/confluent/dock…"   28 minutes ago   Up 28 minutes   2181/t
cp, 2888/tcp, 3888/tcp                           kafkaconnect_01-zookeeper-1
97968a07dfbd   debezium/example-mysql:0.6                 "docker-entrypoint.s…"   28 minutes ago   Up 28 minutes   0.0.0.
0:3306->3306/tcp                                 kafkaconnect_01-mysql-1
e045cc5aab06   debezium/postgres:13                       "docker-entrypoint.s…"   28 minutes ago   Up 28 minutes   0.0.0.
0:5432->5432/tcp                                 kafkaconnect_01-postgres-1
```

9 Now we have to see whether the kafka is capturing our changes or not . For doing that we have to Type below command

docker run --tty --network kafkaconnect55555_default confluentinc/cp-kafkacat kafkacat -b kafka:9092 -C -s key=s -s value=avro -r http://schema-registry:8081 -t postgres.public.student

Kafkaconnect55555 → its the name of the container. You have to change accordingly as per your container name

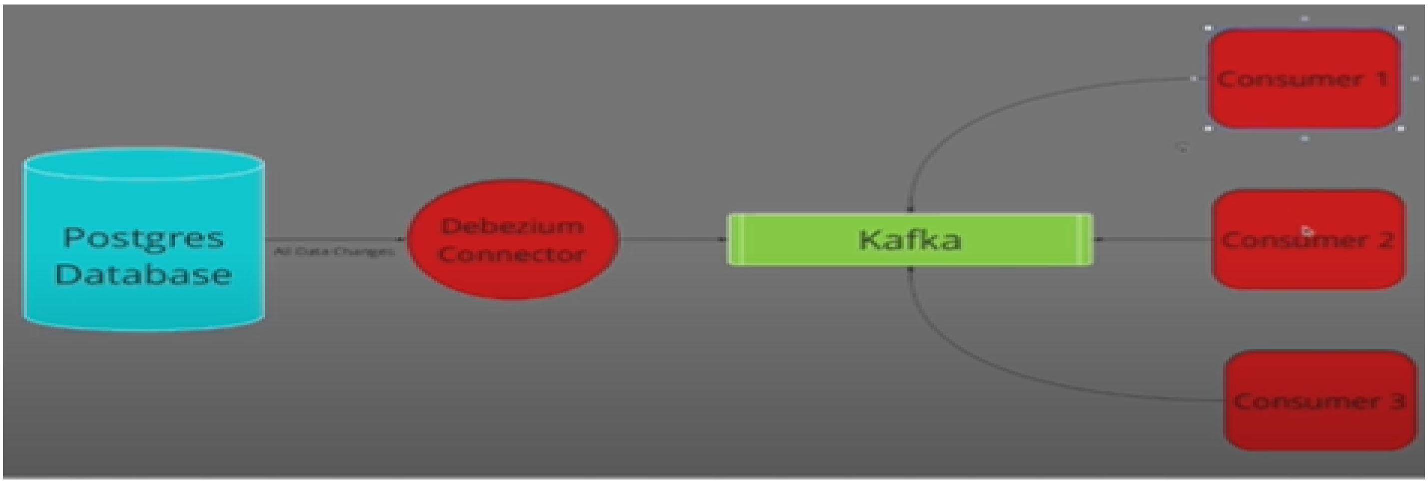
10 Now go to the postgresql cli and insert a record into student table
insert into student values(1,'asd');

```
exampledb=# insert into student values(1,'asd');
INSERT 0 1
exampledb=#
```

Once you insert the record in the table all the changes will be captured by the kafka connect

```
F:\11111\KafkaConnect55555>docker run --tty --network kafkaconnect55555_default confluentinc/cp-kafkacat kafkacat -b kaf
ka:9092 -C -s key=s -s value=avro -r http://schema-registry:8081 -t postgres.public.student
% Reached end of topic postgres.public.student [0] at offset 0
{"before": null, "after": {"Value": {"id": 1, "name": {"string": "asd"}}}, "source": {"version": "1.4.2.Final", "connect
or": "postgresql", "name": "postgres", "ts_ms": 1658471576128, "snapshot": {"string": "false"}, "db": "exampledb", "sche
ma": "public", "table": "student", "txId": {"long": 491}, "lsn": {"long": 23890112}, "xmin": null}, "op": "c", "ts_ms":
{"long": 1658471576568}, "transaction": null}
% Reached end of topic postgres.public.student [0] at offset 1
```

We were able to achieve below architecture



For sink connectors

11 now we have to create **(consumer)** a sink connector so that all the changes which we are doing in postgresql database could be reflected into another database.
We can create as much container as we like only this is you have to give the configuration of sink connector properly

12 create oracle database as a consumer.
First type below command

docker exec -it **c30ec9741d78** bash -c "source /home/oracle/.bashrc; sqlplus /nolog"

c30ec9741d78 —> container id of oracle

