



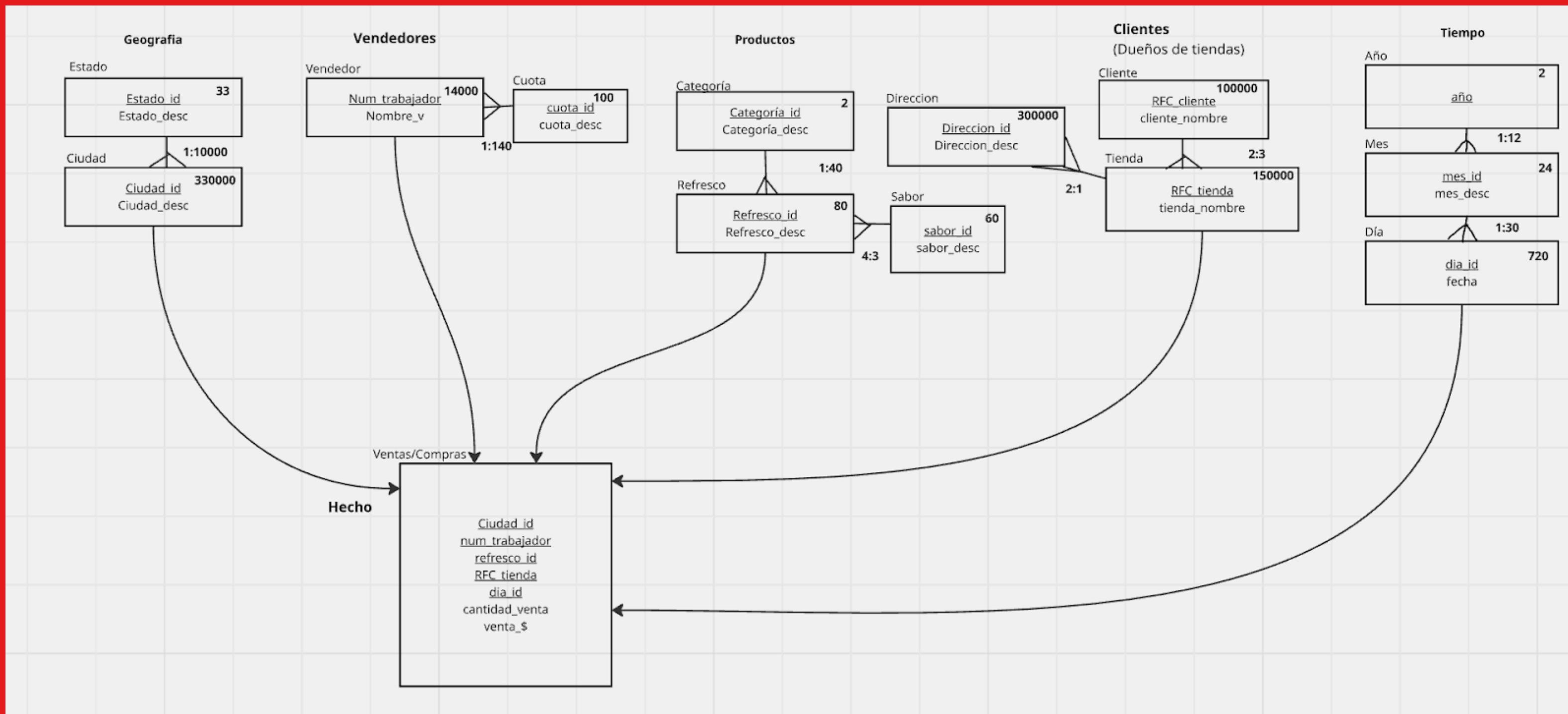
INSTITUTO DE
INVESTIGACIONES
EN MATEMÁTICAS
APLICADAS Y
EN SISTEMAS

Proyecto BDE

Coca-Cola

Milena Fernanda Rivera Hernández
Ángel Noel Pérez Martínez
Ignacio Chuquiure Gil

Diagrama Entidad- Relación del modelado dimensional.



¿Por qué elegimos copo de nieve?

