Universidad Nacional Autónoma de México

Instituto de Investigaciones en Matemáticas Aplicadas y Sistemas

Bases de Datos No Estructurados

Práctica 2: Cassandra

Estudiantes:

-López Sánchez Misael

-Tapia López José de Jesús

Objetivo

- Desarrollar un sistema de consulta de libros en la que multiples usuarios puedan introducir puntuaciones respecto a que tanto les ha agradado un libro en general y poder realizar consultas sobre gustos literarios.
- Desarrollar una plataforma de libros.
- Debe incluir producto, cliente y califiación.
- Cada cliente debe tener información asociada (nombre, pais, membresía, etc).
- Además, cada cliente debe poder elegir una categoría para cada uno de sus libros (fantasía, misterio, etc).
- Todos deben participar como clientes

Introducción.

¿Qué es Apache Cassandra?

Cassandra es un sistema de código abierto distribuido de gestión de base de datos diseñado para manejar grandes cantidades de datos a través de muchos servidores de conveniencia, proporcionando una alta disponibilidad sin ningún punto único de fallo, ofrece un nivel muy elevado de compatibilidad para los clústers que abarcan múltiples centros de datos, con la replicación asíncrona sin muestreo, lo que permite operaciones de baja latencia para todos los clientes es una base de datos tipo $NoSQL$ distribuida y basada en un modelo de almacenamiento de 'clave'-'valor' escrita en Java.

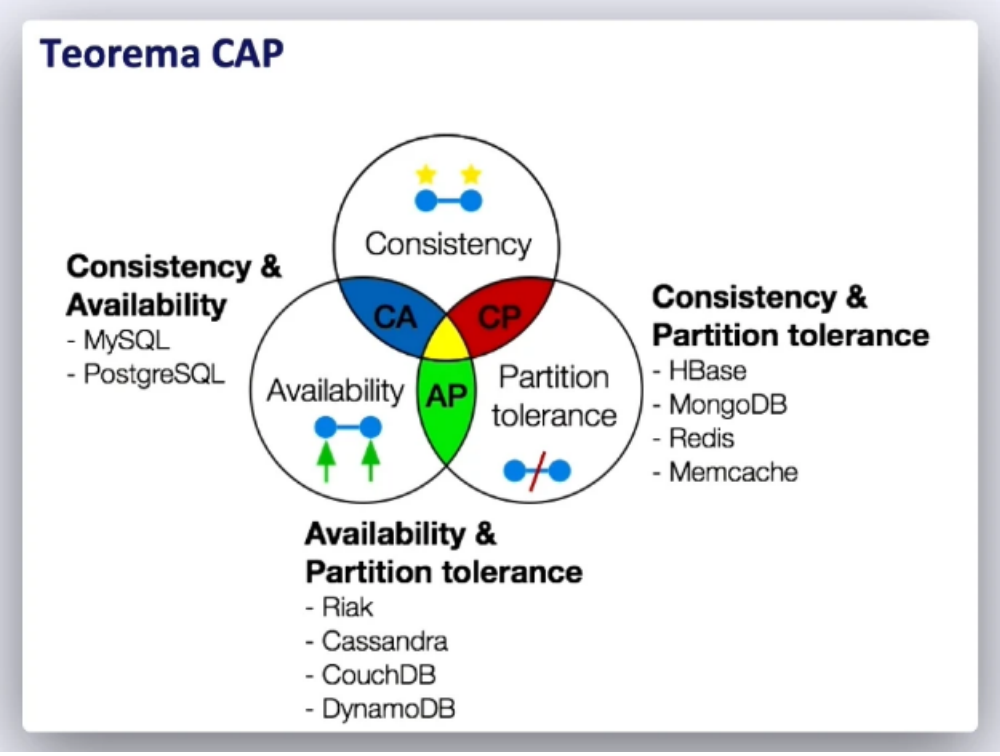
El modelo de datos de Cassandra consiste en partiocionar las filas, que son reorganizadas en tablas. Las clves primarias de cada tabla tiene un primer componente; que es la $partition\ key$, y un un segundo componete; $que\ es\ el\ cluster\ key$.

- La $partition\ key$ es la $primary\ key$ en caso de ser simple. Toda la información que se agrupe bajo un mismo renglón será guardada en un mismo nodo del clúster.

- La *cluster key* existe cuando la *primary key* está compuesta, y esta es desde la segunda parte en adelante, la cual se tomará para formar las familias de columnas, cada valor se agrupará bajo esos grupos, los cuales se mantendrán ordenados en el almacenamiento.

Dentro de una partición las filas son agrupadas por columnas restantes de la clave.

Cassandra no soporta *Joins* o *subconsultas* , sino que se enfatiza en la desnormalización a través de características como colecciones.

