Optimización de Consultas Distribuidas en PostgreSQL

Laboratorio 15

Integrantes:

Fabrizzio Vilchez

Jeffrey Antonio Monja Castro

Universidad de Ingeniería y Tecnología

CS2042 Base de Datos II

Docente: Sanchez Enriquez, Heider Ysaias

2024-I

Contents

1 P1. Creación de Tablas Fragmentadas

Para este laboratorio se crearon dos tablas principales:

- **Venta**(IdVenta, DNI_Cliente, FechaVenta, CodLocal, ImporteTotal, IdEmpleado)
- **Reclamo**(IdReclamo, DNI_Cliente, FechaReclamo, CodLocal, Descripcion, Estado)

Estas tablas se van a fragmentar para posteriormente aplicar algoritmos de consultas distribuidas eficientes. Las consultas a realizar son las siguientes:

Consulta 1
Recuperar todas las tuplas de la tabla `Venta` ordenándolas en base al atributo `ImporteTotal` de manera descendente

```sql
SELECT *
FROM venta
ORDER BY ImporteTotal DESC;
```

Consulta 2
Recuperar todos los `DNI_Cliente` distintos de la tabla `Venta`

```sql
SELECT DISTINCT DNI_Cliente
FROM venta;
```

Consulta 3
Recuperar el `CodLocal` y el promedio de `ImporteTotal` por `CodLocal` ordenado por el `CodLocal` en la tabla `Venta`

```sql
SELECT CodLocal, AVG(ImporteTotal) AS promedio_importe
FROM venta
GROUP BY CodLocal
ORDER BY CodLocal;
```

Consulta 4
Recuperar todas las tuplas de la tabla `Reclamo` donde el `DNI_Cliente` se encuentre en la tabla `Venta`

```sql
SELECT R.*
FROM reclamo R
JOIN venta V
ON R.DNI_Cliente = V.DNI_Cliente;
```

1.1 Población de las tablas

Para poblar las tablas se utilizó un Script de Python para generar 1 millón de datos aleatorios para cada atributo con la ayuda de la librería faker, y también se insertaron en cada tabla con ayuda de la librería de psycopg2. Cabe resaltar que los datos se han insertado de manera uniforme.
Además, se realizaron algunas restricciones sobre los atributos para posteriormente realizar particiones adecuadas:

- Los Ids de las tablas son incrementales
- El atributo `DNI_Cliente` es un número de 8 dígitos entre 70000000 y 77500000.
- El atributo `FechaVenta` de la tabla `Venta` y `FechaReclamo` de la tabla `Reclamo` tienen valores entre 2000-01-01 y 2025-01-01.
- El atributo `CodLocal` son strings de la forma "LOC[00-12]".
- El atributo `ImporteTotal` de la tabla `Venta` es un entero entre 1 y 1000.

