Programación de Objetos Distribuidos
Trabajo Práctico Especial

Se cuenta con un subconjunto del dataset del **Censo Nacional de Población, Hogares y Viviendas 2010: Censo del Bicentenario**[1]. En un archivo CSV, cada una de las filas representa un habitante registrado en el censo. Los valores son los siguientes:

- **edad:** Cantidad de años cumplidos a la fecha de referencia del Censo
- **alfabetismo**: Sabe leer y escribir
    - 1: Si
    - 2: No
- **tipoVivienda:** Tipo de vivienda particular donde habita
    - 1: Casa
    - 2: Rancho
    - 3: Casilla
    - 4: Departamento
    - 5: Pieza en inquilinato
    - 6: Pieza en hotel familiar o pensión Local no construido para habitación Vivienda móvil
    - 7: Persona/s viviendo en la calle
- **nombreDepto**: Nombre del departamento donde habita.
- **nombreProv:** Nombre de la provincia donde habita.
- **hogarId:** Identificador del hogar donde habita.

Se requiere generar una aplicación de consola que utilice el modelo de programación MapReduce junto con el framework HazelCast y calcule las siguientes queries:

1. Cantidad de habitantes total del país agrupados de acuerdo a su edad en tres grupos :
    a. 0 - 14 años
    b. 15 - 64 años
    c. 65 años y más
2. El promedio de habitantes por vivienda para cada tipo de vivienda.
3. Los `n` departamentos con mayor índice de analfabetismo, donde el índice se calcula por el número total de habitantes analfabetos del departamento sobre el total de población del departamento, donde `n` provee el usuario.
4. Los departamentos de la provincia `prov` con una cantidad de habitantes menor a `tope`, donde `prov` y `tope` lo provee el usuario.
5. Los pares de departamentos que tienen la misma cantidad de cientos de habitantes.

Cada corrida de la aplicación realiza una de estas queries sobre los datos obtenidos a partir de un archivo CSV. El archivo CSV siempre debe ser un subconjunto del dataset mencionado, respetando el formato enunciado.

[1] Fuente: http://datar.noip.me/dataset/censo-2010-microdatos

La información de cuál es la query a correr, del path al archivo de texto y al archivo de salida y los posibles parámetros que cada query reciba, se recibe a través de argumentos de línea de comando al llamar a la aplicación.

Por ejemplo:

```
$> java -Daddresses=xx.xx.xx.xx;yy.yy.yy.yy -Dquery=1 -DinPath=censo.csv
-DoutPath=output.txt [queryParams] client.MyClient
```

donde

- `xx.xx.xx.xx;yy.yy.yy.yy` son las direcciones IP de los nodos,
- censo.csv es el archivo de entrada con los datos a procesar
- output.txt es el archivo de salida con los resultados de la query
- `[queryParams]`:
    - `-Dn=XX` para la query 3
    - `-Dprov=XX -Dtope=YY` para la query 4
    - Vacío para las otras queries

De esta forma,

```
$> java -Daddresses=10.6.0.1;10.6.0.2 -Dquery=3 -DinPath=censo.csv
-DoutPath=output.txt -Dn=100 client.MyClient
```

escribe en el archivo output.txt los 100 departamentos con mayor índice de analfabetismo, según los datos presentes en `censo.csv`, utilizando los nodos `10.6.0.1` y `10.6.0.2` para su procesamiento.

Los nombres de los mapas de Hazelcast a utilizar en la implementación deben comenzar con los números de legajo de los integrantes del grupo, para así evitar compartir los mapas y poder hacer pruebas de distintos grupos en simultáneo. Se debe utilizar la siguiente convención:

```
legajo1-legajo2-nombre
```

donde `nombre` queda a elección de los integrantes del grupo de `legajo1` y `legajo2`

La aplicación debe entonces correr la query y escribir en un archivo la respuesta a la query.

Ejemplos del archivo de respuesta de las queries:

- Query 1

```
0-14 = 25376
15-64 = 64160
65-? = 10464
```

- Query 2

```
0 = 15,16
1 = 3,42
2 = 3,97
3 = 3,92
4 = 2,39
5 = 2,48
6 = 1,79
7 = 2,47
8 = 3,00
9 = 1,67
```

- Query 3

```
Catan Lil = 0,40
Telsen = 0,25
Rinconada = 0,20
San Blas de los Sauces = 0,18
Tehuelches = 0,17
```

- Query 4

```
San Fernando = 972
Comandante Fernández = 312
General Güemes = 190
Libertador General San Martín = 156
Mayor Luis J. Fontana = 137
```

- Query 5

```
1300
Quilmes + San Fernando
Almirante Brown + Quilmes
Almirante Brown + San Fernando

1400
Merlo + General San Martín
```

La implementación debe respetar el formato de salida enunciado.

