Cassandra

- *First try to create account on datastax and create keyspace*
- *Try to download the secure connect bundle .zip file*
- *Then try to follow steps like create the token and download it the .json file</h4>*

In [1]:
```
pip install cassandra-driver
```
```
Requirement already satisfied: cassandra-driver in c:\users\varukish\appdata\local\anaconda3\lib\site-packages (3.28.0)
Requirement already satisfied: six>=1.9 in c:\users\varukish\appdata\local\anaconda3\lib\site-packages (from cassandra-driver) (1.16.0)
Requirement already satisfied: geomet<0.3,>=0.1 in c:\users\varukish\appdata\local\anaconda3\lib\site-packages (from cassandra-driver) (0.2.1.post
1)
Requirement already satisfied: click in c:\users\varukish\appdata\local\anaconda3\lib\site-packages (from geomet<0.3,>=0.1->cassandra-driver) (8.0.
4)
Requirement already satisfied: colorama in c:\users\varukish\appdata\local\anaconda3\lib\site-packages (from click->geomet<0.3,>=0.1->cassandra-dri
ver) (0.4.6)
Note: you may need to restart the kernel to use updated packages.
```

In [2]:
```
import cassandra
print(cassandra.__version__)
```
```
3.28.0
```

In [5]:
```
from cassandra.cluster import Cluster
from cassandra.auth import PlainTextAuthProvider
import json

# This secure connect bundle is autogenerated when you donwload your SCB,
# if yours is different update the file name below
cloud_config= {
  'secure_connect_bundle':'secure-connect-cassandra-demo.zip'
}
# This token json file is autogenerated when you donwload your token,
# if yours is different update the file name below
with open("varun.gamer.pro@gmail.com-token.json") as f:
    secrets = json.load(f)

CLIENT_ID = secrets["clientId"]
CLIENT_SECRET = secrets["secret"]

auth_provider = PlainTextAuthProvider(CLIENT_ID, CLIENT_SECRET)
cluster = Cluster(cloud=cloud_config, auth_provider=auth_provider)
session = cluster.connect()

row = session.execute("select release_version from system.local").one()
if row:
  print(row[0])
else:
  print("An error occurred.")
```
```
4.0.0.6816
```

In [7]:
```
try:
    query = "use employee_keyspace"
    session.execute(query)
    print("Inside the keyspace_test")
except Exception as err:
    print("Exception occured while using key_space",err)
```
```
Inside the keyspace_test
```

In [8]:
```
# Command to create a table inside a KEyspace
try:
    query = """create table employee(
                emp_id int,
                emp_name varchar,
                emp_salary int,
                emp_dept varchar,
                emp_email varchar,
                emp_phone varchar,
                primary key (emp_id, emp_dept)
            )
            """
    session.execute(query)
    print("Table created inside the keyspace")
except Exception as err:
    print("Exception Occured while creating the table : ",err)
```
```
Table created inside the keyspace
```

In [9]:
```
# Alter the table in cassandra to drop a column
try:
    query = "alter table employee drop emp_email"
    session.execute(query)
    print("Column dropped successfully !!")
except Exception as err:
    print("Exception Occured while dropping the column: ",err)
```
```
Column dropped successfully !!
```

In [10]:
```
# Alter the table in cassandra to add a new column
try:
    query = "alter table employee add emp_email text"
    session.execute(query)
    print("Column added successfully !!")
except Exception as err:
    print("Exception Occured while adding the column: ",err)
```
```
Column added successfully !!
```

In [11]:
```
# Insert data into cassandra table
try:
    query = "insert into employee(emp_id, emp_name, emp_salary, emp_dept, emp_email, emp_phone) values(1, 'Shashank', 10000, 'Software', 'abc.gmail.
    session.execute(query)
    print("Record inserted successfully !!")
except Exception as err:
    print("Exception Occured while inserting the data into table: ",err)
```
```
Record inserted successfully !!
```

In [12]:
```
# Insert data into cassandra table
try:
    query = "insert into employee(emp_id, emp_name, emp_salary, emp_dept, emp_email, emp_phone) values(2, 'Rahul', 20000, 'IT', 'xyx.gmail.com','+91
    session.execute(query)
    print("Record inserted successfully !!")
except Exception as err:
    print("Exception Occured while inserting the data into table: ",err)
```
```
Record inserted successfully !!
```