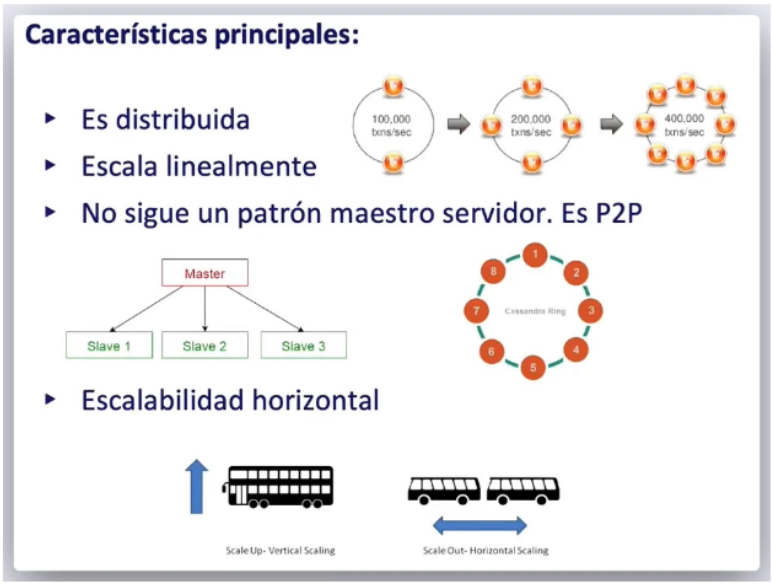


Características de Cassandra.

Apache Cassandra ofrece una serie de beneficios que optimizan el potencial de las aplicaciones online como:

- Arquitectura escalable.
- Diseño activo de inicio a fin pues los nodos se pueden escribir y leer.
- Rendimiento a escala líneal pues se puede añadir nuevos nodos sin tener que frenar el ritmo.
- Disponibilidad continua.
- Detección de fallos y recuperación transparente pues los nodos pueden ser fácilmente restaurados.
- Modelo flexible y dinámico.
- protección de datos solida.
- CQL (Lenguaje de Consulta Cassandra) el cual es similar a SQL que consigue que la transición de una base de datos relacional sea muy sencilla.

Características de Apache Cassandra



Desarrollo

Instalamos

Para comenzar primero es necesario instalar previamente Apache Cassandra en nuestro sistema operativo, para esto descargamos casandra desde el portal www-us.apache.org. Recordemos que primero debemos tener ciertos prerrequisitos como tener instalado el jdk-8** de Java, pues Cassandra esta desarrollado en este lenguaje; y posteriormente instalar el driver que funcionará de conexión entre **Cassandra** y **Python** con el siguiente comando:

- pip install cassandra-driver==3.22 --force-reinstall --upgrade

In [1]:

```
#pip install cassandra-driver
```

Verificamos la instalación

Checamos que la instalación se haya completado y revisamos la versión que tenemos de Cassandra que en este caso es la 3.22.0

In [1]:

```
import cassandra
print(cassandra.__version__)
```

```
3.22.0
```

Conexión a Cassandra

Antes de que podamos comenzar a ejecutar consultas en un clúster Cassandra, necesitamos configurar una instancia de Cluster. Como su nombre lo indica, normalmente tendrá una instancia de Cluster para cada clúster de Cassandra con el que desee interactuar.

La forma más sencilla de crear un clúster es así:

In [2]:

```
#from cassandra.cluster import Cluster
#cluster = Cluster()
```

Esto intentará conectarse a una instancia de Cassandra en su máquina local (127.0.0.1). También puede especificar una lista de direcciones IP para los nodos en su clúster:

In [3]:

```
#cluster = Cluster(['192.168.0.1', '192.168.0.2'])
```

El conjunto de direcciones IP que pasamos al Cluster es simplemente un conjunto inicial de puntos de contacto. Después de que el controlador se conecte a uno de estos nodos, descubrirá automáticamente el resto de los nodos en el clúster y se conectará a ellos, por lo que no necesita enumerar cada nodo en su clúster.

Si necesita usar un puerto no estándar, usar SSL o personalizar el comportamiento del controlador de alguna otra manera, esta es la forma para hacerlo:

In [4]:

```
#cluster = Cluster(['192.168.0.1', '192.168.0.2'], port=..., ssl_context=...)
```

Instanciar un clúster en realidad no nos conecta a ningún nodo. Para establecer conexiones y comenzar a ejecutar consultas, necesitamos una sesión, que se crea llamando a **cluster.connect()**. El método **connect()** toma una *keyspace* como argumento opcional que establece la *keyspace* predeterminado para todas las consultas realizadas a través de esa sesión. Sin embargo, en nuestro caso no le colocamos un argumento pues posteriormente crearemos la keyspace que usaremos para nuestra tabla de la práctica

In [1]:

```
from cassandra.cluster import Cluster

#Realizamos la conexión entre Python y el Clúster de Cassandra
cluster= Cluster()
session = cluster.connect()
```

Creamos una *keyspace* llamada uprofile donde vamos a crear nuestra tabla que usaremos para esta práctica:

In [2]:

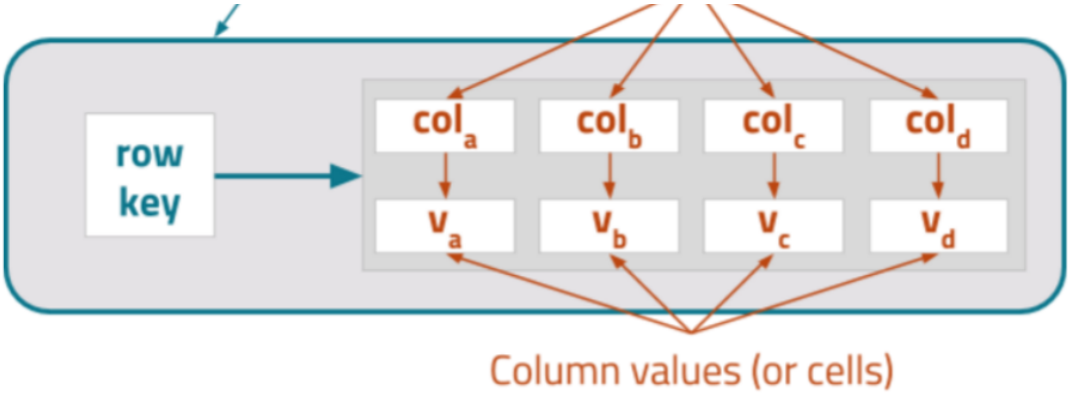
```
#session.execute('CREATE KEYSPACE IF NOT EXISTS uprofile WITH replication =
#                {\'class\': \'NetworkTopologyStrategy\', \'datacenter1\' : \'1\' }
```

Diseño de la base de datos.

