



INSTITUTO POLITÉCNICO NACIONAL



**UNIDAD PROFESIONAL INTERDISCIPLINARIA EN
INGENIERÍA Y TECNOLOGÍAS AVANZADAS**

UPIITA

PROFESOR: Carlos De La Cruz Sosa

ASIGNATURA: Bases de Datos Distribuidas

GRUPO: 3TM3

Tarea 4

EQUIPO 5:

- Bernal Aguilar Yuvia Abigail
- Contreras Jimenez Mariana Montserrat
- Medina Gómez Jimena Zarahí

La entrada dada a ChatGpt fue la siguiente:

“ Apartir del siguiente documento, genera un fragmento de forma horizontal primaria de acuerdo con la teoria de tamer ozsu considerando las tablas productsubcategory (productsubcategoryid, productcategoryid, name, row_guid, modified_date), products(productid, productsubcategoryid), salesorderdetail(salesorderid, productid, orderqty), algoritmo com_min y las consultas”

Lo que nos arrojó fue lo siguiente:

Basado en el documento proporcionado y la teoría de fragmentación horizontal primaria de Tamer Özsu, voy a generar un esquema de fragmentación para la tabla ProductSubCategory utilizando el algoritmo COM_MIN.

Contexto

Tenemos las siguientes tablas:

- ProductSubCategory (productsubcategoryid, productcategoryid, name, row_guid, modified_date)
- Products (productid, productsubcategoryid)
- SalesOrderDetail (salesorderid, productid, orderqty)

Y las consultas frecuentes que requieren fragmentación:

- Listar el producto más vendido por categoría
- Listar clientes con más órdenes por territorio
- Listar órdenes con productos similares a una orden específica

Proceso de Fragmentación

- **Paso 1:** Identificación de Predicados Candidatos (PR)

Para la tabla ProductSubCategory, los predicados candidatos se basan en el atributo productcategoryid:

```
PRProductSubCategory = {  
  
    P1: productcategoryid = 1,  
  
    P2: productcategoryid = 2,  
  
    P3: productcategoryid = 3,  
  
    P4: productcategoryid = 4  
  
}
```

- **Paso 2:** Verificación de Completitud y Minimalidad (PR')

El conjunto PR es completo (cubre todas las categorías posibles) y mínimo (no hay redundancia):

PRProductSubCategory = PRProductSubCategory

- **Paso 3:** Generación de Predicados Mínimos (M)

Aplicamos combinaciones lógicas para crear predicados mínimos mutuamente excluyentes:

MProductSubCategory = {

M1: $P1 \wedge \neg P2 \wedge \neg P3 \wedge \neg P4$ (productcategoryid = 1)

M2: $\neg P1 \wedge P2 \wedge \neg P3 \wedge \neg P4$ (productcategoryid = 2)

M3: $\neg P1 \wedge \neg P2 \wedge P3 \wedge \neg P4$ (productcategoryid = 3)

M4: $\neg P1 \wedge \neg P2 \wedge \neg P3 \wedge P4$ (productcategoryid = 4)

M5: $\neg P1 \wedge \neg P2 \wedge \neg P3 \wedge \neg P4$ (productcategoryid es NULL u otro valor)

}

Nota: Se descartan combinaciones redundantes como $P1 \wedge P2$ ya que productcategoryid no puede tener múltiples valores simultáneos.

- **Paso 4:** Creación de Fragmentos (F)

Cada predicado mínimo genera un fragmento:

FProductSubCategory = {

FPSC1: $\sigma(\text{productcategoryid}=1)(\text{ProductSubCategory})$

FPSC2: $\sigma(\text{productcategoryid}=2)(\text{ProductSubCategory})$

FPSC3: $\sigma(\text{productcategoryid}=3)(\text{ProductSubCategory})$

FPSC4: $\sigma(\text{productcategoryid}=4)(\text{ProductSubCategory})$

FPSC5: $\sigma(\text{productcategoryid IS NULL OR productcategoryid NOT IN (1,2,3,4)})(\text{ProductSubCategory})$