In [13]:
```
# Insert data into cassandra table
try:
    query = "insert into employee(emp_id, emp_name, emp_salary, emp_dept, emp_email, emp_phone) values(3, 'Sunny', 22000, 'HR', 'klm.gmail.com','+91
    session.execute(query)
    print("Record inserted successfully !!")
except Exception as err:
    print("Exception Occured while inserting the data into table: ",err)
```
```
Record inserted successfully !!
```

In [14]:
```
# Insert data into cassandra table
try:
    query = "insert into employee(emp_id, emp_name, emp_salary, emp_dept, emp_email, emp_phone) values(4, 'Vishal', 30000, 'Software', 'mno.gmail.co
    session.execute(query)
    print("Record inserted successfully !!")
except Exception as err:
    print("Exception Occured while inserting the data into table: ",err)
```
```
Record inserted successfully !!
```

In [15]:
```
# Select query on cassandra table
try:
    query = "select * from employee"
    result = session.execute(query)
    for row in result:
        print(row)
except Exception as err:
    print("Exception Occured while selecting the data from table: ",err)
```
```
Row(emp_id=1, emp_dept='Software', emp_email='abc.gmail.com', emp_name='Shashank', emp_phone='+91 768467474', emp_salary=10000)
Row(emp_id=2, emp_dept='IT', emp_email='xyx.gmail.com', emp_name='Rahul', emp_phone='+91 908467474', emp_salary=20000)
Row(emp_id=4, emp_dept='Software', emp_email='mno.gmail.com', emp_name='Vishal', emp_phone='+91 600467474', emp_salary=30000)
Row(emp_id=3, emp_dept='HR', emp_email='klm.gmail.com', emp_name='Sunny', emp_phone='+91 800067474', emp_salary=22000)
```

In [16]:
```
try:
    query = "select emp_dept, emp_email,emp_salary from employee"
    result = session.execute(query)
    for row in result:
        print(row)
except Exception as err:
    print("Exception Occured while selecting the data from table: ",err)
```
```
Row(emp_dept='Software', emp_email='abc.gmail.com', emp_salary=10000)
Row(emp_dept='IT', emp_email='xyx.gmail.com', emp_salary=20000)
Row(emp_dept='Software', emp_email='mno.gmail.com', emp_salary=30000)
Row(emp_dept='HR', emp_email='klm.gmail.com', emp_salary=22000)
```

In [17]:
```
try:
    query = "select emp_dept, emp_email,emp_salary from employee"
    result = session.execute(query)
    for row in result:
        print("Emp_dept: ",row[0])
        print("emp_email: ",row[1])
        print("emp_salary: ",row[2])
except Exception as err:
    print("Exception Occured while selecting the data from table: ",err)
```
```
Emp_dept:  Software
emp_email:  abc.gmail.com
emp_salary:  10000
Emp_dept:  IT
emp_email:  xyx.gmail.com
emp_salary:  20000
Emp_dept:  Software
emp_email:  mno.gmail.com
emp_salary:  30000
Emp_dept:  HR
emp_email:  klm.gmail.com
emp_salary:  22000
```

In [18]:
```
try:
    query = "select * from employee where emp_id=2"
    result = session.execute(query)
    for row in result:
        print(row)
except Exception as err:
    print("Exception Occured while selecting the data from table: ",err)
```
```
Row(emp_id=2, emp_dept='IT', emp_email='xyx.gmail.com', emp_name='Rahul', emp_phone='+91 908467474', emp_salary=20000)
```

In [19]:
```
# Select query for specific columns in cassandra table and how to access from Row object
# Doesn't work
try:
    query = "select emp_id, emp_name from employee"
    result = session.execute(query)
    # option 2
    for row in result:
        row = dict(row)
        print(row)
        print("Emp ID : ", row['emp_id'])
        print("Emp Name : ", row['emp_name'])
except Exception as err:
    print("Exception Occured while selecting the data from table: ",err)
```
```
Exception Occured while selecting the data from table:  cannot convert dictionary update sequence element #0 to a sequence
```

In [20]:
```
# Write a query to get total count of employees
try:
    query = "select count(*) from employee"
    result = session.execute(query)
    row = result.one()
#     print(row)
    print(row[0])
except Exception as err:
    print("Exception Occured while selecting the data from table: ",err)
```
```
4
```