Creamos el diseño de la base de datos de nuestros libros donde los campos a utilizar serán:

- membresia - Integer
- nombre_cliente - Text
- pais - Text
- nombre_libro - Text
- categoria - Text
- puntuacion - Float

Por llave primaria tomamos los campos **categoria** y **nombre_libro, membresia** como *partition key* y *cluster key* respectivamente; pues al momento de realizar consultas lo principal que nos gustaria conocer son los libros que se estan consultando y el individuo de la consulta.



Ahora bien, para empezar con el desarrollo de la práctica empezamos con la creación de la tabla:

In [7]:

```
#session.execute('DROP TABLE uprofile.plataforma_de_libros');
```

In [9]:

```
# Creamos la tabla

session.execute('CREATE TABLE uprofile.plataforma_de_libros(membresia int,
                'nombre_cliente text,
                'pais text, nombre_libro text, categoria text, puntuacion float,
                'PRIMARY KEY (categoria,nombre_libro,membresia))');
```

Realizamos más de 50 registros a la base de datos donde se cuentan con datos de diferentes usuarios de diferentes paises que han leido libros en común y cada uno de ellos les ha dado una calificación diferente, por lo que buscaremos después buscar a aquellos usuarios que tienen gustos literarios comunes.

In [10]:

```
insertar_datos =
session.prepare("INSERT INTO  uprofile.plataforma_de_libros (membresia,
                                'nombre_cliente, pais, nombre_libro, categoria,
                                'puntuacion)
                                'VALUES (?,?,?,?,?,?)")
```

In [11]:

```
# Le agregamos unos registros...
insertar_datos = session.prepare("INSERT INTO  uprofile.plataforma_de_libros (
                                 'membresia,
                                 'nombre_cliente, pais, nombre_libro,
                                 'categoria, puntuacion)
                                 'VALUES (?,?,?,?,?,?)")
```

Out[11]:

```
<cassandra.cluster.ResultSet at 0x7fc660107b38>
```

Realizamos unos registros a manera de ejemplo para ver como es que se insertan en la base de datos, en el dócumento con el código vienen los registros completos que son aproximadamente más de 50

In [ ]:

```
session.execute(insertar_datos, [312284510,'Armando Aguilar Léon',
                                 'México','Rayuela',
                                 'Antinovela',8])
session.execute(insertar_datos, [315332160,'Alfredo Aguilar Valdez'
                                 ,'USA','La Biblia','Religión',7])
session.execute(insertar_datos, [315023046,'Jonhattan Alanis Blancarte',
                                 'España','Gilgamesh','Epopeya',6])
session.execute(insertar_datos, [314026161,'Fernanda Aquino Cruz',
                                 'Colombia','Divina comedia','Epopeya',8])
session.execute(insertar_datos, [315001091,'Epifania Avendaño Montesinos',
                                 'México',
                                 'Orgullo y Prejuicio','Drama ',10])
session.execute(insertar_datos, [315232905,'Yazmín Baltazar Saucedo',
                                 'Francia','Decamerón','Cuentos',10])
session.execute(insertar_datos, [315138670,'María Bolon Sánchez','China',
                                 'El extranjero','Policial',9])
session.execute(insertar_datos, [314129420,'Patricia Caire Ascencio',
                                 'Rusia',
                                 'Don Quijote de la Mancha','Aventura',8])
session.execute(insertar_datos, [316294364,'Valeria Camacho Ávila',
                                 'Rusia','Harry Potter','Ficción',10])
session.execute(insertar_datos, [315042416,'Joanna Carmona Lara',
                                 'India','El señor de los anillos','Ficción',10])
session.execute(insertar_datos, [315200919,'Omar Castañeda Díaz',
                                 'Argentina','La Iliada','Epopeya',5])
```

Aplicación

La aplicación funciona como un sitema clasificador de libros, donde cada usuario que ha leido un determinado libro puede darle una clasificación en una escala nominal del 1 al 10 de qué tanto le ha gustado el libro.

Al ingresar a la aplicación del programa, al usuario se le presenta un menú con las siguientes opciones:

- Ingresar un nuevo registro
- Obtener todos los registros de la tabla
- Obtener todos los registros de un cliente
- Obtener la cantidad de libros leídos de un cliente
- Obtener la cantidad de libros leídos de un cliente por categoría
- Obtener la categoría preferida de un usuario
- Obtener los clientes que más disfrutaron de un libro
- Obtener los mejores libros de una categoría dada
- Obtener los libros favoritos de un usuario
- Estadisticas generales
- Salir

Lo que nos interesa de esta parte es, al momento de realizar consultas sobre la base de datos, descubrir aquellos usuarios que han leido libros en común y ver que calificación le ha dado cada uno, así para que cada usuario nuevo al leer un libro pueda consultar la información que otros usuarios han dejado de ese libro y ver si el libro es bueno o malo de acuerdo a sus gustos clasificados por categoria.

