



INSTITUTO POLITECNICO NACIONAL



UNIDAD PROFESIONAL INTERDISCIPLINARIA EN INGENIERIA
Y TECNOLOGIAS AVANZADAS - IPN

MATERIA

Bases de Datos Distribuidas

PROFESOR

Carlos De La Cruz Sosa

ALUMNOS

Fernández Guerrero Keb Sebastián

Ramírez Orozco Juan Carlos

Sánchez Herrera Armando Eduardo

TEMA

Fragmentación Horizontal

Grupo 3TM3

Equipo 06

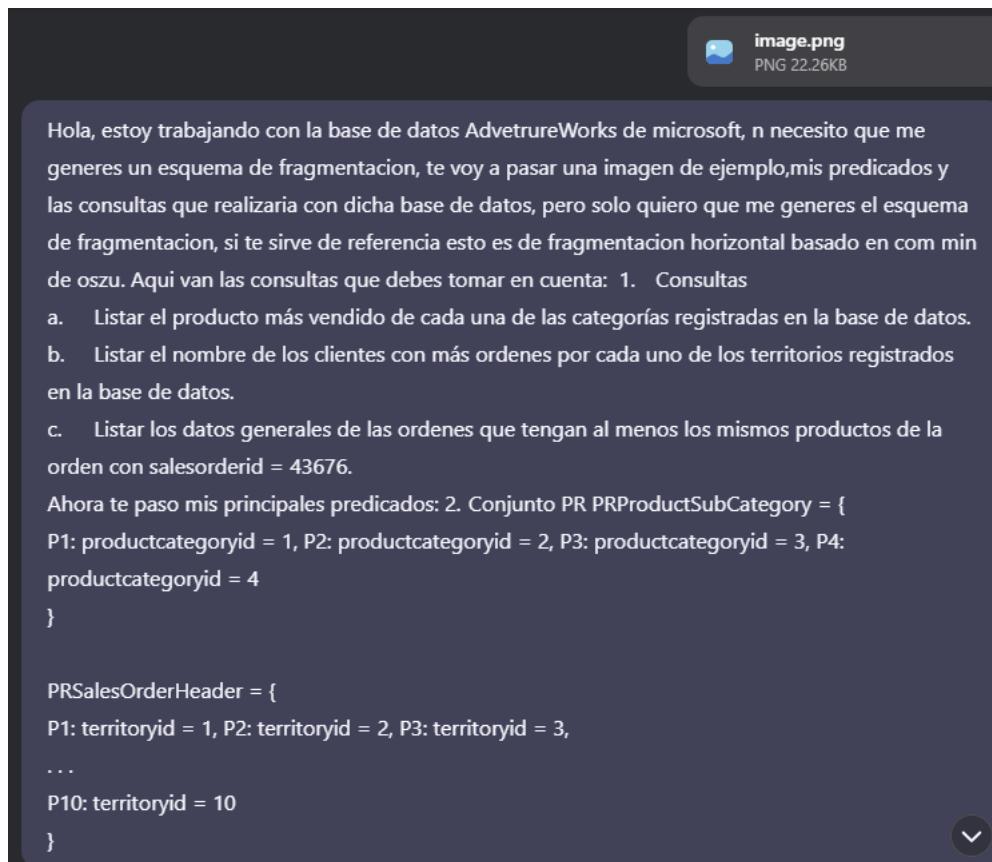
Tarea No. 04

En base a que las instrucciones del profesor fueron:

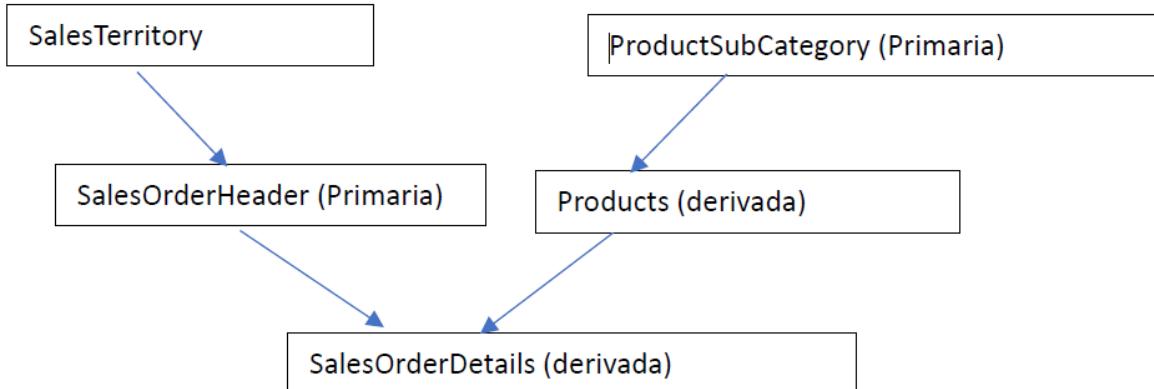
Deben solicitar que genere un esquema de fragmentación solamente con las tablas y los conjuntos iniciales de PR de cada una de las tablas primarias

ahh, e indicar las consultas para que pueda decidir

Generamos un nuevo chat con la IA Deepseek para resolver la presente tarea, por lo cual nuestro prompt fue el siguiente:



Es decir le pasamos la imagen de referencia de cómo se dividirían las tablas de la base de datos de AdventureWorks anexado en el documento titulado Fragmentación Horizontal COM_MIN que compartió el profesor el día miércoles 04 de junio a las 02:59 pm, las consultas que pone de ejemplo en dicho documento, y los predicados PR que sirvieron de ejemplificación , la imagen en cuestión es la siguiente:



La respuesta generada por la IA fue la siguiente, donde destacaremos las partes de código con una diferente tipografía y encerrados en una celda (Hemos decidido copiar el formato y fondo ocupado pro al ia para su mejor apreciación):

Esquema de Fragmentación Horizontal para AdventureWorks

Basado en el enfoque COM_MIN de Ozsu y Valduriez, y considerando las consultas y predicados proporcionados, propongo el siguiente esquema de fragmentación:

Fragmentación Primaria

1. ProductSubCategory (Fragmentación Horizontal Primaria)

```

ProductSubCategory_1: σ(productcategoryid = 1)
ProductSubCategory_2: σ(productcategoryid = 2)
ProductSubCategory_3: σ(productcategoryid = 3)
ProductSubCategory_4: σ(productcategoryid = 4)

```

2. SalesOrderHeader (Fragmentación Horizontal Primaria)

```

SalesOrderHeader_1: σ(territoryid = 1)
SalesOrderHeader_2: σ(territoryid = 2)
SalesOrderHeader_3: σ(territoryid = 3)
...
SalesOrderHeader_10: σ(territoryid = 10)

```

Fragmentación Derivada

1. Products (Fragmentación Horizontal Derivada de ProductSubCategory)

```

Products_1: Products × ProductSubCategory_1
Products_2: Products × ProductSubCategory_2

```

```
Products_3: Products ✕ ProductSubCategory_3  
Products_4: Products ✕ ProductSubCategory_4
```

2. SalesOrderDetails (Fragmentación Horizontal Derivada de SalesOrderHeader)

```
SalesOrderDetails_1: SalesOrderDetails ✕ SalesOrderHeader_1  
SalesOrderDetails_2: SalesOrderDetails ✕ SalesOrderHeader_2  
SalesOrderDetails_3: SalesOrderDetails ✕ SalesOrderHeader_3  
...  
SalesOrderDetails_10: SalesOrderDetails ✕ SalesOrderHeader_10
```

Justificación:

1. La fragmentación de `ProductSubCategory` por `productcategoryid` optimiza la consulta del producto más vendido por categoría.
2. La fragmentación de `SalesOrderHeader` por `territoryid` optimiza la consulta de clientes con más órdenes por territorio.
3. Las fragmentaciones derivadas mantienen la relación semántica entre tablas y permiten ejecutar joins locales en lugar de distribuidos.
4. La consulta sobre órdenes con productos específicos se beneficia de la fragmentación derivada de `SalesOrderDetails`.