Script de Python:

```python
import psycopg2
from faker import Faker
import random
from datetime import date

# Configuracion de la conexion a la base de datos
conn = psycopg2.connect(
    host="localhost",
    database="db2",
    user="postgres",
    password="123"
)

cursor = conn.cursor()
schema = "lab15"
cursor.execute(f"SET search_path TO {schema}")

fake = Faker('es_ES')

ventas = []
reclamos = []

# 12 codigos diferentes para CodLocal
cod_locales = [f"LOC{str(i).zfill(2)}" for i in range(1, 13)]
start_date = date(2000, 1, 1)
end_date = date(2025, 1, 1)
estados = ['Pendiente', 'En Proceso', 'Cerrado']

# Generar datos aleatorios con id incremental
for i in range(1000000):
    dni_cliente = str(random.randint(70000000, 77500000))
    fecha_venta = fake.date_between(start_date=start_date, end_date=
        end_date)
    cod_local = random.choice(cod_locales)
    importe_total = random.randint(1, 1000)
    id_empleado = f"EMP{str(fake.random_number(digits=3, fix_len=True)
        ).zfill(3)}"

    ventas.append((i + 1, dni_cliente, fecha_venta, cod_local,
        importe_total, id_empleado))

    dni_cliente = str(random.randint(70000000, 79999999))
    fecha_reclamo = fake.date_between(start_date=start_date, end_date=
        end_date)
    cod_local = random.choice(cod_locales)
    descripcion = fake.text(max_nb_chars=100)
    estado = random.choice(estados)

    reclamos.append((i + 1, dni_cliente, fecha_reclamo, cod_local,
        descripcion, estado))

# Insertar las tuplas en la base de datos
insert_query_venta = """
INSERT INTO venta (IdVenta, DNI_Cliente, FechaVenta, CodLocal,
    ImporteTotal, IdEmpleado)
```

```
VALUES (%s, %s, %s, %s, %s, %s)
"""

insert_query_reclamo = """
INSERT INTO reclamo (IdReclamo, DNI_Cliente, FechaReclamo, CodLocal,
    Descripcion, Estado)
VALUES (%s, %s, %s, %s, %s, %s)
"""

# Insertar los datos en bloques de 10000 tuplas
batch_size = 10000
for i in range(0, len(reclamos), batch_size):
    batch_v = ventas[i:i + batch_size]
    batch_r = reclamos[i:i + batch_size]
    try:
        cursor.executemany(insert_query_venta, batch_v)
        cursor.executemany(insert_query_reclamo, batch_r)
        conn.commit()
    except Exception as e:
        print(f"Error al insertar los datos: {e}")
        conn.rollback()

print("Datos insertados correctamente en la tabla venta y reclamo")

# Cerrar la conexion
cursor.close()
conn.close()
```

1.2 Fragmentación de las tablas

Las particionesmejoran significativamente el rendimiento. Al dividir grandes tablas en partes más pequeñas, se optimiza el acceso, reduciendo el tiempo de respuesta de las consultas y facilitando la escalabilidad de la base de datos.

1.2.1 Fragmentación en la tabla Venta

Se utilizó la técnica de partición por rango en el atributo `FechaVenta` para crear tres particiones. El vector de partición utilizado es: [2009-01-01, 2017-01-01]

```
-- Crear la tabla venta particionada
CREATE TABLE venta (
    IdVenta SERIAL,
    DNI_Cliente INT,
    FechaVenta DATE,
    CodLocal VARCHAR(5),
    ImporteTotal FLOAT,
    IdEmpleado VARCHAR(6)
) PARTITION BY RANGE (FechaVenta);

-- Crear particiones para la tabla venta
CREATE TABLE venta_2000_2008 PARTITION OF venta
    FOR VALUES FROM ('2000-01-01') TO ('2009-01-01');

CREATE TABLE venta_2009_2016 PARTITION OF venta
    FOR VALUES FROM ('2009-01-01') TO ('2017-01-01');

CREATE TABLE venta_2017_2024 PARTITION OF venta
```

```
    FOR VALUES FROM ('2017-01-01') TO ('2025-01-01');
```

1.2.2 Fragmentación en la tabla Reclamo

Se utilizó la técnica de partición por lista en el atributo `CodLocal` para crear tres particiones. Las particiones se realizaron de la siguiente forma:

- Partición 1: ['LOC01', 'LOC02', 'LOC03', 'LOC04']
- Partición 2: ['LOC05', 'LOC06', 'LOC07', 'LOC08']
- Partición 3: ['LOC09', 'LOC10', 'LOC11', 'LOC12']

```
CREATE TABLE reclamo (
    IdReclamo SERIAL,
    DNI_Cliente INT,
    FechaReclamo DATE,
    CodLocal VARCHAR(5),
    Descripcion VARCHAR(100),
    Estado VARCHAR(10)
) PARTITION BY LIST (CodLocal);

-- Crear particiones para la tabla reclamo
CREATE TABLE reclamo_loc1 PARTITION OF reclamo
    FOR VALUES IN ('LOC01', 'LOC02', 'LOC03', 'LOC04');

CREATE TABLE reclamo_loc2 PARTITION OF reclamo
    FOR VALUES IN ('LOC05', 'LOC06', 'LOC07', 'LOC08');

CREATE TABLE reclamo_loc3 PARTITION OF reclamo
    FOR VALUES IN ('LOC09', 'LOC10', 'LOC11', 'LOC12');
```

2 P2. Algoritmos distribuidos localmente

En este apartado se asumió que cada fragmento se encuentra en sitios diferentes, pero en realidad la consulta se ejecuta en el mismo servidor.