En la siguiente parte del código hemos puesto algunas líneas de código, que anteriormente lo habíamos usado, para facilitarnos la visualización de la aplicación.

In [1]:

```
from cassandra.cluster import Cluster
import os
import pandas as pd
from IPython.display import display

#Realizamos la conexión entre Python y el Clúster de Cassandra
#cluster = Cluster(['localhost'],port=6379)
cluster= Cluster()
session = cluster.connect()

insertar_datos = session.prepare("INSERT INTO  uprofile.plataforma_de_libros (
                                 'membresia,
                                 'nombre_cliente, pais, nombre_libro,
                                 'categoria, puntuacion)
                                 'VALUES (?,?,?,?,?,?)")

#funciones auxiliares para ingresar datos por consola en Python.
os.system('cls')
clear = lambda:os.system('cls')

#Obs: La aplicación no acentos comas así que al registrarlas en Redis se deforman.

def menu_aplication():
    """
    @Author: Jose de Jesus Tapia Lopez, Misael Lopez Sanchez
    summary: Menú que imprime la interfaz gráfica para ser puente entre Python
    y Cassandra
    Parameters: NULL
    Returns: Pantalla de la intefaz gŕafica
    """
    print('+'*10,"PLATAFORMA DE LIBROS", '+'*10, '\n')
    print('+'*20," MENU ", '+'*20, '\n')
    print("1.- Ingresar nuevo registro")
    print("2.- Obtener todos los registros de la tabla")
    print("3.- Obtener todos los registros de un cliente")
    print("4.- Obtener la cantidad de libros leídos de un cliente")
    print("5.- Obtener la cantidad de libros leídos de un cliente por categoría")
    print("6.- Obtener la categoría preferida de un usuario")
    print("7.- Obtener los clientes que más disfrutaron de un libro")
    print("8.- Obtener los mejores libros de una categoría dada")
    print("9.- Obtener los libros favoritos de un usuario")
    print("10.- Estadísticas generales")
    print("11.- Salir")
    print("")


def menu_estadisticas():
    """
    @Author: Jose de Jesus Tapia Lopez, Misael Lopez Sanchez
    summary: Menu auxiliar para que el usuario pueda consultar
    algunas otras estadisticas
    generales extras
    Parameters: NULL
    Returns: Pantalla de la intefaz gŕafica

    """
    print()
    print('1.- Registros totales por país')
    print('2.- Libros más leidos ')
```

```
    print('3.- Puntuaciones de usuarios sobre libros en especifico ')
    print('4.- Regresar')
    print()

def estadisticas():
    """
    @Author: Jose de Jesus Tapia Lopez, Misael Lopez Sanchez
    summary: Funcion que realiza operaciones básicas a insertar en
    la aplicacion general opción 6
    Parameters: Valor de la consulta a generar
    Returns: valor de la consulta especificada

    """
    while True:
        print('-'*5, "Estadísticas generales", '-'*5, '\n')
        menu_estadisticas()
        opEst = input('Ingrese una opción >> ')
        print()

        if opEst == '1':
            # Devuelve el número total de registros que se han realizado por pais
            consultaPais  = session.execute('select pais,membresia,
                                            'nombre_cliente
                                            'from uprofile.plataforma_de_libros;')
#Guardamos valores en DataFrame para poder trabajar consultas desde pandas.
            pais = []
            membresias = []
            nombres = []
            for cons in consultaPais:
                pais.append(cons.pais)
                membresias.append(cons.membresia)
                nombres.append(cons.nombre_cliente)
            #creamos el dataframe
            dfPais = pd.DataFrame({'usuarios':pais,
                                   'membresia':membresias,
                                   'nombre_cliente':nombres},
                                  columns=['usuarios',
                                           'membresia',
                                           'nombre_cliente'])
            mipais = input('Introduce el pais: ')
            dl = dfPais[dfPais['usuarios'] == mipais ]
            #filtramos valores por país
            n = len(dl)
            print(f'Desde {mipais} el número total de registros son:', n)
            print(dl)

        elif opEst == '2':
#Devuelve los libros más leidos de forma descendente
            consultaLib = session.execute('select nombre_libro,
                                          'count(nombre_libro) as "n"
                                          'from uprofile.plataforma_de_libros
                                          'group by categoria,
                                          'nombre_libro;')
#gurardamos valores en listas para después trabajar en DataFrames
            nombres = []
            count = []
            for cons in consultaLib:
                nombres.append(cons.nombre_libro)
                count.append(cons.n)
            #Creamos el dataframe
            dfFrec = pd.DataFrame({'nombre_libro':nombres,
```

```
                                    'Frecuencia':count},
                                   columns=['nombre_libro',
                                            'Frecuencia'])
            dfFrec = dfFrec.sort_values(by=['Frecuencia'],
                                        ascending=[False]).head(3)
            #mostramos los 3 más leidos por categoria
            print("Los tres libros más populares son:")
            print(dfFrec)

        elif opEst == '3':
#Devuelve las puntuaciones que usuarios en general han hecho sobre los libros
            nomLibro = str(input("Ingresa el nombre del libro: "))
            texto = "'"+nomLibro+"'"
            #conctenamos texto para poder realizar la consulta.

            consultaH = session.execute('select membresia,nombre_cliente,
                                        'nombre_libro,
                                        'puntuacion
                                        'from uprofile.plataforma_de_libros
                                        'where
                                        'nombre_libro ='+ str(texto) +'
                                        'allow filtering;')
            #guardamos todo en listas para trabajar con data Frames
            membresias = []
            nombres = []
            puntuacion = []
            clientes = []
            for cons in consultaH:
                membresias.append(cons.membresia)
                nombres.append(cons.nombre_libro)
                puntuacion.append(cons.puntuacion)
                clientes.append(cons.nombre_cliente)

            #Creamos dataFrame

            dfLib = pd.DataFrame({'membresia':membresias,
                                  'nombre_cliente':clientes,
                                  'nombre_libro':nombres,
                                  'puntuacion':puntuacion},
                                 columns=['membresia','nombre_cliente',
                                          'nombre_libro','puntuacion'])
            print(dfLib)


        elif opEst == '4':
            break;

        else:
            print("Opción no válida. Por favor, vuelva a intentarlo.")


        input("Presione ENTER para continuar.")
        print()


# Iniciamos con la interfaz grafica y todo su desarrollo
while True:
    # Inicializamos llamando al menu y eligiendo una opción.
    menu_aplication()
    opMenu = input("Ingrese una opción >> ")
    print()
```

```
    clear()

    if opMenu == '1': #Ingresar un nuevo registro.
    # En este caso, el usuario va a introducir los datos que
    # se le van a agregar a la tabla: nombre_cliente, nombre_libro,
    # membresia, pais, categoria
        print('-'*5," Ingresar Registro:", '-'*5, '\n')
        memb = int(input("Membresía del cliente: "))
        cliente = str(input("Nombre del cliente: "))
        pais = str(input("País del cliente: "))
        libro = str(input("Nombre del libro: "))
        categ = str(input("Cateogría del libro: "))
        punt = float(input("Puntuación del libro: "))

        # insertamos los datos en la tabla
        session.execute(insertar_datos, [memb,cliente,pais,libro,categ,punt])

    elif opMenu == '2': # Obtener toda la tabla
        print('-'*5, "Obtener todos los registros de la tabla", '-'*5, '\n')
        consulta = session.execute('select *
                                   'from uprofile.plataforma_de_libros;')
        membresias = []
        categorias = []
        clientes = []
        paises = []
        libros = []
        puntuaciones = []

        for cons in consulta:
            membresias.append(cons.membresia)
            categorias.append(cons.categoria)
            clientes.append(cons.nombre_cliente)
            paises.append(cons.pais)
            libros.append(cons.nombre_libro)
            puntuaciones.append(cons.puntuacion)

        #Creamos el data frame
        cons2 = pd.DataFrame({'membresia':membresias,
                              'nombre_cliente':clientes,
                              'pais':paises,
                              'nombre_libro':libros,
                              'categoria':categorias,
                              'puntuacion':puntuaciones},
                             columns=['membresia','nombre_cliente',
                                      'pais','nombre_libro',
                                      'categoria','puntuacion'])

        pd.set_option('display.max_rows', len(cons2))
        display(cons2)
        pd.reset_option('display.max_rows')



    elif opMenu == '3': # Obtener todos los registros de un usuario
        print('-'*5, "Obtener todos los registros de un cliente", '-'*5, '\n')
        memb = int(input("Ingrese la membresia del cliente: "))
        # Realizamos la consulta
        consulta = session.execute("select membresia,nombre_cliente,
                                   'pais,nombre_libro,
                                   'categoria,puntuacion
                                   'from uprofile.plataforma_de_libros
```

```
                                'where membresia = "+str(memb)+"
                                'allow filtering;")
        for cons in consulta:
            print(cons.membresia, cons.nombre_cliente, cons.pais,
                  cons.nombre_libro, cons.categoria, cons.puntuacion)

    elif opMenu == '4': #  Obtener la cantidad de libros leídos de un cliente
    print('-'*5, "Obtener la cantidad de libros leídos de un cliente", '-'*5, '\n')
        memb = int(input("Ingrese la membresia del cliente: "))
        # Realizamos la consulta
        consulta = session.execute("select membresia,nombre_cliente,pais,count(*)
                                'as libros_leidos
                                'from uprofile.plataforma_de_libros
                                'where membresia = "+str(memb)+"
                                'allow filtering;")
        for cons in consulta:
            print(cons.membresia, cons.nombre_cliente, cons.pais, cons.libros_leido

    elif opMenu == '5':
    #  Obtener la cantidad de libros leídos de un cliente por categoría
    print('-'*5, "Obtener la cantidad de libros leídos de un cliente por categoría"
        memb = int(input("Ingrese la membresia del cliente: "))
        # Realizamos la consulta
        consulta = session.execute("select membresia,
                                'nombre_cliente,pais,
                                'categoria, count(*) as
                                'libros_leidos_por_categoria
                                'from uprofile.plataforma_de_libros
                                'where membresia = "+str(memb)+"
                                'group by categoria  allow filtering;")
        for cons in consulta:
            print(cons.membresia, cons.nombre_cliente,
                  cons.pais, cons.categoria,
                  cons.libros_leidos_por_categoria)


    elif opMenu == '6':
        #Obtener la categoria preferida de un usuario
        print('-'*5, "Obtener la categoria preferida de un usuario", '-'*5, '\n')
        memb = int(input("Ingrese la membresia del cliente: "))
        # Realizamos la consulta
        consulta = session.execute('select membresia, nombre_cliente, categoria,
                                'avg(puntuacion) as "Promedio"
                                'from uprofile.plataforma_de_libros
                                'where membresia = '+str(memb)+'
                                'group by categoria allow filtering;')
    # Dado que en Cassandra no se pueden realizar consultas anidades, la consulta
    # anterior solo arroja los promedios de la categoria, entonces para mostrar
    # la categoria con mayor promedio en la puntuacion, guardamos los promedios
    # de las puntuaciones en una lista vacia y posteriormente mostramos
    # el que tiene el promedio más grande
        maximo = []
        for cons in consulta:
            maximo.append(cons.Promedio)
        consulta = session.execute('select membresia, nombre_cliente, categoria,
                                'avg(puntuacion) as "Promedio"
                                'from uprofile.plataforma_de_libros
                                'where membresia = '+str(memb)+'
                                'group by categoria allow filtering;')
        for cons in consulta:
            if cons.Promedio == max(maximo):
```

```
            print(cons.membresia, cons.nombre_cliente, cons.categoria, cons.Pro

    elif opMenu == '7':
        #Obtener los clientes que más disfrutaron de un libro
        print('-'*5, 'Obtener los clientes que más disfrutaron de un libro', '-'*5,
        libro = input('Ingrese el nombre del libro: ')
        # Encerramos el libro entre comillas
        libro = '\''+libro+'\''
        # Hacemos la consulta en Cassandra
        consulta = session.execute("select membresia,nombre_cliente,
                                   'nombre_libro, puntuacion
                                   'from uprofile.plataforma_de_libros
                                   'where nombre_libro = "+libro+" and
                                   'puntuacion > 8 allow filtering;")
        # Imprimimos la consulta
        for cons in consulta:
            print(cons.membresia, cons.nombre_libro,
                  cons.puntuacion)

    elif opMenu == '8':
        categ = input('Ingresa la categoría que te interesa conocer: ')
        print('-'*5, f'Los mejores libros para la categoría {categ} son: ', '-'*5,
        consulta4 = session.execute('select nombre_libro, categoria,
                                    'avg(puntuacion) as "catprom"
                                    'from uprofile.plataforma_de_libros
                                    'group by categoria, nombre_libro;')

        categoria = []
        nombre = []
        catProm = []
        for cons in consulta4:
            categoria.append(cons.categoria)
            nombre.append(cons.nombre_libro)
            catProm.append(cons.catprom)

        #Creamos el dataframe
        df = pd.DataFrame({'categoria':categoria,'nombre_libro':nombre,
                  'catprom':catProm},columns=['categoria','nombre_libro','catprom']

        df2 = df[df['categoria'] == categ]
        df3=df2.sort_values(by=['catprom'],ascending=[False])
        print(df3)

    elif opMenu == '9': #Obtener los libros favoritos de un usuario
        print('-'*5, "Obtener los libros favoritos de un usuario", '-'*5, '\n')
        memb = int(input("Ingresa la membresia del cliente: "))
        # Realizamos la consulta
        consulta = session.execute("select membresia, nombre_cliente, nombre_libro
                                   'from uprofile.plataforma_de_libros
                                   'where membresia = "+str(memb)+" and
                                   'puntuacion > 8 allow filtering;")
        for cons in consulta:
            print(cons.membresia, cons.nombre_cliente, cons.nombre_libro)

    elif opMenu == '10':
        estadisticas()

    elif opMenu == '11':
            break;
    else:
        print("Opción no válida. Por favor, vuelva a intentarlo.")
```

```
    input("\n\n Presione ENTER para continuar.")
    print()

    clear()
```

```
++++++++++ PLATAFORMA DE LIBROS ++++++++++

++++++++++++++++++++  MENU  ++++++++++++++++++++

1.- Ingresar nuevo registro
2.- Obtener todos los registros de la tabla
3.- Obtener todos los registros de un cliente
4.- Obtener la cantidad de libros leídos de un cliente
5.- Obtener la cantidad de libros leídos de un cliente por categoría
6.- Obtener la categoría preferida de un usuario
7.- Obtener los clientes que más disfrutaron de un libro
8.- Obtener los mejores libros de una categoría dada
9.- Obtener los libros favoritos de un usuario
10.- Estadísticas generales
11.- Salir

Ingrese una opción >> 11
```

Funcionamiento del programa

A continuación mostramos el funcionamiento del programa de manera general por medio de imagenes de las salidas de cada una de las opciones donde se muestra el resultado de cada una de las consultas realizadas.

```
++++++++++ PLATAFORMA DE LIBROS ++++++++++

++++++++++++++++++++  MENU  ++++++++++++++++++++

1.- Ingresar nuevo registro
2.- Obtener todos los registros de la tabla
3.- Obtener todos los registros de un cliente
4.- Obtener la cantidad de libros leídos de un cliente
5.- Obtener la cantidad de libros leídos de un cliente por categoría
6.- Obtener la categoría preferida de un usuario
7.- Obtener los clientes que más disfrutaron de un libro
8.- Obtener los mejores libros de una categoría dada
9.- Obtener los libros favoritos de un usuario
10.- Estadísticas generales
11.- Salir

Ingrese una opción >> 1

-----  Ingresar Registro: -----

Membresía del cliente: 418004719
Nombre del cliente: José de Jesús Tapia López
País del cliente: México
Nombre del libro: Cuentos para pensar
Cateogría del libro: Cuentos
Puntuación del libro: 9


 Presione ENTER para continuar.
```

```
Ingrese una opción >> 2

----- Obtener todos los registros de la tabla -----
```

	membresia	nombre_cliente	pais	nombre_libro	categoria	puntuacion
0	111002674	Mauricio José Barreiro	India	El juego de Ender	Ficción	6.0
1	315284557	Yoselyn Juárez Casimiro	Italia	El juego de Ender	Ficción	6.0
2	673511509	Misael López Sánchez	España	Neuromante	Ficción	8.0
3	123456789	Alejandro Pimentel Alarcón	Francia	Cuentos para pensar	Cuentos	10.0
4	418004719	José de Jesús Tapia López	México	Cuentos para pensar	Cuentos	9.0
5	315232905	Yazmín Baltazar Saucedo	Francia	Decamerón	Cuentos	10.0
6	316281810	Angel Dimas Huesca	España	Las mil y una noches	Cuentos	6.0
7	314129420	Patricia Caire Ascencio	Rusia	Don Quijote de la Mancha	Aventura	8.0

```
Ingrese una opción >> 3

----- Obtener todos los registros de un cliente -----

Ingrese la membresia del cliente: 418004719
418004719 José de Jesús Tapia López México Cuentos para pensar Cuentos 9.0
418004719 José de Jesús Tapia López México Rayuela Antinovela 9.0
418004719 José de Jesús Tapia López México Carrie Ficción 6.0
418004719 José de Jesús Tapia López México Desde mi cielo Drama 9.0
418004719 José de Jesús Tapia López México Romeo y Julieta Drama 10.0
418004719 José de Jesús Tapia López México Ana Karenina Romance 8.0
418004719 José de Jesús Tapia López México La Biblia Religión 6.0


Ingrese una opción >> 4

----- Obtener la cantidad de libros leídos de un cliente -----

Ingrese la membresia del cliente: 418004719
418004719 José de Jesús Tapia López México 7


Ingrese una opción >> 5

----- Obtener la cantidad de libros leídos de un cliente por categoría -----

Ingrese la membresia del cliente: 418004719
418004719 José de Jesús Tapia López México Cuentos 1
418004719 José de Jesús Tapia López México Antinovela 1
418004719 José de Jesús Tapia López México Ficción 1
418004719 José de Jesús Tapia López México Drama 2
418004719 José de Jesús Tapia López México Romance 1
418004719 José de Jesús Tapia López México Religión 1


Ingrese una opción >> 6

----- Obtener la categoria preferida de un usuario -----

Ingrese la membresia del cliente: 418004719
418004719 José de Jesús Tapia López Drama 9.5


Ingrese una opción >> 7

----- Obtener los clientes que más disfrutaron de un libro -----

Ingrese el nombre del libro: Harry Potter
316294364 Harry Potter 10.0
```

```
Ingrese una opción >> 8

Ingresa la categoría que te interesa conocer: Drama
----- Los mejores libros para la categoría Drama son:  -----

   categoria            nombre_libro    catprom
29     Drama         Romeo y Julieta  10.000000
27     Drama          Desde mi cielo   9.000000
26     Drama               Decamerón   8.000000
28     Drama     Orgullo y Prejuicio   8.000000
25     Drama  Bajo la misma estrella   6.666667


Ingrese una opción >> 9

----- Obtener los libros favoritos de un usuario -----

Ingresa la membresia del cliente: 418004719
418004719 José de Jesús Tapia López Cuentos para pensar
418004719 José de Jesús Tapia López Rayuela
418004719 José de Jesús Tapia López Desde mi cielo
418004719 José de Jesús Tapia López Romeo y Julieta


Ingrese una opción >> 10

----- Estadísticas generales -----


1.- Registros totales por país
2.- Libros más leidos
3.- Puntuaciones de usuarios sobre libros en especifico
4.- Regresar

Ingrese una opción >> 1

Introduce el pais: España
Desde España el número total de registros son: 14
```

```
Introduce el pais: España
Desde España el número total de registros son: 14
   usuarios  membresia              nombre_cliente
2    España  673511509        Misael López Sánchez
6    España  316281810          Angel Dimas Huesca
12   España  673511509        Misael López Sánchez
15   España  404114851   Minerva Del Real Martínez
25   España  316063443      Susan Morales Gónzalez
35   España  316063443      Susan Morales Gónzalez
36   España  673511509        Misael López Sánchez
40   España  404114851   Minerva Del Real Martínez
44   España  404114851   Minerva Del Real Martínez
48   España  316063443      Susan Morales Gónzalez
49   España  673511509        Misael López Sánchez
66   España  404114851   Minerva Del Real Martínez
79   España  315023046  Jonhattan Alanis Blancarte
84   España  673511509        Misael López Sánchez
Presione ENTER para continuar.
```

```
----- Estadísticas generales -----


1.- Registros totales por país
2.- Libros más leidos
3.- Puntuaciones de usuarios sobre libros en especifico
4.- Regresar


Ingrese una opción >> 2

Los tres libros más populares son:
                nombre_libro  Frecuencia
33              Ana Karenina           4
42                 Gilgamesh           4
5   Don Quijote de la Mancha           4
Presione ENTER para continuar.


----- Estadísticas generales -----


1.- Registros totales por país
2.- Libros más leidos
3.- Puntuaciones de usuarios sobre libros en especifico
4.- Regresar

Ingrese una opción >> 3

Ingresa el nombre del libro: Desde mi cielo
   membresia              nombre_cliente    nombre_libro  puntuacion
0  123456789  Alejandro Pimentel Alarcón  Desde mi cielo        10.0
1  315138670         María Bolon Sánchez  Desde mi cielo         9.0
2  315232905     Yazmín Baltazar Saucedo  Desde mi cielo         9.0
3  418004719   José de Jesús Tapia López  Desde mi cielo         9.0
Presione ENTER para continuar.
```

Conclusiones

Al trabajar con Cassandra pudimos encontrar una base de datos potente en el momento de realizar lectura de datos, y principalmente resaltamos su principal cualidad, la cual es la **escalabilidad**. La base de datos que trabajamos era pequeña, con menos de 100 registros pero aún así eran los suficientes para poder trabajar y realizar consultas interesantes.

