Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación
Primer Semestre de 2024

IIC 2440 – Procesamiento de Datos Masivos
MidTerm

Instrucciones. El examen tiene una duración de dos horas.

Resuelve cada uno de los problemas en una hoja separada. Las preguntas cortas generalmente tienen una mayor probabilidad de estar correctas.

Todas las preguntas tienen 6 puntos, la nota final se calcula como el promedio de los puntos de cada pregunta, más un punto base.

Las preguntas se hacen en voz alta; solo en los siguientes intervalos de tiempo:

- De 14:50 a 15:10
- De 16:00 a 16:20

Problema 1 Considera la siguiente modificación sobre el algoritmo A-priori.

El input sigue siendo un conjunto de canastas sobre elementos: el conjunto es $C_1, \ldots, C_n$, donde cada $C_i \subseteq E$. Se desean calcular todas las reglas de asociación que superen un soporte sn y una confianza c.

En la primera pasada, como en el algoritmo original, se calcula el soporte de cada conjunto $\{e\}$, para cada $e \in E$,. Pero además, hacemos lo siguiente: Para un entero N, sea $h : E \times E \to N$ una función de hash y B un arreglo de tamaño N. Para cada canasta C_i, para cada par e, e' de elementos en esa canasta, toma $h(e, e')$. Luego, incrementa en 1 la posición de B en $h(e, e')$ (osea, ejecuta $B[h(e, e')] += 1$).

Luego, en la segunda pasada, cuando toque computar el soporte de los conjuntos de tamaño dos, solo te concentrarás en los conjuntos $\{e, e'\}$ que cumplen las siguientes dos condiciones. Primero, tal como en el algoritmo original, que tanto $\{e\}$ como $\{e'\}$ tengan soporte mayor a sn. Adicionalmente (y esto es lo nuevo), solo vas a revisar el soporte de los conjuntos $\{e, e'\}$ en donde $B[h(e, e')] > sn$. Así, el algoritmo procede a computar el soporte solo de aquellos conjuntos $\{e, e'\}$ que cumplen ambas condiciones.

Todas las otras pasadas son iguales.

Discute las ventajas y desventajas de este algoritmo.

Problema 2 Te encuentras en una reunión en La Moneda, están discutiendo el nuevo sistema integrado de datos de Salud (SIS) y tu estás trabajando para uno de los equipos involucrados. En corto, todos los hospitales del país se han puesto de acuerdo para entregar todos sus datos al SIS. Esto lo haran disponibilizando una API desde donde se pueden descargar los datos; el SIS debe encargarse de tomar periódicamente estos datos desde estas APIs, y construir una infraestructura de datos desde donde se puedan realizar análisis como:

- Ver los hospitales con mayor o menor lista de espera.
- Ver los hospitales con mayor o menor disponibilidad de cama.
- Ver los hospitales con más variabilidad en su ocupación.

Entre varios otros requerimientos. Estás en esta reunión para planificar cómo van a armar esta base de datos integrada. Las propuestas de los analistas presentes son las siguientes:

La **analista 1** dice que lo que es importante es poder usar esos datos para el posterior análisis. Pero como aún no se sabe del todo que se quiere analizar, propone un almacén de datos (*data warehouse*) en PostgreSQL con pocas tablas, pero en donde las tablas tengan todas las columnas indexadas. Esto lo propone porque cree que así, cualquier cosa que hagan en SQL va a correr rapidísimo.

El **analista 2** opina que hay que llevar todo a la nube. Propone instalar Big Query, llevar todos los datos a un gran almacen en Google Cloud y que todas las consultas se deben hacer sobre esta base de datos. La frase que más repitió fue "*siempre hay que usar BigQuery para todo*".

La **analista 3** comenta que hay dos partes: por un lado, el SIS debe ir cargando a algún sistema los datos desde cada hospital, consultando la API día a día. Por otro lado, cree que solo algunos de estos datos van a ser importantes. Por esto, propone no hacer nada hasta que se sepa con exactitud que tipo de análisis o consultas quieren hacer.

Ahora es tu turno de hablar!

- Explícale a cada uno de los tres analistas los puntos en los que estas de acuerdo, y en los que estas en desacuerdo, justificando donde sea necesario.
- Propón una arquitectura de *data warehousing* que se pueda acomodar a este caso de uso.

Problema 3 La gente del SIS te menciona que les interesa usar Big Query, pero realmente no saben por qué funciona tan bien, ni que ventajas pueden obtener de este sistema, ni cuáles son los casos de uso donde brilla. Para darles tranquilidad, tienes que responder las siguientes preguntas:

1. ¿En qué se basa la capacidad de cómputo de Big Query?
2. Explica cómo lo hace Big Query para responder una consulta de join.
3. Explica cómo lo hace Big Query para responder una consulta de agregación.
4. Explica cómo lo hace Big Query para responder una consulta que utiliza Window Functions.

Problema 4 La interpol tiene una base de datos con todo lo que han comprado o vendido en tiendas físicas los criminales más buscados en el mundo.

El problema, es que los datos de la interpol son llenados a mano por las tiendas, y pueden tener typos u otros errores en el nombre del criminal. Por ejemplo, uno podría encontrar Adrisn en vez de Adrian, y no siempre se registran los nombres de la misma forma (existen entradas tipo Juan L. Reutter, o Juan Reutter, y hasta hay una bajo Juanito Reutter).

La interpol necesita saber el detalle de compras y ventas por cada criminal, para hacerles un seguimiento. Pero son terabytes de datos y no saben como cruzar correctamente los nombres de los criminales: por todos los errores descritos arriba no basta con buscar que el nombre sea el mismo, si no que hay que encontrar una forma de unificar todas las distintas entradas que pertenezcan a un mismo criminal. Por ejemplo, podría ser que Adrisn haya comprado algo en Calama, y que Adrián haya vendido algo en Macul, pero para vincular esta información es necesario tomar a ambas entradas de nombre como la misma persona.

Como acabas de ver locally-sensitive hashing (LSH), hace sentido pensar en ayudarle a la Interpol con esto. Asumiendo que tienes una familia de funciones (d_1, d_2, p_1, p_2)-sensitiva para la distancia entre strings[1], elabora una estrategia basada en LSH para que la interpol tenga una lista desambiguada de quién compro o vendió donde.

Explica si tu estrategia puede tener falsos positivos, negativos, y como podrías reducirlos.

[1] comúnmente usamos la llamada *edit distance*, que mide cuantos caracteres hay que borrar o agregar para ir de un string a otro.