Lo elegimos pensando en garantizar la calidad y gestión de los datos ya que hablamos de millones de clientes de Coca Cola México y con copo de nieve tenemos:

- Mayor integridad de datos
- Menor sobrecarga en el almacenamiento
- Mayor adaptabilidad a los cambios
- Mayor control en las relaciones 1:M



1.Consulta DICE:

“Mostrar las ventas de refrescos de sabor Cola Original, en la categoría Refresco Regular, solo para los meses de diciembre a febrero durante los últimos 2 años”

Justificación

Observamos que la consulta fija tipo de sabor (Cola Original), la categoría (Refresco Regular) solo para meses específicos (diciembre a febrero) en los 2 últimos años. Por lo que podemos concluir que es una consulta de tipo dice

anio	mes	categoria	sabor	total_unidades_vendidas	total_ventas_dinero
2023	Diciembre	Refresco Regular	Cola Original	46424	\$ 947,430.00
2023	Enero	Refresco Regular	Cola Original	42100	\$ 850,430.00
2023	Febrero	Refresco Regular	Cola Original	44773	\$ 891,845.00
2024	Diciembre	Refresco Regular	Cola Original	1416	\$ 27,730.00
2024	Enero	Refresco Regular	Cola Original	41077	\$ 797,115.00
2024	Febrero	Refresco Regular	Cola Original	43153	\$ 854,785.00



2.Consulta SLICE:

“Para evitar producir refrescos que no generan ganancias, desea saber qué refresco es el que menos se ha vendido en los últimos dos años.”

Justificación

Es una consulta de tipo slice ya que solo se está filtrando el año en la dimensión tiempo. En cambio no se filtra con el refresco ni la categoría (por lo cual, esta consulta no es de tipo dice). El filtro sobre el año viene dado cuando en SQL escribimos:

“WHERE a.anio >= (SELECT MAX(anio) FROM anio) -1”. A partir de esto podemos en definitiva clasificar esta consulta como una consulta de tipo slice.



```
proyectococacola=> -- Deseamos obtener el refresco menos vendido en los 2 ultimos años
SELECT
    r.refresco_desc AS "Refresco menos vendido",
    c.categoría_desc AS "Categoria",
    s.sabor_desc AS "Sabor",
    SUM(v.cantidad_venta) AS "Total vendido"
FROM ventas v
JOIN refresco r ON v.refresco_id = r.refresco_id
JOIN categoría c ON r.categoría_id = c.categoría_id
JOIN sabor s ON r.sabor_id = s.sabor_id
JOIN dia d ON v.dia_id = d.dia_id
JOIN mes m ON d.mes_id = m.mes_id
JOIN anio a ON m.anio = a.anio
WHERE a.anio >= (SELECT MAX(anio) FROM anio) - 1
GROUP BY r.refresco_desc, c.categoría_desc, s.sabor_desc
ORDER BY SUM(v.cantidad_venta) ASC
LIMIT 1; -- solo nos interesa 1 refresco
Refresco menos vendido | Categoría | Sabor | Total vendido
-----+-----+-----+
Sidral Mundet 500ml | Refresco Regular | Manzana | 161943
(1 fila)
```

3.Consulta PIVOT:

“La empresa desea saber las ventas de refrescos cada mes (columnas) por categoría de refrescos (filas).”

```
proyectococacola=>
-- Consulta de tipo PIVOT: Categoría (Filas) vs Meses (Columnas)
```

```
SELECT
    cat.categoria_desc AS "Categoría",
    -- Aquí ocurre la ROTACIÓN (Pivot): Convertimos los meses en columnas
    SUM(CASE WHEN m.mes_desc = 'Enero' THEN v.venta_$ ELSE 0::MONEY END) AS "Enero",
    -- Aquí ocurre la ROTACION (Pivot): Convertimos los meses en columnas
    SUM(CASE WHEN m.mes_desc = 'Enero' THEN v.venta_$ ELSE 0::MONEY END) AS "Enero",
    SUM(CASE WHEN m.mes_desc = 'Febrero' THEN v.venta_$ ELSE 0::MONEY END) AS "Febrero",
    SUM(CASE WHEN m.mes_desc = 'Marzo' THEN v.venta_$ ELSE 0::MONEY END) AS "Marzo",
    SUM(CASE WHEN m.mes_desc = 'Abril' THEN v.venta_$ ELSE 0::MONEY END) AS "Abril",
    SUM(CASE WHEN m.mes_desc = 'Mayo' THEN v.venta_$ ELSE 0::MONEY END) AS "Mayo",
    SUM(CASE WHEN m.mes_desc = 'Junio' THEN v.venta_$ ELSE 0::MONEY END) AS "Junio",
    SUM(CASE WHEN m.mes_desc = 'Julio' THEN v.venta_$ ELSE 0::MONEY END) AS "Julio",
    SUM(CASE WHEN m.mes_desc = 'Agosto' THEN v.venta_$ ELSE 0::MONEY END) AS "Agosto",
    SUM(CASE WHEN m.mes_desc = 'Septiembre' THEN v.venta_$ ELSE 0::MONEY END) AS "Septiembre",
    SUM(CASE WHEN m.mes_desc = 'Octubre' THEN v.venta_$ ELSE 0::MONEY END) AS "Octubre",
    SUM(CASE WHEN m.mes_desc = 'Noviembre' THEN v.venta_$ ELSE 0::MONEY END) AS "Noviembre",
    SUM(CASE WHEN m.mes_desc = 'Diciembre' THEN v.venta_$ ELSE 0::MONEY END) AS "Diciembre",
    -- Columna totalizadora (opcional pero recomendada en pivots)
    SUM(v.venta_$) AS "Total Anual"
FROM ventas v
JOIN refresco r ON v.refresco_id = r.refresco_id
JOIN categoria cat ON r.categoria_id = cat.categoria_id
JOIN dia d ON v.dia_id = d.dia_id
JOIN mes m ON d.mes_id = m.mes_id
JOIN anio a ON m.anio = a.anio
ORDER BY "Total Anual" DESC;
```