2.1 Consulta 1

El atributo ImporteTotal se generó en el rango de 1 a 1000, por lo que se tomó el vector de partición [333,666] en el atributo `ImporteTotal`

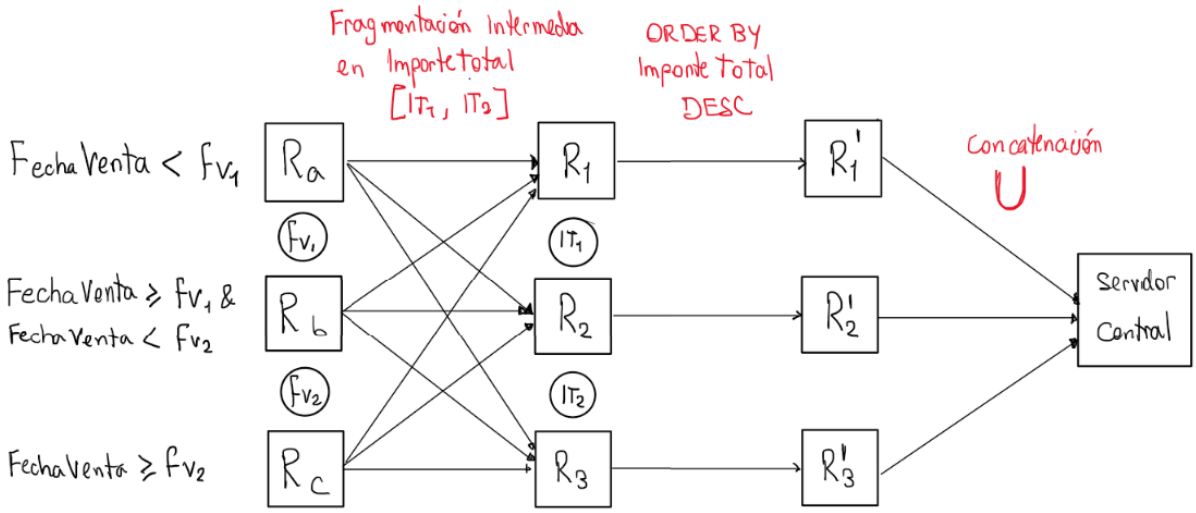

Figure 1: Optimización de la consulta 1 distribuida

2.1.1 Sentencia SQL de la implementación del algoritmo distribuido optimizado

```sql
SELECT *
FROM (
    SELECT *
    FROM (
        SELECT V.*
        FROM venta_2000_2008 V
        WHERE V.ImporteTotal >= 666
        UNION ALL
        SELECT V.*
        FROM venta_2009_2016 V
        WHERE V.ImporteTotal >= 666
        UNION ALL
        SELECT V.*
        FROM venta_2017_2024 V
        WHERE V.ImporteTotal >= 666
    ) AS temp_venta3
    ORDER BY ImporteTotal DESC
) AS temp_venta6

UNION ALL

SELECT *
FROM (
    SELECT *
    FROM (
        SELECT V.*
        FROM venta_2000_2008 V
        WHERE V.ImporteTotal >= 333 AND V.ImporteTotal < 666
        UNION ALL
        SELECT V.*
        FROM venta_2009_2016 V
        WHERE V.ImporteTotal >= 333 AND V.ImporteTotal < 666
        UNION ALL
        SELECT V.*
```

```
        FROM venta_2017_2024 V
        WHERE V.ImporteTotal >= 333 AND V.ImporteTotal < 666
    ) AS temp_venta2
    ORDER BY ImporteTotal DESC
) AS temp_venta5

UNION ALL

SELECT *
FROM (
    SELECT *
    FROM (
        SELECT V.*
        FROM venta_2000_2008 V
        WHERE V.ImporteTotal < 333
        UNION ALL
        SELECT V.*
        FROM venta_2009_2016 V
        WHERE V.ImporteTotal < 333
        UNION ALL
        SELECT V.*
        FROM venta_2017_2024 V
        WHERE V.ImporteTotal < 333
    ) AS temp_venta1
    ORDER BY ImporteTotal DESC
) AS temp_venta4;
```

2.1.2 Gráfico del plan de ejecución resultante

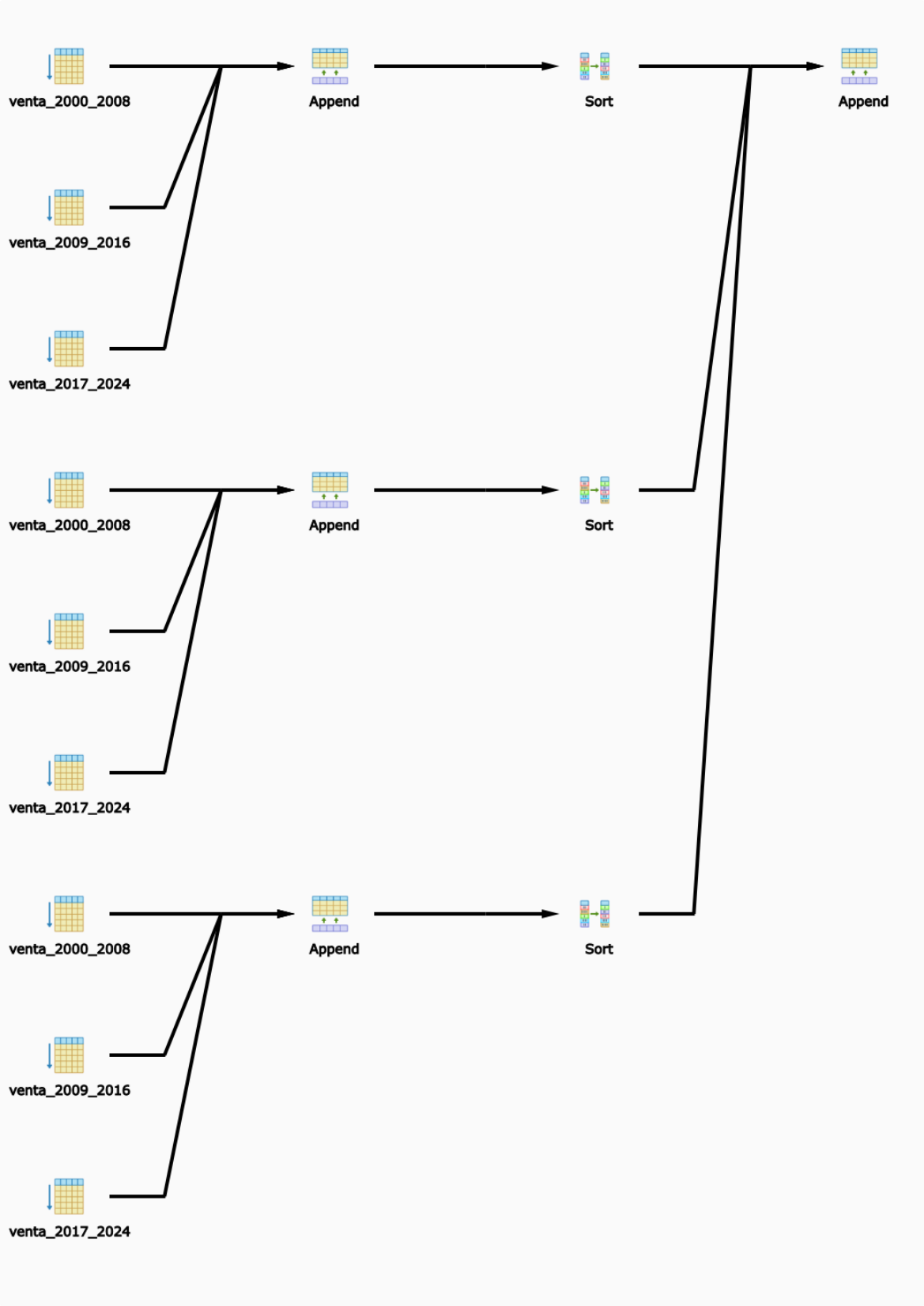

Figure 2: Grafico plan de ejecución en la consulta 1

2.2 Consulta 2

El atributo `DNI_Cliente` se generó en el rango de 70000000 a 77500000, por lo que se tomó el vector de partición [72500000, 75000000] en ese atributo.