Para medir performance, se deberán imprimir en pantalla los *timestamp* de los siguientes momentos:

- Inicio de la lectura del archivo.
- Fin de lectura del archivo.
- Inicio de un trabajo MapReduce.

- Fin de un trabajo MapReduce (incluye la escritura del archivo de respuesta).

Todos estos momentos deben ser impresos en la salida luego de la respuesta con el timestamp en formato: `dd/mm/yyyy hh:mm:ss:xxxx` y deben ser claramente identificables.

Ejemplo de la salida en pantalla:

```
09/11/2016 14:43:09:0223 INFO  [main] client.MyClient (MyClient.java:76)
- Inicio de la lectura del archivo
09/11/2016 14:43:23:0011 INFO  [main] client.MyClient
(MyClient.java:173) - Fin de lectura del archivo
09/11/2016 14:43:23:0013 INFO  [main] client.MyClient (MyClient.java:87)
- Inicio del trabajo map/reduce
09/11/2016 14:43:23:0490 INFO  [main] client.MyClient
(MyClient.java:166) - Fin del trabajo map/reduce
```

Condiciones del trabajo práctico

- El trabajo práctico debe realizarse en grupos de a dos personas exclusivamente.
- Cada una de las opciones debe ser implementada como un job MapReduce que pueda correr en un ambiente distribuido utilizando un grid de hazelcast.
- Los componentes del job, clases del modelo, test y el diseño de cada elemento del proyecto queda a criterio del equipo, pero debe estar enfocado en:
    - Que funcione correctamente en un ambiente concurrente, MapReduce en hazelcast.
    - Que sea eficiente para un gran volumen de datos.
- En **campus.itba.edu.ar,** en la sección Contenido / TPE, se irán dejando subconjuntos del dataset del Censo con diferentes cantidades de habitantes para así poder pobar con diversos volúmenes de datos. Todos estos archivos deberán ser incluidos en el proyecto que se entrega y serán utilizados para realizar pruebas.

Se debe entregar

- El ***código fuente*** de la aplicación:
    - Utilizando el arquetipo de Maven utilizado en las clases.
    - Con una correcta separación de las clases de cliente y servidor.
- Un ***documento*** explicando:
    - Como preparar el entorno a partir del código fuente para ejecutar la aplicación en un ambiente con varios nodos.
    - Brevemente como se diseñaron los componentes de cada trabajo MapReduce, qué decisiones se tomaron y con qué objetivos. Además alguna alternativa de diseño que se evaluó y descartó, comentando el porqué.
    - El análisis de cada proceso corriendo en clusters variando la cantidad de nodos e indicando cuál sería la cantidad de nodos mejor para cada uno (analizando clusteres de hasta 6 nodos). Esa configuración el elegida será la utilizada contra un archivo a determinar por la cátedra el día de la presentación del TPE.

Entregas

- **Hasta el día 11/11/2016** se deben informar los nombres de los integrantes de cada equipo en la sección Discusiones / TPE de Campus ITBA
- **Antes del 23/11/2016 a las 18 hs** uno de los integrantes del equipo debe cargar el ***código fuente*** y el ***documento*** del trabajo práctico especial en la evaluación Entrega TPE localizada en la sección Contenido / TPE de Campus ITBA.
- **El día 23/11/2016 a las 18hs** cada grupo tendrá 15 minutos para configurar su entorno y mostrar la ejecución de la aplicación a la cátedra realizando dos corridas sobre una misma configuración de nodos a utilizar, con 2 archivos (datasets) diferentes a determinar por la cátedra. Se tomará nota de las respuestas y los tiempos de ejecución y esto será parte de la evaluación del trabajo. A criterio de la cátedra también se podrán realizar preguntas sobre la implementación de los mismo.
- **El día 30/11/2016** se publicarán las notas de los equipos en Campus ITBA. Si se requiere que el equipo corrija algún elemento del proyecto a modo de recuperar el trabajo, la nueva versión del mismo deberá ser entregada al fin de la clase de ese día.
- ***No se aceptarán entregas fuera de los días y horarios dados*** .

Sugerencia de investigación

Pueden resultar útiles para la resolución del trabajo práctico investigar los siguientes temas:

- Hazelcast Collator
- Hazelcast KeyPredicate