Cassandra se presenta como una buena solución a la problématica de los grandes bancos de datos, aún cuando su manera de modelar datos dista del modelo clásico relacional este posee herramientas intereesantes para administrar **Primary Keys** pues la primera gran diferencia es que en casandra la información se administra por:

- Nodos
- Primary Key
- Cluster Key

Lo cual nos permite clasificar la información de forma muy diferente a lo que se suele hacer con las bases de datos relacionales, pues los registros se van registrando por renglón, identificados por la **primary key** y ordenados por la **cluster key**. Una de las ventajas que tiene trabajar con Apache Cassandra es que podemos trabajar con más de un campo como **primary Key**, lo que dependerá del diseño de la base de datos.

La principal ventaja de Apache Cassandra es **la escalabilidad** pues podemos agregar al esquema de nuestra base de datos nuevas columnas sin la necesidad de redistribuir toda la estructura de nuestra Base de Datos, como pasaba con redis donde para poder agregar un nuevo nodo, teniamos que redistribuir toda la estructura del anillo.

Otra de las diferencias es que Apache Cassandra tiene su propio lenguaje de consulta de datos la cual es **CQL**, el cual nos provee de funciones bastante parecidas a las acostumbradas en **SQL**, con ligeras diferencias como por ejemplo que en CQL no existe el ORDER BY. Además, para poder usar la función GROOUP BY, necesariamente se tiene que realizar con la *partition key*, y no se puede con ningún otro campo, estas pueden ser ventajas o desventajas dependiendo del tipo de información que el usuario pretenda almacenar en la base de datos. En nuestro caso, como nosotros habíamos diseñado inicialmente, no habíamos considerado este punto; y como en una de las consultas que nos pedía la práctica teníamos que agrupar por categorías, es por eso que consideramos la columna *categoria* como *partition key*; y para que la primary key cumpliera con ser única, seleccionamos los atributos *nombre_libro, membresia* como la *cluster key*.

Como habíamos mencionado antes, dado que no se pueden realizar consultas anidadas, en el desarrollo del menú de la práctica nos hemos apoyado de las herramientas de Python para satisfacer los requerimientos de los objetivos.

Cassandra es una solución brillante para muchos casos de uso que podemos encontrar en el mundo del Big Data. Sin embargo, no es adecuada para alojar **data warehouse**.

Lo ideal antes de trabajar con esta base de datos es tener claro desde un principio el caso de uso y el tipo de consultas que realizaremos para diseñar de una manera coherente la base de datos, para poder manejar un gran volúmen de datos aprovechando las cualidades de esta base de datos distribuida.

Por sus características, Apache Cassandra es una de las Bases de datos de tipo columnar más famosas que hay, algunas de las aplicaciones que la usan son las siguientes:

Finalmente, podemos mencionar entonces algunas ventajas y desventajas:

Ventajas:

- Cassandra resuelve el problema de los sistemas distribuidos y escalables, y está diseñado para hacer frente a los desafíos de gestión de datos de las empresas modernas.
- Cassandra es un sistema descentralizado: no existe un único punto de falla, si está presente la configuración mínima requerida para el clúster: cada nodo del clúster tiene la misma función y cada nodo puede atender cualquier solicitud. Las estrategias de replicación se pueden configurar. Es posible agregar nuevos nodos al clúster de servidores de manera muy fácil. Además, si un nodo falla, los datos se pueden recuperar de algunos de los otros nodos (se puede ajustar la redundancia). Es especialmente adecuado para la implementación de múltiples centros de datos, redundancia, recuperación de fallas y recuperación ante desastres, con posibilidad de replicación en múltiples centros de datos.
- Cassandra tiene integración Hadoop, con soporte MapReduce, también para Apache Pig y Apache Hive.

Desventajas:

- No hay integridad referencial, no hay un concepto de conexiones JOIN en Cassandra.
- Las opciones de consulta para recuperar datos son muy limitadas.
- La clasificación de datos es una decisión de diseño; se puede hacer a través de una de las formas predefinidas; los datos pueden recuperarse en el mismo orden; eso es todo: no hay cosas como ORDER BY ni subconsultas.
- Diseño de base de datos diferente; en RDBMS pensamos primero en el modelado de datos y luego creamos consultas; aquí, pensamos primero en las consultas más comunes, y después de eso, los datos se modelan en torno a esas consultas.

Los modelos de base de datos NoSQL no pueden reemplazar completamente la tecnología RDBMS, pero la importancia de NoSQL crecerá debido a la escala, flexibilidad y facilidad de uso. Estamos tratando con más y más datos; queremos aplicaciones duraderas y tolerantes a fallas; queremos aplicaciones que escalen y aplicaciones que sean rápidas. Debido a todo esto, NoSQL nos rodeará cada vez más y definitivamente vale la pena explorar la tecnología.