```
F:\11111\KafkaConnect_01>docker ps
CONTAINER ID   IMAGE                                                       COMMAND                  CREATED          STAT
US                              PORTS                                                       NAMES
09495104634f   debezium/connect:1.4                                        "/docker-entrypoint.…"   17 minutes ago   Up 1
7 minutes                       8778/tcp, 9092/tcp, 0.0.0.0:8083->8083/tcp, 9779/tcp        kafkaconnect_01-debezium-1
717f382e8832   confluentinc/cp-schema-registry:5.5.3                       "/etc/confluent/dock…"   17 minutes ago   Up 1
7 minutes                       0.0.0.0:8081->8081/tcp                                      kafkaconnect_01-schema-registry-1
d8d85fca08b1   confluentinc/cp-enterprise-kafka:5.5.3                      "/etc/confluent/dock…"   18 minutes ago   Up 1
7 minutes                       0.0.0.0:9092->9092/tcp                                      kafkaconnect_01-kafka-1
8c9f0e680124   debezium/example-mysql:0.6                                  "docker-entrypoint.s…"   18 minutes ago   Up 1
7 minutes                       0.0.0.0:3306->3306/tcp                                      kafkaconnect_01-mysql-1
9e35125fdfbb   confluentinc/cp-zookeeper:5.5.3                             "/etc/confluent/dock…"   18 minutes ago   Up 1
7 minutes                       2181/tcp, 2888/tcp, 3888/tcp                                kafkaconnect_01-zookeeper-1
d8bca4e74282   debezium/postgres:13                                        "docker-entrypoint.s…"   18 minutes ago   Up 1
7 minutes                       0.0.0.0:5432->5432/tcp                                      kafkaconnect_01-postgres-1
c30ec9741d78   container-registry.oracle.com/database/enterprise:latest   "/bin/sh -c 'exec $0…"   18 minutes ago   Up 1
7 minutes (health: starting)    0.0.0.0:1521->1521/tcp                                      kafkaconnect_01-oracle-db-1
```

Then type below commands one by one

connect sys as sysdba;

Here enter the password as **'Oradoc_db1'**

alter session set "_ORACLE_SCRIPT"=true;

```
create user dummy identified by dummy;
GRANT ALL PRIVILEGES TO dummy;
connect dummy;
```

==========

Challenges/ errors you might get

ORA-03114: not connected to ORACLE

Then again start with connect sys as sysdba;

If you are getting ORA-28000: The account is locked.

```
ALTER USER dummy ACCOUNT UNLOCK;
```

```
su - oracle

sqlplus / as sysdba

select name,DB_UNIQUE_NAME from v$database;

SELECT v.name, v.open_mode, NVL(v.restricted, 'n/a') "RESTRICTED", d.status

     FROM v$pdbs v, dba_pdbs d

     WHERE v.guid = d.guid

     ORDER BY v.create_scn;
```

==========

```
Now open your local oracle database
And create a new connection with following details
Username:-dummy
Password:-dummy
Hostname:-localhost
Port:-1521
SID:-ORCLCDB

And then press connect
```

```
create table student(id INTEGER PRIMARY key, name VARCHAR(20));
insert into student values(1,'asd');
```

To list all the topics goto debezium cli and type

```
bin/kafka-topics.sh --list --zookeeper localhost:2181
```

```
kafka-topics --zookeeper 127.0.0.1:2181 --topic postgres.public.student --describe
```

kafka-topics --zookeeper 127.0.0.1:2181 --list

```
kafka-console-consumer --bootstrap-server 127.0.0.1:9092 --topic postgres.public.student
```

Mysql sink connector

```
{
    "name": "jdbc_source_mysql_01",
    "config": {
    "connector.class": "io.confluent.connect.jdbc.JdbcSinkConnector",
    "task.max": 1,
    "connection.url": "jdbc:mysql://192.168.1.107:3306/inventory?serverTimezone=UTC",
    "connection.user": "root",
    "connection.password": "mysql",
    "update.mode":"update",
    "auto.create":true

    }
}
```

```
{
    "error_code": 400,
```

```
    "message": "Connector configuration is invalid and contains the following 1 error(s):\nUnable to connect: Failed to resolve Oracle database version\nYou can also find the above list of errors at the endpoint `/connector-plugins/{connectorType}/config/validate`"
}
```



```
{
    "name": "inventory-connfghgfhfghector",
    "config": {
        "connector.class" : "io.debezium.connector.oracle.OracleConnector",
        "tasks.max" : "1",
        "database.server.name" : "server1",
        "database.hostname" : "localhost",
        "database.port" : "1521",
        "database.user" : "dummy",
        "database.password" : "dummy",
        "database.dbname" : "ORCLCDB",
        "database.pdb.name" : "ORCLPDB1",
        "database.out.server.name" : "dbzxout",
        "database.history.kafka.bootstrap.servers" : "kafka:9092",
        "database.history.kafka.topic": "schema-changes.inventory"

  }
}
```


```
use local;
create database local;

create table student(id int(10) primary key, name varchar(10));


select * from student;


insert into student values(1,'abc');
```

```
insert into customers values(default, 'John', 'Doe', 'john.doe@example.com');
update customers set first_name='Jane', last_name='Roe' where last_name='Doe';
delete from customers where email='john.doe@example.com';
```

Update and insert is happening properly