In [21]:
```
# Write a query to get total count of employees
try:
    query = "select max(emp_salary) from employee"
    result = session.execute(query)
    row = result.one()
#     print(row)
    print(row[0])
except Exception as err:
    print("Exception Occured while selecting the data from table: ",err)
```
```
30000
```

In [22]:
```
# Write a query to filter data from cassandra table or how to use where clause
# Rules for where clause - It can be used effectively with high performance for given below type of columns
# 1.) Partition Key (Single or Composite)
                #OR
# 2.) if Cluster column  used in where clause then it should be with Partition Key
                #OR
# 3.) A column on which we have applied the index
                #OR
# 4.) A column which is not part of partition key or index column or clustering column then we can use where clause but we have to
# use keyword ALLOW FILTERING - it will be a super slow performance when data volume is very high

try:
    query = "select * from employee where emp_name='Shashank' ALLOW FILTERING"
    result = session.execute(query)
    row = result.one()
    print(row)
except Exception as err:
    print("Exception Occured while selecting the data from table: ",err)
```
```
Row(emp_id=1, emp_dept='Software', emp_email='abc.gmail.com', emp_name='Shashank', emp_phone='+91 768467474', emp_salary=10000)
```

In [23]:
```
# where clause for Partition key only or Rule no -1

try:
    query = "select * from employee where emp_id=2"
    result = session.execute(query)
    row = result.one()
    print(row)
except Exception as err:
    print("Exception Occured while selecting the data from table: ",err)
```
```
Row(emp_id=2, emp_dept='IT', emp_email='xyx.gmail.com', emp_name='Rahul', emp_phone='+91 908467474', emp_salary=20000)
```

In [24]:
```
# where clause for Clustering key only or Rule no - 2

try:
    query = "select * from employee where emp_dept='Software' and emp_id=1"
    result = session.execute(query)
    row = result.one()
    print(row)
except Exception as err:
    print("Exception Occured while selecting the data from table: ",err)
```
```
Row(emp_id=1, emp_dept='Software', emp_email='abc.gmail.com', emp_name='Shashank', emp_phone='+91 768467474', emp_salary=10000)
```

In [25]:
```
# We have another composite partition key

priamry key ( (employeeid, fname) , dept )

select * from employee where employeeid = 'x' and dept = 'software' and fname = 'shashank'  # Wrong query

select * from employee where dept = 'software' and employeeid = 'x' and fname = 'shashank'   # Correct query
```
```
  Cell In[25], line 3
    priamry key ( (employeeid, fname) , dept )
            ^
SyntaxError: invalid syntax
```

In [26]:
```
# Group by in cassandra  -  Allowed for all columns which are part of Primary Key
# Follow given below rules
# Rule - 1 : Use only partition key in the group by
                #OR
# Rule - 2 : if Cluster key column is used then follow the actual declared sequence in the primary key
try:
    query = "select emp_id, sum(emp_salary) as sum_salary from employee group by emp_id"
    result = session.execute(query)
    row = result.one()
    for row in result:
        print("Emp ID : ", row[0])
        print("Sum Of Salary : ", row[1])
except Exception as err:
    print("Exception Occured while selecting the data from table: ",err)
```
```
Emp ID :  1
Sum Of Salary :  10000
Emp ID :  2
Sum Of Salary :  20000
Emp ID :  4
Sum Of Salary :  30000
Emp ID :  3
Sum Of Salary :  22000
```

In [28]:
```
# Group by in cassandra
# Rule - 2 : if Cluster key column is used then follow the actual declared sequence in the primary key
try:
    query = "select emp_id, emp_dept, sum(emp_salary) as sum_salary from employee group by emp_id,emp_dept"
    result = session.execute(query)
    row = result.one()
    for row in result:
        print("Emp ID : ", row[0])
        print("Emp Dept : ", row[1])
        print("Sum Of Salary : ", row[2])
except Exception as err:
    print("Exception Occured while selecting the data from table: ",err)
```
```
Emp ID :  1
Emp Dept :  Software
Sum Of Salary :  10000
Emp ID :  2
Emp Dept :  IT
Sum Of Salary :  20000
Emp ID :  4
Emp Dept :  Software
Sum Of Salary :  30000
Emp ID :  3
Emp Dept :  HR
Sum Of Salary :  22000
```

In [ ]: