

Universidad Mayor de San Andrés
Facultad de Ciencias Puras y Naturales

Práctica Cassandra

DAT 251 - Base de datos 3

Docente:

Lic. Celia Elena Tarquino Peralta

Estudiante:

Gabriel Muñoz Marcelo Callisaya

Dupla:

23

Fecha de entrega:

30 de abril del 2025

La Paz - Bolivia



Informe de Práctica: Gestor NoSQL Apache Cassandra

Ejercicio 1: Diseño de un Esquema de Base de Datos

Objetivo

Diseñar un esquema en Apache Cassandra para una aplicación de seguimiento de pedidos en tiempo real, optimizado para tres consultas específicas: pedidos de un cliente por rango de fechas, detalles de un pedido específico, y pedidos por estado.

Diseño del Esquema

El esquema se define en el keyspace `pedidos_app` con seis tablas para soportar las consultas iniciales y las optimizaciones posteriores. A continuación, se presentan todas las tablas, sus columnas, claves primarias, y propósito.

Tabla `pedidos_por_cliente`

- Propósito: Soporta consultas de pedidos de un cliente en un rango de fechas.
- Columnas:
 - `nombre_cliente` (TEXT): Nombre del cliente (clave de partición).
 - `fecha_pedido` (TIMESTAMP): Fecha y hora del pedido (clave de clustering).
 - `id_pedido` (UUID): Identificador único del pedido (clave de clustering).
 - `estado` (TEXT): Estado del pedido (pendiente, enviado, entregado).
 - `productos` (LIST<FROZEN<MAP<TEXT, INT>>>): Lista de mapas con ID de producto y cantidad.
- Clave primaria: (`nombre_cliente`, `fecha_pedido`, `id_pedido`) CON `fecha_pedido` ordenado DESC.
- CQL:

```
CREATE TABLE IF NOT EXISTS pedidos_app.pedidos_por_cliente (  
    nombre_cliente TEXT,  
    fecha_pedido TIMESTAMP,  
    id_pedido UUID,  
    estado TEXT,  
    productos LIST<FROZEN<MAP<TEXT, INT>>>,  
    PRIMARY KEY (nombre_cliente, fecha_pedido, id_pedido)  
) WITH CLUSTERING ORDER BY (fecha_pedido DESC);
```

Tabla `pedidos_por_id`

- Propósito: Permite consultar los detalles de un pedido específico por su ID.
- Columnas:
 - `id_pedido` (UUID): Identificador único del pedido (clave de partición).
 - `nombre_cliente` (TEXT): Nombre del cliente.
 - `fecha_pedido` (TIMESTAMP): Fecha y hora del pedido.
 - `estado` (TEXT): Estado del pedido.
 - `productos` (LIST<FROZEN<MAP<TEXT, INT>>>): Lista de productos.
- Clave primaria: (`id_pedido`).
- CQL:

```
CREATE TABLE IF NOT EXISTS pedidos_app.pedidos_por_id (
  id_pedido UUID,
  nombre_cliente TEXT,
  fecha_pedido TIMESTAMP,
  estado TEXT,
  productos LIST<FROZEN<MAP<TEXT, INT>>>,
  PRIMARY KEY (id_pedido)
);
```

Tabla pedidos_por_estado

- Propósito: Soporta consultas de pedidos por estado (por ejemplo, todos los entregados).
- Columnas: Igual que pedidos_por_cliente, pero con estado como clave de partición.
- Clave primaria: (estado, fecha_pedido, id_pedido) CON fecha_pedido ordenado DESC.
- CQL:

```
CREATE TABLE IF NOT EXISTS pedidos_app.pedidos_por_estado (
  estado TEXT,
  fecha_pedido TIMESTAMP,
  id_pedido UUID,
  nombre_cliente TEXT,
  productos LIST<FROZEN<MAP<TEXT, INT>>>,
  PRIMARY KEY (estado, fecha_pedido, id_pedido)
) WITH CLUSTERING ORDER BY (fecha_pedido DESC);
```

Tabla compras_por_producto

- Propósito: Registra compras de productos para el sistema de recomendaciones.
- Columnas:
 - producto_id (TEXT): ID del producto (clave de partición).
 - fecha_compra (TIMESTAMP): Fecha de la compra (clave de clustering).
 - cliente (TEXT): Nombre del cliente (clave de clustering).
 - cantidad (INT): Cantidad comprada.
- Clave primaria: (producto_id, fecha_compra, cliente) CON fecha_compra ordenado DESC.
- CQL:

```
CREATE TABLE IF NOT EXISTS pedidos_app.compras_por_producto (
  producto_id TEXT,
  fecha_compra TIMESTAMP,
  cliente TEXT,
  cantidad INT,
  PRIMARY KEY (producto_id, fecha_compra, cliente)
) WITH CLUSTERING ORDER BY (fecha_compra DESC);
```

Tabla pedidos_por_estado_fecha

- Propósito: Optimización para consultas de pedidos por estado y rango de fechas, eliminando ALLOW FILTERING.
- Columnas: Igual que pedidos_por_estado.
- Clave primaria: (estado, fecha_pedido, id_pedido) CON fecha_pedido ordenado DESC.

- CQL:

```
CREATE TABLE IF NOT EXISTS pedidos_app.pedidos_por_estado_fecha (
    estado TEXT,
    fecha_pedido TIMESTAMP,
    id_pedido UUID,
    nombre_cliente TEXT,
    productos LIST<FROZEN<MAP<TEXT, INT>>>,
    PRIMARY KEY (estado, fecha_pedido, id_pedido)
) WITH CLUSTERING ORDER BY (fecha_pedido DESC);
```

Tabla compras_por_cliente

- Propósito: Optimización para consultas de compras de un cliente, eliminando índices secundarios.
- Columnas:
 - cliente (TEXT): Nombre del cliente (clave de partición).
 - fecha_compra (TIMESTAMP): Fecha de la compra (clave de clustering).
 - producto_id (TEXT): ID del producto (clave de clustering).
 - cantidad (INT): Cantidad comprada.
- Clave primaria: (cliente, fecha_compra, producto_id) con fecha_compra ordenado DESC.
- CQL:

```
CREATE TABLE IF NOT EXISTS pedidos_app.compras_por_cliente (
    cliente TEXT,
    fecha_compra TIMESTAMP,
    producto_id TEXT,
    cantidad INT,
    PRIMARY KEY (cliente, fecha_compra, producto_id)
) WITH CLUSTERING ORDER BY (fecha_compra DESC);
```

Justificación del Diseño

El diseño sigue las mejores prácticas de Cassandra:

- Denormalización: Cada tabla duplica datos para evitar JOINS, optimizando consultas específicas.
- Orientación a consultas: Las claves primarias están diseñadas para soportar patrones de acceso directos (por cliente, ID, estado, o producto).
- Particionamiento eficiente: nombre_cliente, estado, producto_id, y cliente distribuyen datos uniformemente.
- Ordenamiento: fecha_pedido y fecha_compra ordenados DESC aseguran acceso rápido a los datos más recientes.
- Estructura de datos: LIST<FROZEN<MAP<TEXT, INT>>> permite almacenar múltiples productos con cantidades, manteniendo integridad.
- Optimizaciones posteriores: pedidos_por_estado_fecha y compras_por_cliente eliminan ALLOW FILTERING e índices secundarios, mejorando el rendimiento.

Ejecución

El esquema se crea ejecutando:

```
cqlsh -f scripts/schema.cql
```

La captura de la ejecución:

```
zinko@win11-5znk:~$ cqlsh -f publico/proyCassandra/scripts/schema.cql
```

Ejercicio 2: Importación y Consultas

Objetivo

Importar datos ficticios desde un CSV a las tablas diseñadas y ejecutar consultas para validar el esquema.

Implementación

Generación de Datos (generate_pedidos.py)

- Propósito: Crea un archivo CSV (data/pedidos.csv) con 100 pedidos ficticios.
- Funcionamiento:
 - Genera UUIDs para id_pedido.
 - Selecciona nombres de clientes de una lista predefinida (por ejemplo, JuanPerez, AnaRodriguez).
 - Crea fechas aleatorias en 2025 (formato YYYY-MM-DDTHH:MM:SSZ).
 - Asigna estados (pendiente, enviado, entregado) con distribución uniforme.
 - Genera listas de productos (1-5 productos por pedido, con IDs prod1 a prod20 y cantidades de 1 a 10) en formato JSON.
- Salida: data/pedidos.csv con columnas: id_pedido, nombre_cliente, fecha_pedido, estado, productos.
- Ejecución:

```
python scripts/generate_pedidos.py
```

Captura:

```
(venvCassandra) zinko@win11-5znk:~/publico/proyCassandra$ python scripts/generate_pedidos.py
Archivo CSV generado exitosamente en: /home/zinko/publico/proyCassandra/data/pedidos.csv
(venvCassandra) zinko@win11-5znk:~/publico/proyCassandra$ |
```

Importación (import_pedidos.py)

- Propósito: Importa los datos de pedidos.csv a las tablas pedidos_por_cliente, pedidos_por_id, y pedidos_por_estado.
- Funcionamiento:
 - Lee pedidos.csv usando la biblioteca csv.
 - Conecta a Cassandra usando el driver cassandra-driver.
 - Usa sentencias preparadas para insertar datos en las tres tablas.
 - Agrupa inserciones en batches para mejorar la eficiencia.
 - Convierte fechas a formato TIMESTAMP y productos a LIST<FROZEN<MAP<TEXT, INT>>>.
- Ejecución:

```
python scripts/import_pedidos.py
```

Captura:

```
(venvCassandra) zinko@win11-5znk:~/publico/proyCassandra$ python scripts/import_pedidos.py
Conexión exitosa a Cassandra
Importación completada. 100 filas procesadas exitosamente.
```

Consultas (consultas.cql)

- Propósito: Valida el esquema con tres consultas específicas.
- Consultas:

1. Pedidos de un cliente en el último mes:

- ▶ Propósito: Obtener los pedidos de JuanPerez desde el 1 de marzo de 2025.
- ▶ CQL:

```
SELECT *
FROM pedidos_app.pedidos_por_cliente
WHERE nombre_cliente = 'JuanPerez'
AND fecha_pedido >= '2025-03-01T00:00:00Z';
```

- ▶ Resultado:

```
cqlsh> SELECT *
... FROM pedidos_app.pedidos_por_cliente
... WHERE nombre_cliente = 'JuanPerez'
... AND fecha_pedido >= '2025-03-01T00:00:00Z';
... AND fecha_pedido <= '2025-04-30T23:59:59Z';
```

| nombre_cliente | fecha_pedido | id_pedido | estado | productos |
|----------------|---------------------------------|--------------------------------------|-----------|---|
| JuanPerez | 2025-04-29 09:24:21.000000+0000 | 51dc7379-515b-4a72-993a-4013770192a6 | enviado | [{'prod4': 7}, {'prod9': 4}] |
| JuanPerez | 2025-04-17 09:19:02.000000+0000 | eb35ca29-448e-4520-a914-e61fad219de8 | pendiente | [{'prod12': 1}, {'prod10': 6}, {'prod17': 5}, {'prod16': 9}, {'prod15': 5}] |
| JuanPerez | 2025-04-03 00:39:28.000000+0000 | 54df5762-3e41-486d-9965-9050edf07108 | enviado | [{'prod2': 6}, {'prod12': 6}, {'prod9': 5}, {'prod1': 5}] |

(3 rows)

2. Productos de un pedido específico:

- ▶ Propósito: Obtener los productos de un pedido con un id_pedido específico (por ejemplo, 01749128-37ff-4939-af74-17188af036dd).
- ▶ CQL:

```
SELECT id_pedido, productos
FROM pedidos_app.pedidos_por_id
WHERE id_pedido = '01749128-37ff-4939-af74-17188af036dd';
```

- ▶ Resultado:

```
cqlsh> SELECT id_pedido, productos
... FROM pedidos_app.pedidos_por_id
... WHERE id_pedido = 49c90f56-3afe-4c87-af61-2dd4032cc72f;
```

| id_pedido | productos |
|--------------------------------------|--|
| 49c90f56-3afe-4c87-af61-2dd4032cc72f | [{'prod13': 10}, {'prod15': 4}, {'prod5': 6}, {'prod19': 5}] |

(1 rows)

3. Pedidos entregados en un rango de fechas:

- ▶ Propósito: Listar pedidos con estado entregado entre el 1 de enero y el 30 de abril de 2025.
- ▶ CQL:

```
SELECT *
FROM pedidos_app.pedidos_por_estado
WHERE estado = 'entregado'
AND fecha_pedido >= '2025-01-01T00:00:00Z'
AND fecha_pedido <= '2025-04-30T23:59:59Z'
ALLOW FILTERING;
```

- ▶ Resultado:

```
cqlsh> SELECT id_pedido, productos
... FROM pedidos_app.pedidos_por_estado
... WHERE id_pedido = 49c90f56-3afe-4c87-af61-2dd4032cc72f;
```

| id_pedido | productos |
|--------------------------------------|--|
| 49c90f56-3afe-4c87-af61-2dd4032cc72f | [{'prod13': 10}, {'prod15': 4}, {'prod5': 6}, {'prod19': 3}] |

```
(1 rows)
cqlsh>
cqlsh>
cqlsh> SELECT *
... FROM pedidos_app.pedidos_por_estado
... WHERE estado = 'entregado'
... AND Fecha_pedido >= '2025-01-01T00:00:00Z'
... AND Fecha_pedido <= '2025-04-30T23:59:59Z'
... ALLOW FILTERING;
```

| estado | fecha_pedido | id_pedido | nombre_cliente | productos |
|-----------|---------------------------------|--------------------------------------|-----------------|--|
| entregado | 2025-04-30 04:34:26.000000+0000 | f054a2e4-57e9-45f1-af85-5a536d46489c | MariaGomez | [{'prod8': 7}, {'prod1': 1}, {'prod5': 6}, {'prod17': 3}, {'prod3': 3}] |
| entregado | 2025-04-29 11:14:36.000000+0000 | 861c91d3-2481-4ce5-b739-e2d7d28f64ef | SofiaFernandez | [{'prod20': 8}] |
| entregado | 2025-04-22 20:30:08.000000+0000 | 9090c247-497a-4b53-9810-b9c7ce83f1c3 | AnaRodriguez | [{'prod1': 1}, {'prod6': 9}] |
| entregado | 2025-04-20 07:37:08.000000+0000 | 948cc1d7-0782-4ba8-aa30-5466af4a4a90 | MariaGomez | [{'prod20': 6}] |
| entregado | 2025-04-20 05:35:27.000000+0000 | 7e4ae06a-037c-4ac4-94cc-b1d8e8c3b1e1 | SofiaFernandez | [{'prod5': 2}, {'prod16': 4}] |
| entregado | 2025-04-19 18:19:43.000000+0000 | de852f12-b81f-4004-8048-98a7d06fcl6d | MiguelTorres | [{'prod17': 1}, {'prod5': 5}, {'prod1': 10}] |
| entregado | 2025-04-08 10:23:11.000000+0000 | 964e7670-238c-4b68-af7d-3d0fa11922f4 | MariaGomez | [{'prod1': 4}, {'prod15': 8}, {'prod19': 3}] |
| entregado | 2025-04-03 10:22:30.000000+0000 | f091e174-7d16-4b01-b9da-9328a5c6f663 | SofiaFernandez | [{'prod3': 7}, {'prod10': 8}, {'prod4': 2}, {'prod16': 8}] |
| entregado | 2025-03-24 23:43:25.000000+0000 | d4953d9d-2ac1-4513-b5ad-c148db5ab2a2 | MiguelTorres | [{'prod18': 7}, {'prod12': 7}, {'prod8': 6}, {'prod1': 8}] |
| entregado | 2025-03-18 15:53:06.000000+0000 | e9338129-2dfe-4945-887d-1a085a5abb2d | SantiagoAbascal | [{'prod9': 5}, {'prod14': 5}] |
| entregado | 2025-02-19 18:30:59.000000+0000 | ea883fee-4c29-42fb-b878-c6bd3eca2892 | LauraMartinez | [{'prod9': 5}, {'prod14': 5}] |
| entregado | 2025-02-18 10:23:11.000000+0000 | 51ecdafe-8817-4fab-07f9-f500ea30cb61 | RobertoJimenez | [{'prod12': 9}, {'prod6': 9}, {'prod14': 1}, {'prod15': 9}, {'prod20': 6}] |
| entregado | 2025-02-17 03:56:20.000000+0000 | 103d8f45-e915-4395-a1db-b5fc4051494b | DavidGarcia | [{'prod6': 7}, {'prod9': 5}] |
| entregado | 2025-02-17 03:56:20.000000+0000 | f0b1b1c1-b7fd-440b-9486-d4dd9338bc74 | RobertoJimenez | [{'prod6': 7}, {'prod9': 5}] |
| entregado | 2025-01-28 00:51:26.000000+0000 | 3888bb0d-57b8-4160-918b-d3d062cee16e | MiguelTorres | [{'prod20': 10}, {'prod9': 2}, {'prod10': 3}, {'prod15': 5}] |
| entregado | 2025-01-13 10:14:25.000000+0000 | 20cb54bd-b9e0-4423-b374-7b18f94d633b | SofiaFernandez | [{'prod15': 7}, {'prod14': 1}, {'prod18': 10}, {'prod11': 6}] |
| entregado | 2025-01-01 01:14:36.000000+0000 | b5521700-6cf2-45c9-85c3-c087bba06426 | AnaRodriguez | [{'prod10': 5}, {'prod6': 3}] |

```
(17 rows)
```

- Ejecución de las tres consultas en conjunto:

`cqlsh -f scripts/consultas.cql`

```
zinko@win11-52nw:~/publico/proyCassandra$ cqlsh -f scripts/consultas.cql
```

| nombre_cliente | fecha_pedido | id_pedido | estado | productos |
|----------------|---------------------------------|--------------------------------------|-----------|---|
| JuanPerez | 2025-04-29 09:24:21.000000+0000 | 51dc7379-515b-4a72-993a-4013770193a6 | enviado | [{'prod4': 7}, {'prod9': 4}] |
| JuanPerez | 2025-04-17 09:19:02.000000+0000 | eb35ce29-448e-4528-a914-e61fad2196eb | pendiente | [{'prod12': 1}, {'prod10': 6}, {'prod17': 5}, {'prod16': 9}, {'prod15': 5}] |
| JuanPerez | 2025-04-03 00:39:28.000000+0000 | 54df5762-3e41-406d-9965-90566df07108 | enviado | [{'prod2': 6}, {'prod12': 5}, {'prod9': 5}, {'prod1': 5}] |

```
(3 rows)
```

| id_pedido | productos |
|--------------------------------------|--|
| 49c90f56-3afe-4c87-af61-2dd4032cc72f | [{'prod13': 10}, {'prod15': 4}, {'prod5': 6}, {'prod19': 3}] |

```
(1 rows)
```

| estado | fecha_pedido | id_pedido | nombre_cliente | productos |
|-----------|---------------------------------|--------------------------------------|-----------------|--|
| entregado | 2025-04-30 04:34:26.000000+0000 | f054a2e4-57e9-45f1-af85-5a536d46489c | MariaGomez | [{'prod8': 7}, {'prod1': 1}, {'prod5': 6}, {'prod17': 3}, {'prod3': 3}] |
| entregado | 2025-04-29 11:14:36.000000+0000 | 861c91d3-2481-4ce5-b739-e2d7d28f64ef | SofiaFernandez | [{'prod20': 8}] |
| entregado | 2025-04-22 20:30:08.000000+0000 | 9090c247-497a-4b53-9810-b9c7ce83f1c3 | AnaRodriguez | [{'prod1': 1}, {'prod6': 9}] |
| entregado | 2025-04-20 07:37:08.000000+0000 | 948cc1d7-0782-4ba8-aa30-5466af4a4a90 | MariaGomez | [{'prod20': 6}] |
| entregado | 2025-04-20 05:35:27.000000+0000 | 7e4ae06a-037c-4ac4-94cc-b1d8e8c3b1e1 | SofiaFernandez | [{'prod5': 2}, {'prod16': 4}] |
| entregado | 2025-04-19 18:19:43.000000+0000 | de852f12-b81f-4004-8048-98a7d06fcl6d | MiguelTorres | [{'prod17': 1}, {'prod5': 5}, {'prod1': 10}] |
| entregado | 2025-04-08 10:23:11.000000+0000 | 964e7670-238c-4b68-af7d-3d0fa11922f4 | MariaGomez | [{'prod1': 4}, {'prod15': 8}, {'prod19': 3}] |
| entregado | 2025-04-03 10:22:30.000000+0000 | f091e174-7d16-4b01-b9da-9328a5c6f663 | SofiaFernandez | [{'prod3': 7}, {'prod10': 8}, {'prod4': 2}, {'prod16': 8}] |
| entregado | 2025-03-24 23:43:25.000000+0000 | d4953d9d-2ac1-4513-b5ad-c148db5ab2a2 | MiguelTorres | [{'prod18': 7}, {'prod12': 7}, {'prod8': 6}, {'prod1': 8}] |
| entregado | 2025-03-18 15:53:06.000000+0000 | e9338129-2dfe-4945-887d-1a085a5abb2d | SantiagoAbascal | [{'prod9': 5}, {'prod14': 5}] |
| entregado | 2025-02-19 18:30:59.000000+0000 | ea883fee-4c29-42fb-b878-c6bd3eca2892 | LauraMartinez | [{'prod9': 5}, {'prod14': 5}] |
| entregado | 2025-02-18 10:23:11.000000+0000 | 51ecdafe-8817-4fab-07f9-f500ea30cb61 | RobertoJimenez | [{'prod12': 9}, {'prod6': 9}, {'prod14': 1}, {'prod15': 9}, {'prod20': 6}] |
| entregado | 2025-02-17 03:56:20.000000+0000 | 103d8f45-e915-4395-a1db-b5fc4051494b | DavidGarcia | [{'prod6': 7}, {'prod9': 5}] |
| entregado | 2025-02-17 03:56:20.000000+0000 | f0b1b1c1-b7fd-440b-9486-d4dd9338bc74 | RobertoJimenez | [{'prod6': 7}, {'prod9': 5}] |
| entregado | 2025-01-28 00:51:26.000000+0000 | 3888bb0d-57b8-4160-918b-d3d062cee16e | MiguelTorres | [{'prod20': 10}, {'prod9': 2}, {'prod10': 3}, {'prod15': 5}] |
| entregado | 2025-01-13 10:14:25.000000+0000 | 20cb54bd-b9e0-4423-b374-7b18f94d633b | SofiaFernandez | [{'prod15': 7}, {'prod14': 1}, {'prod18': 10}, {'prod11': 6}] |
| entregado | 2025-01-01 01:14:36.000000+0000 | b5521700-6cf2-45c9-85c3-c087bba06426 | AnaRodriguez | [{'prod10': 5}, {'prod6': 3}] |

```
(17 rows)
```

Resultados

La generación creó un CSV válido con 100 pedidos. La importación pobló las tablas correctamente, y las consultas devolvieron los resultados esperados: 3 pedidos para JuanPerez, 1 fila para el pedido específico, y 17 pedidos entregados.

Ejercicio 3: Solución Creativa (Sistema de Recomendaciones)

Objetivo

Diseñar un sistema de recomendaciones basado en los productos más comprados, con consultas para analizar patrones de compra.

Implementación

Diseño de Tabla (compras_por_producto)

- Definida en `schema.cql` (ver Ejercicio 1), permite consultar compras por producto o cliente.

Población (populate_recomendaciones.py)

- Propósito: Extrae datos de pedidos.csv e inserta compras individuales en compras_por_producto.
- Funcionamiento:
 - Lee pedidos.csv y descompone las listas de productos.
 - Para cada producto en un pedido, inserta una fila con producto_id, fecha_compra, cliente, y cantidad.
 - Usa sentencias preparadas para eficiencia.
 - Genera 328 filas a partir de los 100 pedidos.
- Ejecución:

`python scripts/populate_recomendaciones.py`

Captura:

```
(venvCassandra) zinko@win11-Sznk:~/publico/proyCassandra$ python scripts/populate_recomendaciones.py
Conexión exitosa a Cassandra
Importación completada. 288 compras procesadas exitosamente.
```

Consultas (recomendaciones.cql)

- Propósito: Analiza patrones de compra para recomendaciones.
- Consultas:
 1. Compras de un producto específico:
 - Propósito: Obtener todas las compras de prod1.
 - CQL:

```
SELECT fecha_compra, cliente, cantidad
FROM pedidos_app.compras_por_producto
WHERE producto_id = 'prod1';
```
 - Resultado:


```
cqlsh> SELECT fecha_compra, cliente, cantidad
... FROM pedidos_app.compras_por_producto
... WHERE producto_id = 'prod1';
```

| fecha_compra | cliente | cantidad |
|---------------------------------|-----------------|----------|
| 2025-04-30 04:34:26.000000+0000 | MariaGomez | 1 |
| 2025-04-22 20:30:08.000000+0000 | AnaRodriguez | 1 |
| 2025-04-19 18:19:43.000000+0000 | MiguelTorres | 10 |
| 2025-04-16 21:43:48.000000+0000 | MiguelTorres | 10 |
| 2025-04-07 08:37:19.000000+0000 | CarlosLopez | 3 |
| 2025-04-03 00:39:28.000000+0000 | JuanPerez | 5 |
| 2025-03-25 18:24:34.000000+0000 | MiguelTorres | 2 |
| 2025-03-18 15:53:06.000000+0000 | SantiagoAbascal | 8 |
| 2025-03-09 17:51:45.000000+0000 | IsabelRuiz | 7 |
| 2025-03-05 20:39:43.000000+0000 | IsabelRuiz | 5 |
| 2025-03-05 06:04:27.000000+0000 | RobertoJimenez | 1 |
| 2025-03-02 19:16:24.000000+0000 | RobertoJimenez | 4 |
| 2025-02-21 12:43:46.000000+0000 | MariaGomez | 3 |
| 2025-02-13 13:11:34.000000+0000 | AnaRodriguez | 2 |
| 2025-02-08 19:54:25.000000+0000 | CarlosLopez | 7 |
| 2025-02-01 21:23:13.000000+0000 | JuanPerez | 8 |
| 2025-01-17 19:02:19.000000+0000 | LauraMartinez | 3 |
| 2025-01-12 23:46:40.000000+0000 | JuanPerez | 2 |
| 2025-01-12 11:08:04.000000+0000 | MiguelTorres | 7 |
| 2025-01-01 21:22:08.000000+0000 | MiguelTorres | 1 |

(20 rows)

cqlsh>

2. Compras de un cliente específico:

- Propósito: Listar las compras de JuanPerez.
- Requiere un índice secundario:

```
CREATE INDEX IF NOT EXISTS idx_compras_cliente
ON pedidos_app.compras_por_producto(cliente);
```

- CQL:

```
SELECT producto_id, fecha_compra, cantidad
FROM pedidos_app.compras_por_producto
WHERE cliente = 'JuanPerez';
```

- Resultado:

```
cqlsh> SELECT producto_id, fecha_compra, cantidad
... FROM pedidos_app.compras_por_producto
... WHERE cliente = 'JuanPerez';
```

| producto_id | fecha_compra | cantidad |
|-------------|---------------------------------|----------|
| prod19 | 2025-03-22 23:54:50.000000+0000 | 9 |
| prod19 | 2025-01-08 05:56:06.000000+0000 | 10 |
| prod6 | 2025-03-22 23:54:50.000000+0000 | 10 |
| prod10 | 2025-04-17 09:19:02.000000+0000 | 6 |
| prod16 | 2025-04-17 09:19:02.000000+0000 | 9 |
| prod16 | 2025-03-22 23:54:50.000000+0000 | 1 |
| prod9 | 2025-04-29 09:24:21.000000+0000 | 4 |
| prod9 | 2025-04-03 00:39:28.000000+0000 | 5 |
| prod9 | 2025-02-27 19:07:10.000000+0000 | 10 |
| prod9 | 2025-01-08 05:56:06.000000+0000 | 10 |
| prod1 | 2025-04-03 00:39:28.000000+0000 | 5 |
| prod1 | 2025-02-01 21:23:13.000000+0000 | 8 |
| prod1 | 2025-01-12 23:46:40.000000+0000 | 2 |
| prod15 | 2025-04-17 09:19:02.000000+0000 | 5 |
| prod15 | 2025-03-22 23:54:50.000000+0000 | 1 |
| prod15 | 2025-01-08 05:56:06.000000+0000 | 8 |
| prod11 | 2025-03-12 13:38:51.000000+0000 | 9 |
| prod7 | 2025-02-03 23:17:52.000000+0000 | 9 |
| prod7 | 2025-01-14 01:34:49.000000+0000 | 3 |
| prod7 | 2025-01-08 05:56:06.000000+0000 | 2 |
| prod5 | 2025-01-14 01:34:49.000000+0000 | 6 |
| prod5 | 2025-01-12 23:46:40.000000+0000 | 8 |
| prod4 | 2025-04-29 09:24:21.000000+0000 | 7 |
| prod2 | 2025-04-03 00:39:28.000000+0000 | 6 |
| prod2 | 2025-01-18 06:30:41.000000+0000 | 3 |
| prod17 | 2025-04-17 09:19:02.000000+0000 | 5 |
| prod12 | 2025-04-17 09:19:02.000000+0000 | 1 |
| prod12 | 2025-04-03 00:39:28.000000+0000 | 5 |
| prod12 | 2025-02-03 23:17:52.000000+0000 | 1 |
| prod12 | 2025-01-14 01:34:49.000000+0000 | 5 |
| prod20 | 2025-03-22 23:54:50.000000+0000 | 1 |

(31 rows)

3. Compras en un rango de fechas:

- Propósito: Listar compras de prod1 entre el 1 de enero y el 30 de abril de 2025.
- CQL:

```
SELECT producto_id, fecha_compra, cliente, cantidad
FROM pedidos_app.compras_por_producto
WHERE producto_id = 'prod1'
  AND fecha_compra >= '2025-01-01'
  AND fecha_compra <= '2025-04-30';
```

- Resultado:

```
cqlsh> SELECT producto_id, fecha_compra, cliente, cantidad
... FROM pedidos_app.compras_por_producto
... WHERE producto_id = 'prod1'
... AND fecha_compra >= '2025-01-01'
... AND fecha_compra <= '2025-04-30';
```

| producto_id | fecha_compra | cliente | cantidad |
|-------------|---------------------------------|-----------------|----------|
| prod1 | 2025-04-22 20:30:08.000000+0000 | AnaRodriguez | 1 |
| prod1 | 2025-04-19 18:19:43.000000+0000 | MiguelTorres | 10 |
| prod1 | 2025-04-16 21:43:48.000000+0000 | MiguelTorres | 10 |
| prod1 | 2025-04-07 08:37:19.000000+0000 | CarlosLopez | 3 |
| prod1 | 2025-04-03 00:39:28.000000+0000 | JuanPerez | 5 |
| prod1 | 2025-03-25 18:24:34.000000+0000 | MiguelTorres | 2 |
| prod1 | 2025-03-18 15:53:06.000000+0000 | SantiagoAbascal | 8 |
| prod1 | 2025-03-09 17:51:45.000000+0000 | IsabelRuiz | 7 |
| prod1 | 2025-03-05 20:39:43.000000+0000 | IsabelRuiz | 5 |
| prod1 | 2025-03-05 06:04:27.000000+0000 | RobertoJimenez | 1 |
| prod1 | 2025-03-02 19:16:24.000000+0000 | RobertoJimenez | 4 |
| prod1 | 2025-02-21 12:43:46.000000+0000 | MariaGomez | 3 |
| prod1 | 2025-02-13 13:11:34.000000+0000 | AnaRodriguez | 2 |
| prod1 | 2025-02-08 19:54:25.000000+0000 | CarlosLopez | 7 |
| prod1 | 2025-02-01 21:23:13.000000+0000 | JuanPerez | 8 |
| prod1 | 2025-01-17 19:02:19.000000+0000 | LauraMartinez | 3 |
| prod1 | 2025-01-12 23:46:40.000000+0000 | JuanPerez | 2 |
| prod1 | 2025-01-12 11:08:04.000000+0000 | MiguelTorres | 7 |
| prod1 | 2025-01-01 21:22:08.000000+0000 | MiguelTorres | 1 |

(19 rows)

- Ejecución de las tres consultas de recomendacion en conjunto:

```
cqlsh -f scripts/recomendaciones.cql
```

```
zinko@win11-5znk:~/publico/proyCassandra$ cqlsh -f scripts/recomendaciones.cql
```

| fecha_compra | cliente | cantidad |
|---------------------------------|-----------------|----------|
| 2025-04-30 04:34:26.000000+0000 | MariaGomez | 1 |
| 2025-04-22 20:30:08.000000+0000 | AnaRodriguez | 1 |
| 2025-04-19 18:19:43.000000+0000 | MiguelTorres | 10 |
| 2025-04-16 21:43:48.000000+0000 | MiguelTorres | 10 |
| 2025-04-07 08:37:19.000000+0000 | CarlosLopez | 3 |
| 2025-04-03 00:39:28.000000+0000 | JuanPerez | 5 |
| 2025-03-25 18:24:34.000000+0000 | MiguelTorres | 2 |
| 2025-03-18 15:53:06.000000+0000 | SantiagoAbascal | 8 |
| 2025-03-09 17:51:45.000000+0000 | IsabelRuiz | 7 |
| 2025-03-05 20:39:43.000000+0000 | IsabelRuiz | 5 |
| 2025-03-05 06:04:27.000000+0000 | RobertoJimenez | 1 |
| 2025-03-02 19:16:24.000000+0000 | RobertoJimenez | 4 |
| 2025-02-21 12:43:46.000000+0000 | MariaGomez | 3 |
| 2025-02-13 13:11:34.000000+0000 | AnaRodriguez | 2 |
| 2025-02-08 19:54:25.000000+0000 | CarlosLopez | 7 |
| 2025-02-01 21:23:13.000000+0000 | JuanPerez | 8 |
| 2025-01-17 19:02:19.000000+0000 | LauraMartinez | 3 |
| 2025-01-12 23:46:40.000000+0000 | JuanPerez | 2 |
| 2025-01-12 11:08:04.000000+0000 | MiguelTorres | 7 |
| 2025-01-01 21:22:08.000000+0000 | MiguelTorres | 1 |

(20 rows)

| producto_id | fecha_compra | cantidad |
|-------------|---------------------------------|----------|
| prod19 | 2025-03-22 23:54:50.000000+0000 | 9 |
| prod19 | 2025-01-08 05:56:06.000000+0000 | 10 |
| prod6 | 2025-03-22 23:54:50.000000+0000 | 10 |
| prod10 | 2025-04-17 09:19:02.000000+0000 | 6 |
| prod16 | 2025-04-17 09:19:02.000000+0000 | 9 |
| prod16 | 2025-03-22 23:54:50.000000+0000 | 1 |
| prod9 | 2025-04-29 09:24:21.000000+0000 | 4 |
| prod9 | 2025-04-03 00:39:28.000000+0000 | 5 |
| prod9 | 2025-02-27 19:07:10.000000+0000 | 10 |
| prod9 | 2025-01-08 05:56:06.000000+0000 | 10 |
| prod1 | 2025-04-03 00:39:28.000000+0000 | 5 |
| prod1 | 2025-02-01 21:23:13.000000+0000 | 8 |
| prod1 | 2025-01-12 23:46:40.000000+0000 | 2 |
| prod15 | 2025-04-17 09:19:02.000000+0000 | 5 |
| prod15 | 2025-03-22 23:54:50.000000+0000 | 1 |
| prod15 | 2025-01-08 05:56:06.000000+0000 | 8 |
| prod11 | 2025-03-12 13:38:51.000000+0000 | 9 |
| prod7 | 2025-02-03 23:17:52.000000+0000 | 9 |
| prod7 | 2025-01-14 01:34:49.000000+0000 | 3 |
| prod7 | 2025-01-08 05:56:06.000000+0000 | 2 |
| prod5 | 2025-01-14 01:34:49.000000+0000 | 6 |
| prod5 | 2025-01-12 23:46:40.000000+0000 | 8 |
| prod4 | 2025-04-29 09:24:21.000000+0000 | 7 |
| prod2 | 2025-04-03 00:39:28.000000+0000 | 6 |
| prod2 | 2025-01-18 06:30:41.000000+0000 | 3 |
| prod17 | 2025-04-17 09:19:02.000000+0000 | 5 |
| prod12 | 2025-04-17 09:19:02.000000+0000 | 1 |
| prod12 | 2025-04-03 00:39:28.000000+0000 | 5 |
| prod12 | 2025-02-03 23:17:52.000000+0000 | 1 |
| prod12 | 2025-01-14 01:34:49.000000+0000 | 5 |
| prod20 | 2025-03-22 23:54:50.000000+0000 | 1 |

(31 rows)

| producto_id | fecha_compra | cliente | cantidad |
|-------------|---------------------------------|-----------------|----------|
| prod1 | 2025-04-22 20:30:08.000000+0000 | AnaRodriguez | 1 |
| prod1 | 2025-04-19 18:19:43.000000+0000 | MiguelTorres | 10 |
| prod1 | 2025-04-16 21:43:48.000000+0000 | MiguelTorres | 10 |
| prod1 | 2025-04-07 08:37:19.000000+0000 | CarlosLopez | 3 |
| prod1 | 2025-04-03 00:39:28.000000+0000 | JuanPerez | 5 |
| prod1 | 2025-03-25 18:24:34.000000+0000 | MiguelTorres | 2 |
| prod1 | 2025-03-18 15:53:06.000000+0000 | SantiagoAbascal | 8 |
| prod1 | 2025-03-09 17:51:45.000000+0000 | IsabelRuiz | 7 |
| prod1 | 2025-03-05 20:39:43.000000+0000 | IsabelRuiz | 5 |
| prod1 | 2025-03-05 06:04:27.000000+0000 | RobertoJimenez | 1 |
| prod1 | 2025-03-02 19:16:24.000000+0000 | RobertoJimenez | 4 |
| prod1 | 2025-02-21 12:43:46.000000+0000 | MariaGomez | 3 |
| prod1 | 2025-02-13 13:11:34.000000+0000 | AnaRodriguez | 2 |
| prod1 | 2025-02-08 19:54:25.000000+0000 | CarlosLopez | 7 |
| prod1 | 2025-02-01 21:23:13.000000+0000 | JuanPerez | 8 |
| prod1 | 2025-01-17 19:02:19.000000+0000 | LauraMartinez | 3 |
| prod1 | 2025-01-12 23:46:40.000000+0000 | JuanPerez | 2 |
| prod1 | 2025-01-12 11:08:04.000000+0000 | MiguelTorres | 7 |
| prod1 | 2025-01-01 21:22:08.000000+0000 | MiguelTorres | 1 |

(19 rows)

```
zinko@win11-5znk:~/publico/proyCassandra$
```

Agregaciones (agregaciones_recomendaciones.py)

- Propósito: Calcula los productos más comprados para recomendaciones.
- Funcionamiento:
 - Lee datos de `compras_por_producto`.
 - Calcula:
 1. Top 5 productos generales (suma de cantidades por `producto_id`).
 2. Top 5 productos de JuanPerez.
 3. Top 5 en un rango de fechas (1 de enero a 30 de abril de 2025).
 - Usa consultas CQL y procesamiento en Python para ordenar resultados.
- Resultados:
 - Top 5 generales: `prod10: 144, prod4: 139, prod5: 128, prod19: 111, prod13: 110`.
 - Top 5 de JuanPerez: `prod5: 29, prod6: 23, prod2: 15, prod8: 15, prod15: 14`.
 - Top 5 en rango de fechas: Igual que generales.
- Ejecución:

`python scripts/agregaciones_recomendaciones.py`

Captura:

```
(venvCassandra) zinko@win11-5znk:~/publico/proyCassandra$ python scripts/agregaciones_recomendaciones.py

Top 5 productos más comprados en general:
Producto: prod10, Cantidad total: 236
Producto: prod20, Cantidad total: 236
Producto: prod5, Cantidad total: 222
Producto: prod18, Cantidad total: 212
Producto: prod4, Cantidad total: 209

Top 5 productos más comprados por JuanPerez:
Producto: prod6, Cantidad total: 33
Producto: prod2, Cantidad total: 31
Producto: prod15, Cantidad total: 30
Producto: prod5, Cantidad total: 30
Producto: prod20, Cantidad total: 26

Top 5 productos más vendidos entre 2025-01-01 00:00:00 y 2025-04-30 00:00:00:
Producto: prod10, Cantidad total: 236
Producto: prod20, Cantidad total: 236
Producto: prod5, Cantidad total: 222
Producto: prod18, Cantidad total: 212
Producto: prod4, Cantidad total: 209
```

Relevancia

El sistema permite a la tienda en línea:

- Identificar productos populares para promociones.
- Personalizar recomendaciones basadas en el historial de un cliente.
- Analizar tendencias temporales para ajustar inventarios.

La tabla `compras_por_producto` es eficiente para consultas por producto, y el índice secundario soporta consultas por cliente, aunque se optimizó en el Ejercicio 4.

Ejercicio 4: Investigación y Mejora de Rendimiento

Objetivo

Investigar prácticas de optimización en Cassandra, medir tiempos de consultas, y mejorar el esquema y consultas.

Investigación

Se estudiaron las siguientes prácticas:

- Diseño orientado a consultas: Crear tablas específicas para cada consulta.
- Evitar ALLOW FILTERING: Usar tablas con claves primarias que soporten filtros directamente.
- Minimizar índices secundarios: Reemplazarlos con tablas denormalizadas.
- Denormalización: Duplicar datos para evitar consultas complejas.
- Particionamiento eficiente: Claves de partición que distribuyan datos uniformemente.
- Materialized views: Para consultas frecuentes con filtros predefinidos.
- Configuración de nodos: Ajustar replication_factor y estrategias como NetworkTopologyStrategy para clústeres grandes.

Implementación

Problemas Identificados

- Consulta 3 de consultas.cql: Usa ALLOW FILTERING para filtrar por fecha_pedido, lo que escanea particiones innecesariamente.
- Consulta 2 de recomendaciones.cql: Depende de un índice secundario (idx_compras_cliente), que es menos eficiente que una tabla dedicada.

Mejoras Aplicadas

1. Tabla pedidos_por_estado_fecha:
 - Definida en schema.cql (ver Ejercicio 1).
 - Elimina la necesidad de ALLOW FILTERING al usar estado y fecha_pedido en la clave primaria.
2. Tabla compras_por_cliente:
 - Definida en schema.cql.
 - Reemplaza el índice secundario, permitiendo consultas directas por cliente y fecha_compra.
3. Población (populate_optimizaciones.py):
 - Propósito: Pobra pedidos_por_estado_fecha y compras_por_cliente con datos de pedidos.csv.
 - Funcionamiento:
 - Lee pedidos.csv.
 - Inserta en pedidos_por_estado_fecha los mismos datos que en pedidos_por_estado.
 - Descompone productos y los inserta en compras_por_cliente (328 filas).
 - Usa sentencias preparadas y batches.
 - Ejecución:

`python scripts/populate_optimizaciones.py`

Captura:

```
(venvCassandra) zinko@win11-5znk:~/publico/proyCassandra$ python scripts/populate_optimizaciones.py
Conexión exitosa a Cassandra
Se insertaron 100 filas en pedidos_por_estado_fecha
Se insertaron 288 filas en compras_por_cliente

Migración completada exitosamente:
- Pedidos migrados: 100
- Compras migradas: 288
Conexión cerrada
```

4. Consultas Optimizadas (optimizaciones.cql):

• Consulta 3 optimizada:

- Propósito: Listar pedidos entregados entre el 1 de enero y el 30 de abril de 2025.

► CQL:

```
SELECT *
FROM pedidos_app.pedidos_por_estado_fecha
WHERE estado = 'entregado'
    AND fecha_pedido >= '2025-01-01T00:00:00Z'
    AND fecha_pedido <= '2025-04-30T23:59:59Z';
```

► Resultado:

```
cqlsh> SELECT *
... FROM pedidos_app.pedidos_por_estado_fecha
... WHERE estado = 'entregado'
... AND fecha_pedido >= '2025-01-01T00:00:00Z'
... AND fecha_pedido <= '2025-04-30T23:59:59Z';
```

| estado | fecha_pedido | id_pedido | nombre_cliente | productos |
|-----------|---------------------------------|---------------------------------------|-----------------|--|
| entregado | 2025-04-30 00:30:26.000000+0000 | 4854a24f-57e9-46f1-af85-5a536d66409e | MariaGomez | [{'prod8': 7}, {'prod1': 1}, {'prod5': 6}, {'prod17': 3}, {'prod3': 3}] |
| entregado | 2025-04-29 11:18:36.000000+0000 | 861c91d3-2481-4ce5-b739-e207d28f64ef | SofiaFernandez | [{'prod28': 8}] |
| entregado | 2025-04-22 20:30:06.000000+0000 | 989dc247-497a-4b53-9d10-b9c7ce83f1c3 | AnaRodriguez | [{'prod1': 1}, {'prod6': 9}] |
| entregado | 2025-04-20 07:37:00.000000+0000 | 048ce1d7-0782-48a8-aa38-5466afaa4a90 | MariaGomez | [{'prod28': 6}] |
| entregado | 2025-04-20 03:35:27.000000+0000 | 7e4aa86a-937c-4ac9-94c1-b1d6e8c301e1 | SofiaFernandez | [{'prod5': 2}, {'prod16': 4}] |
| entregado | 2025-04-19 18:19:43.000000+0000 | de852f12-b81f-4804-8948-98ad7d0fc1ed | MiguelTorres | [{'prod17': 1}, {'prod5': 5}, {'prod1': 10}] |
| entregado | 2025-04-08 10:23:11.000000+0000 | 964e7670-238c-4b68-af7d-3b0fall18922f | MariaGomez | [{'prod16': 4}, {'prod15': 8}, {'prod19': 8}] |
| entregado | 2025-04-03 10:22:30.000000+0000 | fd91e174-7d16-4d01-b9da-9328a5c6f763 | SofiaFernandez | [{'prod3': 7}, {'prod10': 8}, {'prod4': 2}, {'prod16': 8}] |
| entregado | 2025-03-24 23:43:25.000000+0000 | d403c09d-2a1c1-4612-bbae-c1a8dbabb2a2 | MiguelTorres | [{'prod8': 5}] |
| entregado | 2025-03-18 15:53:06.000000+0000 | e9338129-2d46-4945-887d-1a885a5bbbd2 | SantiagoAbascal | [{'prod18': 7}, {'prod12': 7}, {'prod8': 6}, {'prod1': 8}] |
| entregado | 2025-02-19 18:30:59.000000+0000 | ea883fee-4c29-42fb-b878-c6bd3eca2892 | LauraMartinez | [{'prod9': 5}, {'prod14': 5}] |
| entregado | 2025-02-18 19:14:19.000000+0000 | 51ecdafa-8617-47ab-8749-f508ea10cb61 | RobertoJimenez | [{'prod12': 9}, {'prod6': 9}, {'prod14': 1}, {'prod15': 9}, {'prod28': 6}] |
| entregado | 2025-02-17 03:56:20.000000+0000 | 10359fde-a915-4095-afdb-b5fc4021409a | DavidGarcia | [{'prod8': 7}] |
| entregado | 2025-02-09 21:14:05.000000+0000 | fb81d1c1-b7fd-448b-9486-d4dd9338bc74 | RobertoJimenez | [{'prod6': 7}, {'prod9': 5}] |
| entregado | 2025-01-20 00:51:26.000000+0000 | 3888bbbd-57b8-4168-918b-d3d862cee16a | MiguelTorres | [{'prod28': 10}, {'prod9': 2}, {'prod10': 3}, {'prod15': 5}] |
| entregado | 2025-01-13 10:14:25.000000+0000 | 28c8540a-b9e8-4023-b374-7018f9a0633b | SofiaFernandez | [{'prod15': 7}, {'prod14': 1}, {'prod18': 10}, {'prod1': 6}] |
| entregado | 2025-01-01 01:10:36.000000+0000 | b5521700-6cf2-45c9-85c3-c887bba06426 | AnaRodriguez | [{'prod10': 5}, {'prod6': 3}] |

(17 rows)

• Consulta 2 optimizada:

- Propósito: Listar compras de JuanPerez en 2025.

► CQL:

```
SELECT producto_id, fecha_compra, cantidad
FROM pedidos_app.compras_por_cliente
WHERE cliente = 'JuanPerez'
    AND fecha_compra >= '2025-01-01'
    AND fecha_compra <= '2025-04-30';
```

► Resultado:


```
cqlsh> SELECT producto_id, fecha_compra, cantidad
... FROM pedidos_app.compras_por_cliente
... WHERE cliente = 'JuanPerez'
... AND fecha_compra >= '2025-01-01'
... AND fecha_compra <= '2025-04-30';
```

| producto_id | fecha_compra | cantidad |
|-------------|---------------------------------|----------|
| prod4 | 2025-04-29 09:24:21.000000+0000 | 7 |
| prod9 | 2025-04-29 09:24:21.000000+0000 | 4 |
| prod10 | 2025-04-17 09:19:02.000000+0000 | 6 |
| prod12 | 2025-04-17 09:19:02.000000+0000 | 1 |
| prod15 | 2025-04-17 09:19:02.000000+0000 | 5 |
| prod16 | 2025-04-17 09:19:02.000000+0000 | 9 |
| prod17 | 2025-04-17 09:19:02.000000+0000 | 5 |
| prod1 | 2025-04-03 00:39:28.000000+0000 | 5 |
| prod12 | 2025-04-03 00:39:28.000000+0000 | 5 |
| prod2 | 2025-04-03 00:39:28.000000+0000 | 6 |
| prod9 | 2025-04-03 00:39:28.000000+0000 | 5 |
| prod15 | 2025-03-22 23:54:50.000000+0000 | 1 |
| prod16 | 2025-03-22 23:54:50.000000+0000 | 1 |
| prod19 | 2025-03-22 23:54:50.000000+0000 | 9 |
| prod20 | 2025-03-22 23:54:50.000000+0000 | 1 |
| prod6 | 2025-03-22 23:54:50.000000+0000 | 10 |
| prod11 | 2025-03-12 13:38:51.000000+0000 | 9 |
| prod9 | 2025-02-27 19:07:10.000000+0000 | 10 |
| prod12 | 2025-02-03 23:17:52.000000+0000 | 1 |
| prod7 | 2025-02-03 23:17:52.000000+0000 | 9 |
| prod1 | 2025-02-01 21:23:13.000000+0000 | 8 |
| prod2 | 2025-01-18 06:30:41.000000+0000 | 3 |
| prod12 | 2025-01-14 01:34:49.000000+0000 | 5 |
| prod5 | 2025-01-14 01:34:49.000000+0000 | 6 |
| prod7 | 2025-01-14 01:34:49.000000+0000 | 3 |
| prod1 | 2025-01-12 23:46:40.000000+0000 | 2 |
| prod5 | 2025-01-12 23:46:40.000000+0000 | 8 |
| prod15 | 2025-01-08 05:56:06.000000+0000 | 8 |
| prod19 | 2025-01-08 05:56:06.000000+0000 | 10 |
| prod7 | 2025-01-08 05:56:06.000000+0000 | 2 |
| prod9 | 2025-01-08 05:56:06.000000+0000 | 10 |

(31 rows)

- Ejecución de las dos consultas optimizadas en conjunto:

```
cqlsh -f scripts/optimizaciones.cql
```


| zinko@win11-s2mk:~/publico/proyCassandra\$ cqlsh -f scripts/optimizaciones.cql | | | | | |
|--|---------------------------------|--------------------------------------|-----------------|--|--|
| estado | fecha_pedido | id_pedido | nombre_cliente | productos | |
| entregado | 2025-04-30 00:34:36.000000+0000 | f05da2e4-57e9-45f1-af85-5a536d4b489c | MariaGomez | [{'prod8': 7}, {'prod1': 1}, {'prod5': 6}, {'prod17': 3}, {'prod3': 3}] | |
| entregado | 2025-04-29 11:14:36.000000+0000 | 861c91d3-2461-4ce5-b739-e2d7d28f64ef | SofiaFernandez | [{'prod20': 8}] | |
| entregado | 2025-04-22 20:30:08.000000+0000 | 989dc247-497a-4b53-9810-b9c7ce83f1c3 | AnaRodriguez | [{'prod1': 1}, {'prod6': 9}] | |
| entregado | 2025-04-20 07:37:08.000000+0000 | 048cc1d7-0782-40a8-aa38-5466af4a4a90 | MariaGomez | [{'prod20': 6}] | |
| entregado | 2025-04-20 05:35:27.000000+0000 | 7e4ae06a-037c-4ac4-94cc-b1d8e8c3b1c1 | SofiaFernandez | [{'prod5': 2}, {'prod16': 4}] | |
| entregado | 2025-04-19 10:19:43.000000+0000 | 6e82f212-b81f-4084-9048-98ad7d9f1ed | MiguelTorres | [{'prod17': 1}, {'prod5': 5}, {'prod1': 10}] | |
| entregado | 2025-04-08 10:23:11.000000+0000 | 964e7670-238c-4b68-af7d-3b0fa118922f | MariaGomez | [{'prod16': 4}, {'prod15': 8}, {'prod19': 8}] | |
| entregado | 2025-04-03 10:22:30.000000+0000 | fd91e174-7d16-4b01-b9da-9320a5c6ff63 | SofiaFernandez | [{'prod3': 7}, {'prod10': 8}, {'prod4': 2}, {'prod16': 8}] | |
| entregado | 2025-03-24 23:43:25.000000+0000 | dd953d0e-2ac1-4513-b5ad-c148db5ab2a2 | MiguelTorres | [{'prod18': 7}, {'prod12': 7}, {'prod8': 6}, {'prod5': 5}] | |
| entregado | 2025-03-19 15:53:06.000000+0000 | e9318129-2d66-4945-897d-1a85a5b0bd2d | SantiagoAbascal | [{'prod9': 5}, {'prod14': 5}] | |
| entregado | 2025-02-19 18:30:59.000000+0000 | ea883fee-4c29-42fb-b878-c6bd3eca2892 | LauraMartinez | [{'prod12': 9}, {'prod6': 9}, {'prod14': 1}, {'prod15': 9}, {'prod20': 6}] | |
| entregado | 2025-02-18 19:14:19.000000+0000 | 51ecdafa-8817-4fab-87f9-f500ea30cb61 | RobertoJimenez | [{'prod6': 7}, {'prod5': 5}] | |
| entregado | 2025-02-17 10:56:20.000000+0000 | 103d0f05-e915-4395-a1db-b5fc0451494b | DavidGarcia | [{'prod6': 7}, {'prod5': 5}] | |
| entregado | 2025-02-09 21:14:05.000000+0000 | f081b1c1-b7fd-448b-9486-d46dd938bc74 | RobertoJimenez | [{'prod20': 10}, {'prod9': 2}, {'prod10': 3}, {'prod15': 5}] | |
| entregado | 2025-01-28 00:51:26.000000+0000 | 3888bb0d-57b8-4160-918b-d3d062cee16e | MiguelTorres | [{'prod15': 7}, {'prod14': 1}, {'prod18': 10}, {'prod11': 6}] | |
| entregado | 2025-01-13 10:14:25.000000+0000 | 28cb54bd-b9e0-4423-b374-7b18f94d633b | SofiaFernandez | [{'prod10': 5}, {'prod6': 3}] | |
| entregado | 2025-01-01 01:14:36.000000+0000 | b5521700-6cf2-45c9-85c3-c087bba06426 | AnaRodriguez | | |
| (17 rows) | | | | | |
| producto_id | fecha_compra | cantidad | | | |
| prod4 | 2025-04-29 09:24:21.000000+0000 | 7 | | | |
| prod9 | 2025-04-29 09:24:21.000000+0000 | 4 | | | |
| prod10 | 2025-04-17 09:19:02.000000+0000 | 6 | | | |
| prod12 | 2025-04-17 09:19:02.000000+0000 | 1 | | | |
| prod15 | 2025-04-17 09:19:02.000000+0000 | 9 | | | |
| prod16 | 2025-04-17 09:19:02.000000+0000 | 5 | | | |
| prod17 | 2025-04-17 09:19:02.000000+0000 | 5 | | | |
| prod1 | 2025-04-03 00:39:28.000000+0000 | 5 | | | |
| prod12 | 2025-04-03 00:39:28.000000+0000 | 5 | | | |
| prod2 | 2025-04-03 00:39:28.000000+0000 | 6 | | | |
| prod9 | 2025-04-03 00:39:28.000000+0000 | 5 | | | |
| prod15 | 2025-03-22 23:54:50.000000+0000 | 1 | | | |
| prod16 | 2025-03-22 23:54:50.000000+0000 | 9 | | | |
| prod19 | 2025-03-22 23:54:50.000000+0000 | 1 | | | |
| prod20 | 2025-03-22 23:54:50.000000+0000 | 1 | | | |
| prod6 | 2025-03-22 23:54:50.000000+0000 | 10 | | | |
| prod11 | 2025-03-12 13:38:51.000000+0000 | 9 | | | |
| prod9 | 2025-02-27 19:07:10.000000+0000 | 10 | | | |
| prod12 | 2025-02-03 23:17:52.000000+0000 | 1 | | | |
| prod7 | 2025-02-03 23:17:52.000000+0000 | 9 | | | |
| prod1 | 2025-02-01 21:23:13.000000+0000 | 8 | | | |
| prod2 | 2025-01-18 06:30:41.000000+0000 | 3 | | | |
| prod12 | 2025-01-14 01:34:49.000000+0000 | 5 | | | |
| prod5 | 2025-01-14 01:34:49.000000+0000 | 6 | | | |
| prod7 | 2025-01-14 01:34:49.000000+0000 | 3 | | | |
| prod1 | 2025-01-12 23:46:40.000000+0000 | 2 | | | |
| prod5 | 2025-01-12 23:46:40.000000+0000 | 8 | | | |
| prod15 | 2025-01-08 05:56:06.000000+0000 | 8 | | | |
| prod19 | 2025-01-08 05:56:06.000000+0000 | 10 | | | |
| prod7 | 2025-01-08 05:56:06.000000+0000 | 2 | | | |
| prod9 | 2025-01-08 05:56:06.000000+0000 | 10 | | | |
| (31 rows) | | | | | |

Tiempos de Ejecución

Se usó TRACING ON en cqlsh para medir el rendimiento:

- Consulta 3 original (pedidos_por_estado): 3.147 ms.
- Consulta 3 optimizada (pedidos_por_estado_fecha): 1.624 ms (48% más rápida).
- Consulta 2 original (compras_por_producto con índice): 7.898 ms.
- Consulta 2 optimizada (compras_por_cliente): 4.215 ms (47% más rápida).

La captura de la ejecución es demasiado larga, pero se puede ver en detalle en el archivo trazado_completo.png.

Resultados y Conclusiones

Las optimizaciones lograron:

- Eliminación de ALLOW FILTERING: pedidos_por_estado_fecha permite filtros directos, reduciendo el tiempo de 3.147 ms a 1.624 ms.
- Reemplazo de índices secundarios: compras_por_cliente mejora el tiempo de 7.898 ms a 4.215 ms.
- Escalabilidad: Las mejoras serán más significativas con datasets grandes, ya que evitan escaneos completos.
- Limitaciones: El dataset pequeño (100 pedidos, 328 compras) limita las diferencias de tiempo, pero el diseño es robusto para entornos reales.

Recomendaciones

1. Materialized views: Crear vistas para consultas frecuentes, como pedidos por cliente y estado, para reducir duplicación manual.
2. Ajustar particionamiento: Usar NetworkTopologyStrategy y aumentar replication_factor en clústeres multi-nodo.
3. Monitoreo: Usar herramientas como nodetool para analizar el tamaño de particiones y evitar particiones demasiado grandes.

4. Caching: Habilitar caché de filas para consultas repetitivas.
5. Compresión: Configurar compresión de datos en tablas para reducir el uso de disco.