}

- **Implementación SQL:**

-- Fragmento FPSC1 (Categoría 1)

CREATE TABLE FPSC1 AS

SELECT * FROM ProductSubCategory

WHERE productcategoryid = 1;

-- Fragmento FPSC2 (Categoría 2)

CREATE TABLE FPSC2 AS

SELECT * FROM ProductSubCategory

WHERE productcategoryid = 2;

-- Fragmento FPSC3 (Categoría 3)

CREATE TABLE FPSC3 AS

SELECT * FROM ProductSubCategory

WHERE productcategoryid = 3;

-- Fragmento FPSC4 (Categoría 4)

CREATE TABLE FPSC4 AS

SELECT * FROM ProductSubCategory

WHERE productcategoryid = 4;

-- Fragmento FPSC5 (Otras categorías o NULL)

CREATE TABLE FPSC5 AS

SELECT * FROM ProductSubCategory

WHERE productcategoryid IS NULL

OR productcategoryid NOT IN (1, 2, 3, 4);

- **Fragmentación derivada:**

-- Fragmento derivado FP1 (Productos de categoría 1)

CREATE TABLE FP1 AS

SELECT P.*

FROM Products P JOIN FPSC1

ON P.productsubcategoryid = FPSC1.productsubcategoryid;

-- Fragmento derivado FSOD1 (Detalles de orden con productos de categoría 1)

CREATE TABLE FSOD1 AS

SELECT SOD.*

FROM SalesOrderDetail SOD JOIN FP1

ON SOD.productid = FP1.productid;

CONCLUSIONES.

Medina Gómez Jimena Sarahí.

La fragmentación horizontal primaria permite dividir una tabla en subconjuntos más manejables según condiciones específicas (predicados), lo que mejora notablemente el rendimiento y eficiencia de las consultas más frecuentes. Aplicando el algoritmo COM_MIN de Tamer Özsu, se garantiza que los fragmentos sean completos (cubren todos los datos posibles), mínimos (sin redundancias) y mutuamente excluyentes, lo cual es clave para mantener la consistencia y optimizar el procesamiento distribuido.

Contreras Jiménez Mariana Montserrat.

A través de la fragmentación derivada, se extiende la división original a tablas relacionadas mediante claves foráneas, asegurando que los datos fragmentados conserven su integridad referencial. Esto permite que consultas complejas que involucran múltiples tablas (como ventas por categoría) se ejecuten localmente en cada fragmento, reduciendo el tráfico de red, mejorando los tiempos de respuesta y facilitando la administración de bases de datos distribuidas.

Berna Aguilarl Yuvia Abigail.

La aplicación de la fragmentación horizontal primaria no solo organiza los datos según patrones de uso, sino que también permite un diseño lógico de bases de datos alineado con las necesidades reales del sistema. Al fragmentar con base en atributos como **productcategoryid**, se facilita el acceso rápido a la información relevante para cada categoría, lo cual resulta esencial en entornos donde se busca disponibilidad, escalabilidad y rendimiento óptimo en consultas específicas.

CONCLUSIÓN GENERAL.

La fragmentación horizontal primaria, basada en la teoría de Tamer Özsu y aplicada mediante el algoritmo COM_MIN, representa una estrategia eficiente para optimizar el rendimiento de las bases de datos distribuidas. Al dividir los datos en fragmentos completos, mínimos y mutuamente excluyentes, se facilita el acceso localizado a la información, especialmente cuando las consultas siguen patrones específicos como la categoría de productos. Además, la fragmentación derivada asegura la integridad de los datos entre tablas relacionadas, permitiendo que las operaciones complejas se realicen con menor costo y mayor velocidad. En conjunto, esta técnica no solo mejora el desempeño del sistema, sino que también promueve un diseño más escalable, organizado y alineado con las necesidades reales de consulta y distribución de datos.

