IIC 2413 – Bases de Datos
Guía I3

Pregunta 1: Almacenamiento, índices y Map Reduce

a) En un esquema que contiene a la relación `R(a int, b varchar(10))` el 70 % de las consultas son `SELECT * FROM R WHERE a = i`, donde i es un entero. El otro 30 % son consultas del tipo `SELECT * FROM R WHERE a >i AND a <j`. Si tuviera que escoger un índice a utilizar, ¿cuál sería?. Justifique su respuesta.

b) En un sistema, el 99 % de las consultas son de proyección y agregación. ¿Qué tipo de bases de datos conviene usar? Justifique su respuesta.

c) Considere una relación de 3.000.000 páginas.

- Indique el número de fases y de I/O del External Merge Sort no optimizado.
- Indique el número de fases y de I/O del External Merge Sort optimizado que imprime directamente el resultado final, sin materializar el resultado final. Suponga buffer óptimo.
- ¿Qué significa que el buffer sea óptimo? ¿Por qué un buffer en el que caben 3.000.000 de páginas no es necesario? Demuestre su respuesta.

d) Sea la relación $R(a, b, c, d)$ cuyo tamaño es de 2 millón de tuplas, en que cada página contiene en promedio P tuplas. Las tuplas de R están ordenados de manera aleatoria (cuando no hay índice). El atributo a es además un candidato a llave primaria, cuyos valores van del 0 al 3.999.999 (distribuídos uniformemente). Para cada una de las consultas a continuación, diga el número de I/O que se harán en cada uno de los siguientes casos:

- Analizar R sin ningún índice.
- Usar un *B+Tree Unclustered* sobre el atributo a. El árbol es de altura h. La cantidad de punteros por pagina es de M. El índice de ocupación de las hojas es del 90 %.

Las consultas son:

1. Encontrar todas las tuplas de R.
2. Encontrar todas las tuplas de R tal que $a < 100$.
3. Encontrar todas las tuplas de R tal que $a = 100303$.
4. Encontrar todas las tuplas de R tal que $a > 50$ y $a \leq 150$.

Comente sus resultados e indique que índice prefiere en los distintos casos.

e) Considere los datos de la pregunta anterior. Suponga que P es cercano a 5. Considere un Hash Index Clustered con 200.000 Buckets. ¿Cuanto I/O requiere la consulta "Encontrar todas las tuplas de R tal que $a = 100303$"?. ¿Cómo puede ayudar un Hash Index dinámico?.

f) Considere un esquema de dos tablas `A(id int, name varchar(10))` y `B(id int, name varchar(10))`. Suponga que quiere hacer la intersección entre A y B ($A \cap B$) comparando según id, pero solamente dispone de un archivo que se ve de la siguiente forma:

T1	T2	T3	T4
			R(d)
X			
	R(b)		
	W(a)		
		Y	
		W(b)	
			R(c)
Z			

Cuadro 1: Schedule pregunta 2 parte b

```
A,1,palabra1
A,2,palabra2
A,3,palabra3
B,1,palabra1
...
```

El primer término antes de la coma representa la tabla, el segundo el id y el tercero el name. Entregue un algoritmo (basta explicar con palabras) Map - Reduce que ejecute la consulta deseada. Note que `A` y `B` pueden tener duplicados.

Pregunta 2: Algoritmos internos y transacciones

a) En clases se vio que el costo en I/O del Block Nested Loop Join es de $Pags(R) + (Pags(R)/Buffer) \cdot Pags(S)$. Existen otros algoritmos de join que pueden hacer usos de índices sobre las bases de datos.

- Entregue un algoritmo (explicado con palabras) que compute el join $R \bowtie S$ entre `R(a, b)` y `S(a, c)` en el que ambas llaves primarias están indexadas por un B+Tree clustered. Entregue el costo estimado en I/O (debería ser mejor que el costo del Block Nested Loop Join).
  Hint: El B+Tree mantiene la relación ordenada. Debe hacer uso de esto.
- Considere que ahora ambas llaves primarias están indexadas por un B+Tree Unclustered. Diga cómo se ve afectado el costo en I/O. Entregue un ejemplo en el que sea mejor hacer el join utilizando el índice y otro donde sea mejor hacer el Block Nested Loop Join.
  Hint: No todas las tuplas de R y S forman parte del output. El principal costo de un índice unclustered es tener que ir a buscar lo que indican los punteros a disco.

b) Sea el schedule del cuadro 1:

- De un valor para X, Y y Z tal que representen acciones (`R`, `W`) sobre variables (`a`, `b`, `c`, `d`) para que el schedule sea no serializable? Justifique su respuesta.
- ¿Por qué al utilizar *Strict 2PL* los schedules necesariamente son *Serializables*? De un valor para X, Y y Z tal que representen acciones (`R`, `W`) sobre variables (`a`, `b`, `c`, `d`) que puedan ocurrir si se está utilizando Strict 2PL.

Pregunta 3: MongoDB

Piense en una base de datos en MongoDB con tres colecciones, una de usuarios de una red social, otra de compras y otra que relaciona ambas colecciones. Se muestra a continuación una instancia de documento

para cada colección:

```
// Usuario
{
  "id_usuario": 1,
  "name": "Florencia Barrios",
  "desc": "Me encanta jugar hockey!"
}

// Relación
{
  "id_usuario": 1,
  "id_compras": [1, 3, 4]
}

// Compra
{
  "id_compra": 1,
  "fecha": "10-09-2016",
  "compras": [
    {
      "producto": "Palo de Hockey",
      "tipo": "Deporte",
      "valor": 20000
    },
    {
      "producto": "Té Chai",
      "tipo": "Comida",
      "valor": 3000
    }
  ]
}
```

Entregue las siguientes consultas en MongoDB. Si va a utilizar un índice indique cómo lo creó. Puede hacer uso de JavaScript o Python en caso de ser necesario:

- Entregue el nombre de cada usuario que contenga en su descripción la palabra "Estudiante" y la frase "Ingeniería UC" pero no la frase "Bases de Datos" junto al id de cada compra que ha hecho.
- Entregue el nombre de cada persona junto al nombre de cada producto tipo "Entretención" que ha comprado.
- Para cada persona que en su descripción contenga la frase "Estudiante de Ingeniería" y no la frase "Ingeniería UC" indique el nombre de la persona junto al id de cada compra, junto al promedio de los valores de los productos adquiridos en esa compra.