Categoría	Enero	Febrero	Marzo	Abril	Mayo	Junio	Julio
Agosto	Septiembre	Octubre	Noviembre	Diciembre	Total Anual		
Refresco Regular	\$ 1,273,297.00	\$ 1,369,303.00	\$ 40,003.00	\$ 19,607.00	\$ 13,832.00	\$ 26,836.00	\$ 15,687.00
Refresco Light	\$ 543,655.00	\$ 556,500.00	\$ 8,515.00	\$ 7,860.00	\$ 2,910.00	\$ 8,920.00	\$ 18,795.00
Refresco Zero	\$ 216,315.00	\$ 189,630.00	\$ 4,545.00	\$ 2,655.00	\$ 5,400.00	\$ 6,345.00	\$ 5,475.00
Agua Saborizada	\$ 193,248.00	\$ 182,664.00	\$ 1,728.00	\$ 2,952.00	\$ 3,852.00	\$ 6,822.00	\$ 1,800.00
	\$ 5,292.00	\$ 4,194.00	\$ 0.00	\$ 2,898.00	\$ 2,034.00	\$ 407,484.00	



Justificación

A parte de cómo redactamos la consulta, podemos saber que es de tipo pivot pues estamos cambiando de algo que aparecería como filas (es decir los meses) lo convertimos en columnas. Esto se observa en consulta en SQL vista abajo, en donde al momento de realizar el SELECT especificamos también que aparezcan como columnas los meses. Para así después tener un contraste con las filas de las categorías de refrescos, haciendo de esta forma que esta sea una consulta tipo pivot.

4.Consulta DRILL-DOWN

"Se desea saber en qué mes del año se vende más refresco y en particular que día del año se vende más refresco."

```
proyectococacola=> SELECT
    mes_top.mes_desc AS "Mes con más ventas",
    mes_top.anio AS "Año del mes",
    mes_top.ventas_mes AS "Ventas del mes",
    dia_top.fecha AS "Día con más ventas",
    dia_top.ventas_dia AS "Ventas del día"
FROM (
    -- Obtenemos el mes con mas ventas
    SELECT m.mes_desc, a.anio, SUM(v.cantidad_venta) as ventas_mes
    FROM ventas v
    JOIN dia d ON v.dia_id = d.dia_id
    JOIN mes m ON d.mes_id = m.mes_id
    JOIN anio a ON m.anio = a.anio
    GROUP BY m.mes_desc, a.anio
    ORDER BY SUM(v.cantidad_venta) DESC
    LIMIT 1
) mes_top,
(
    -- Vemos el dia con mas ventas
    SELECT d.fecha, SUM(v.cantidad_venta) as ventas_dia
    FROM ventas v
    JOIN dia d ON v.dia_id = d.dia_id
    GROUP BY d.fecha
    ORDER BY SUM(v.cantidad_venta) DESC
    LIMIT 1
) dia_top;
Mes con más ventas | Año del mes | Ventas del mes | Día con más ventas | Ventas del día
-----+-----+-----+-----+-----+
Agosto | 2023 | 192195 | 2023-05-15 | 51643
(1 fila)
```



Justificación

Es una consulta tipo drill-down pues primero fijamos el año en la dimensión tiempo, después bajamos y en mes buscamos cual es el que vende más refresco. Finalmente nos vamos hasta el día para buscar el día donde se vendió más refresco. Cómo lo hacemos desde la misma dimensión de tiempo y estamos buscando (o minando) de arriba hacia abajo para obtener la información que se nos pide, entonces debemos clasificar a esta consulta como una consulta de tipo drill-down.

5.Consulta DRILL-UP:

“La empresa desea saber cuáles fueron las ventas por ciudad, cuáles fueron los totales por estado y cuál fue la venta total.”

Justificación

Ya que primero deseamos saber sobre las ventas en ciudades para así después pasar a ventas por estado dentro de la dimensión geografía. Entonces, claramente realizamos una búsqueda (o minería) de abajo hacia arriba dentro de esta misma dimensión. Al igual que al emplear el ROLLUP en el código en SQL, en consiguiente con facilidad podemos clasificar a esta consulta como una consulta de tipo drill-up.



```
proyectococacola=> SELECT
    e.estado_desc AS "Estado",
    ci.ciudad_desc AS "Ciudad",
    SUM(v.venta_$/) AS "Ventas Totales"
FROM ventas v
JOIN ciudad ci ON v.ciudad_id = ci.ciudad_id
JOIN estado e ON ci.estado_id = e.estado_id
-- ROLLUP genera las filas extra con los totales automáticamente
GROUP BY ROLLUP (e.estado_desc, ci.ciudad_desc)
```

Estado	Ciudad	Ventas Totales
Baja California	Tijuana	\$ 3,874,328.00
Baja California		\$ 3,874,328.00
Chihuahua	Ciudad Juárez	\$ 3,806,508.00
Chihuahua		\$ 3,806,508.00
Ciudad de México	Ciudad de México	\$ 3,987,922.00
Ciudad de México		\$ 3,987,922.00
Estado de México	Naucalpan	\$ 3,827,467.00
Estado de México		\$ 3,827,467.00
Jalisco	Guadalajara	\$ 4,105,523.00
Jalisco	Zapopan	\$ 3,823,171.00
Jalisco		\$ 7,928,694.00
Nuevo León	Monterrey	\$ 3,775,010.00
Nuevo León		\$ 3,775,010.00
Puebla	Puebla	\$ 3,958,917.00
Puebla		\$ 3,958,917.00
Querétaro	Querétaro	\$ 3,842,709.00
Querétaro		\$ 3,842,709.00
Yucatán	Mérida	\$ 3,838,809.00
Yucatán		\$ 3,838,809.00
		\$ 38,840,364.00

(20 filas)

Resto de Consultas



6. Consulta tipo Drill-Down

“A la Compañía le interesa saber qué ciudad es la que vende más, de esas ventas qué refresco se vende más y que vendedor es el que ha vendido más de ese refresco en los últimos dos años.”

7. Consulta tipo Drill-Down

“Se desea saber qué ciudad tiene más órdenes de compra y quién es su cliente que más refrescos compra.”

8. Consulta tipo Drill-Down

“La empresa quiere conocer el desempeño de sus vendedores respecto a sus cuotas mensuales asignadas para garantizar que se estén cumpliendo sus metas, para que de lo contrario se puedan tomar acciones para mejorar.”

Recomendaciones y Conclusiones Generales:

1. **Producto estrella:** Coca Cola Regular 1L en GUADALAJARA
2. **Vendedor principal:** Fernando Pérez Ortiz ha generado \$41,348 con 5,579 unidades
3. **Cliente Principal:** Oxxo México – 90,147 unidades
4. **Oportunidades estacionales:** Agosto y Mayo
5. **¿Sidral Mundet?** \$161,943
6. **Revaluación del equipo de ventas** – cumplimiento cuotas
7. **Caída en ventas 2024** – investigación → causas externas
8. **Mercado principal:** Jalisco





Dashboard

<https://ck5d931f0lqa8fy.usqlikcloud.com/sense/app/542722c5-d767-4db4-855d-ee2dd84a87ef>

Implementación Columnar

```
cursodbe=> CREATE DATABASE proyectococacola_columnar;
CREATE DATABASE
cursodbe=> \c proyectococacola_columnar
Ahora está conectado a la base de datos «proyectococacola_columnar» con el usuario «alumno06».
proyectococacola_columnar=> CREATE EXTENSION IF NOT EXISTS cstore_fdw;
[NOTICE: la extensión «cstore_fdw» ya existe, omitiendo
CREATE EXTENSION
proyectococacola_columnar=> CREATE SERVER dw_columnar FOREIGN DATA WRAPPER cstore_fdw;
[CREATE SERVER
proyectococacola_columnar=>
```



Solo la tabla de hecho: ventas se hizo columnar.
→ velocidad funciones agregadas
→ menos espacio

```
proyectococacola_columnar=> CREATE FOREIGN TABLE ventas (
    ciudad_id INT,
    num_trabajador INT,
    refresco_id INT,
    rfc_cliente CHAR(13),
    rfc_tienda CHAR(13),
    dia_id INT,
    cantidad_venta INT,
    venta_$ MONEY
)
[SERVER dw_columnar;
CREATE FOREIGN TABLE
```

¡Muchas Gracias!