Este esquema minimiza el acceso a datos no relevantes para las consultas especificadas y permite el procesamiento paralelo de fragmentos cuando sea posible.

Comparando con el archivo completo del profesor nos pudimos dar cuenta de que la IA tiende simplificar el procedimiento de una forma muy plana y simple repitiendo la misma estructura varias veces o haciendo cambios mínimos, aunque destacamos el uso de los operadores del álgebra relacional, cuestionando a la inteligencia artificial sobre su uso, rescatamos la siguiente información

Son símbolos que se utilizan para definir formalmente las operaciones en bases de datos relacionales.

σ (Sigma minúscula): Operador de Selección Horizontal

- **Función:** Filtra filas de una tabla que cumplen una condición.

- **Ejemplo:**

$\sigma(\text{territoryid} = 1) (\text{SalesOrderHeader})$

Devuelve todas las filas de `SalesOrderHeader` donde el `territoryid` sea igual a 1.

- En el esquema de fragmentación se usa para crear fragmentos primarios basados en predicados (ej: $\text{ProductSubCategory_1} = \sigma(\text{productcategoryid} = 1)$).

\bowtie (Símbolo de "Semi-join"): Operador de Semi-unión

- **Función:** Devuelve las filas de una tabla que tienen coincidencias en otra tabla, sin duplicar los datos de la segunda tabla.

- Ejemplo:

$\text{Products} \bowtie \text{ProductSubCategory_1}$

Devuelve los productos que pertenecen a la subcategoría filtrada por $\text{productcategoryid} = 1$.

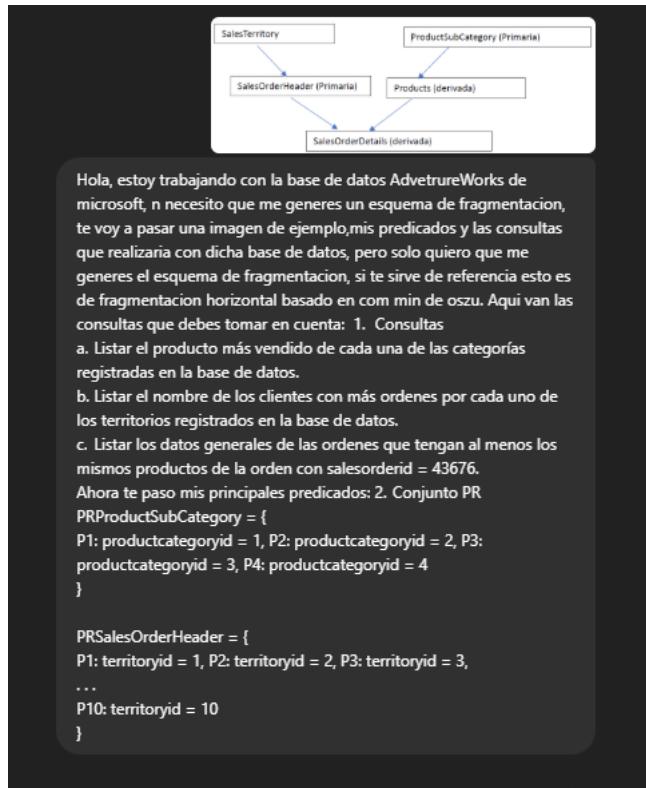
- En el esquema de fragmentación se usa para fragmentación derivada (ej: $\text{Products_1} = \text{Products} \bowtie \text{ProductSubCategory_1}$).

Relación con la fragmentación

- **σ :** Define fragmentos **primarios** (ej: dividir SalesOrderHeader por territoryid).
- **\bowtie :** Define fragmentos **derivados** (ej: SalesOrderDetails se fragmenta en base a SalesOrderHeader).

Estos operadores garantizan que los fragmentos mantengan la integridad referencial y optimicen las consultas que mencionaste (ej: *consultas por territorio o categoría*).

Por ejercicio de comparativa, realizamos el mismo prompt en ChatGPT



Gracias por la información y por compartir la imagen del esquema base. Tomando en cuenta:

- Las **consultas** que realizarás,
- Tus **predicados de fragmentación horizontal** (`PRProductSubCategory` y `PRSalesOrderHeader`), y
- El enfoque basado en el **esquema de fragmentación horizontal de Com Min (Oszu)**,

te propongo el siguiente **esquema de fragmentación horizontal derivada** para la base de datos AdventureWorks, basado en las relaciones mostradas:

Esquema de Fragmentación Horizontal (basado en Com Min)

◆ Fragmentación primaria

1. ProductSubCategory

- Se fragmenta horizontalmente con base en `productcategoryid`
- Predicados:
 - P1: `productcategoryid = 1`
 - P2: `productcategoryid = 2`
 - P3: `productcategoryid = 3`

- P4: productcategoryid = 4

Resultan: ProductSubCategory_1, ProductSubCategory_2,
ProductSubCategory_3, ProductSubCategory_4

2. SalesOrderHeader

- Se fragmenta horizontalmente con base en territoryid
- Predicados:
 - P1: territoryid = 1
 - ...
 - P10: territoryid = 10

Resultan: SalesOrderHeader_1, ... SalesOrderHeader_10

◆ Fragmentación derivada

3. Products

- Se fragmenta derivadamente a partir de ProductSubCategory
- Fragmentos:
 - Products_1 con subcategorías de ProductSubCategory_1
 - ...
 - Products_4 con subcategorías de ProductSubCategory_4

4. SalesOrderDetails

- Se fragmenta derivadamente a partir de:
 - SalesOrderHeader (por SalesOrderID)
 - Products (por ProductID)
- Para mantener consistencia y facilitar las consultas cruzadas:
 - Crear fragmentos SalesOrderDetails_i_j, donde:
 - i corresponde a territoryid (1-10)
 - j a productcategoryid (1-4)

Resultan hasta 40 fragmentos combinados: SalesOrderDetails_1_1,
SalesOrderDetails_1_2, ..., SalesOrderDetails_10_4

◆ Otras tablas:

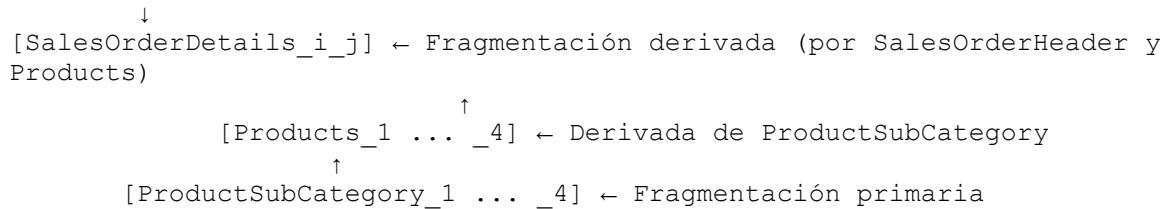
5. SalesTerritory

- No se fragmenta (dimensión pequeña)

✿ Resumen visual (simplificado)

```

scss
CopiarEditar
[SalesTerritory]
  ↓
[SalesOrderHeader_1 ... _10] ← Fragmentación primaria
  
```



Justificación

- **Consulta a:** necesita `Products` fragmentados por categoría para hallar el más vendido → justifica fragmentación por `ProductCategoryID`
- **Consulta b:** necesita `SalesOrderHeader` por territorio → justifica fragmentación por `TerritoryID`
- **Consulta c:** accede a `SalesOrderDetails` cruzando productos y órdenes → se necesita fragmentación derivada sobre ambas tablas para eficiencia.

Por lo que podemos notar aquí se fue por un enfoque en arreglos y tablas, sin embargo no es muy claro como se relacionan entre si.