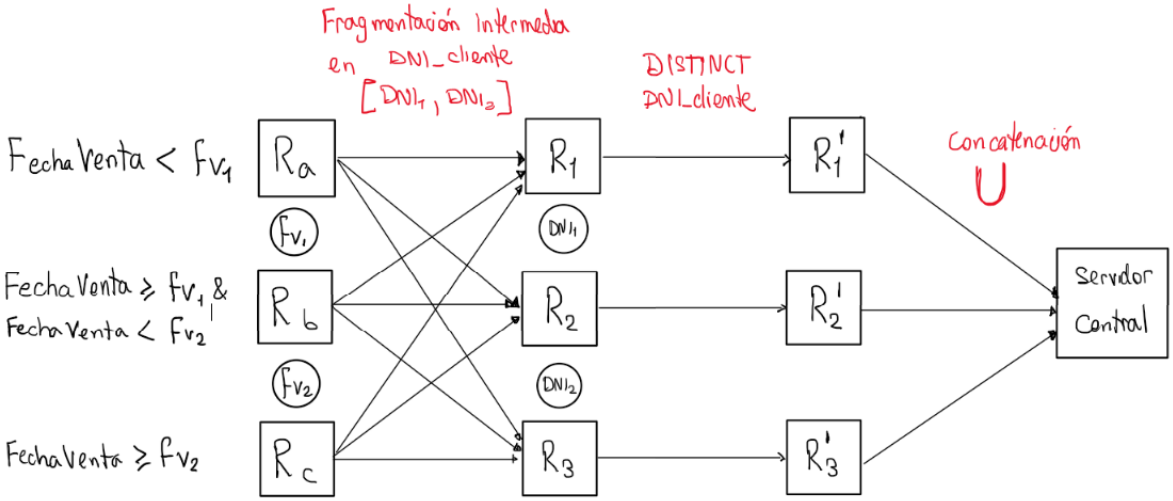

Figure 3: Optimización de la consulta 2 distribuida

2.2.1 Sentencia SQL de la implementación del algoritmo distribuido optimizado

```sql
SELECT *
FROM (
    SELECT DISTINCT DNI_Cliente
    FROM (
        SELECT V.DNI_Cliente
        FROM venta_2000_2008 V
        WHERE V.DNI_Cliente >= 75000000
        UNION ALL
        SELECT V.DNI_Cliente
        FROM venta_2009_2016 V
        WHERE V.DNI_Cliente >= 75000000
        UNION ALL
        SELECT V.DNI_Cliente
        FROM venta_2017_2024 V
        WHERE V.DNI_Cliente >= 75000000
    ) AS temp_venta3

    UNION ALL

    SELECT DISTINCT DNI_Cliente
    FROM (
        SELECT V.DNI_Cliente
        FROM venta_2000_2008 V
        WHERE V.DNI_Cliente >= 72500000 AND V.DNI_Cliente < 75000000
        UNION ALL
        SELECT V.DNI_Cliente
        FROM venta_2009_2016 V
        WHERE V.DNI_Cliente >= 72500000 AND V.DNI_Cliente < 75000000
        UNION ALL
```

```
        SELECT V.DNI_Cliente
        FROM venta_2017_2024 V
        WHERE V.DNI_Cliente >= 72500000 AND V.DNI_Cliente < 75000000
    ) AS temp_venta2

    UNION ALL

    SELECT DISTINCT DNI_Cliente
    FROM (
        SELECT V.DNI_Cliente
        FROM venta_2000_2008 V
        WHERE V.DNI_Cliente < 72500000
        UNION ALL
        SELECT V.DNI_Cliente
        FROM venta_2009_2016 V
        WHERE V.DNI_Cliente < 72500000
        UNION ALL
        SELECT V.DNI_Cliente
        FROM venta_2017_2024 V
        WHERE V.DNI_Cliente < 72500000
    ) AS temp_venta1
) AS final_result;
```

2.2.2 Gráfico del plan de ejecución resultante

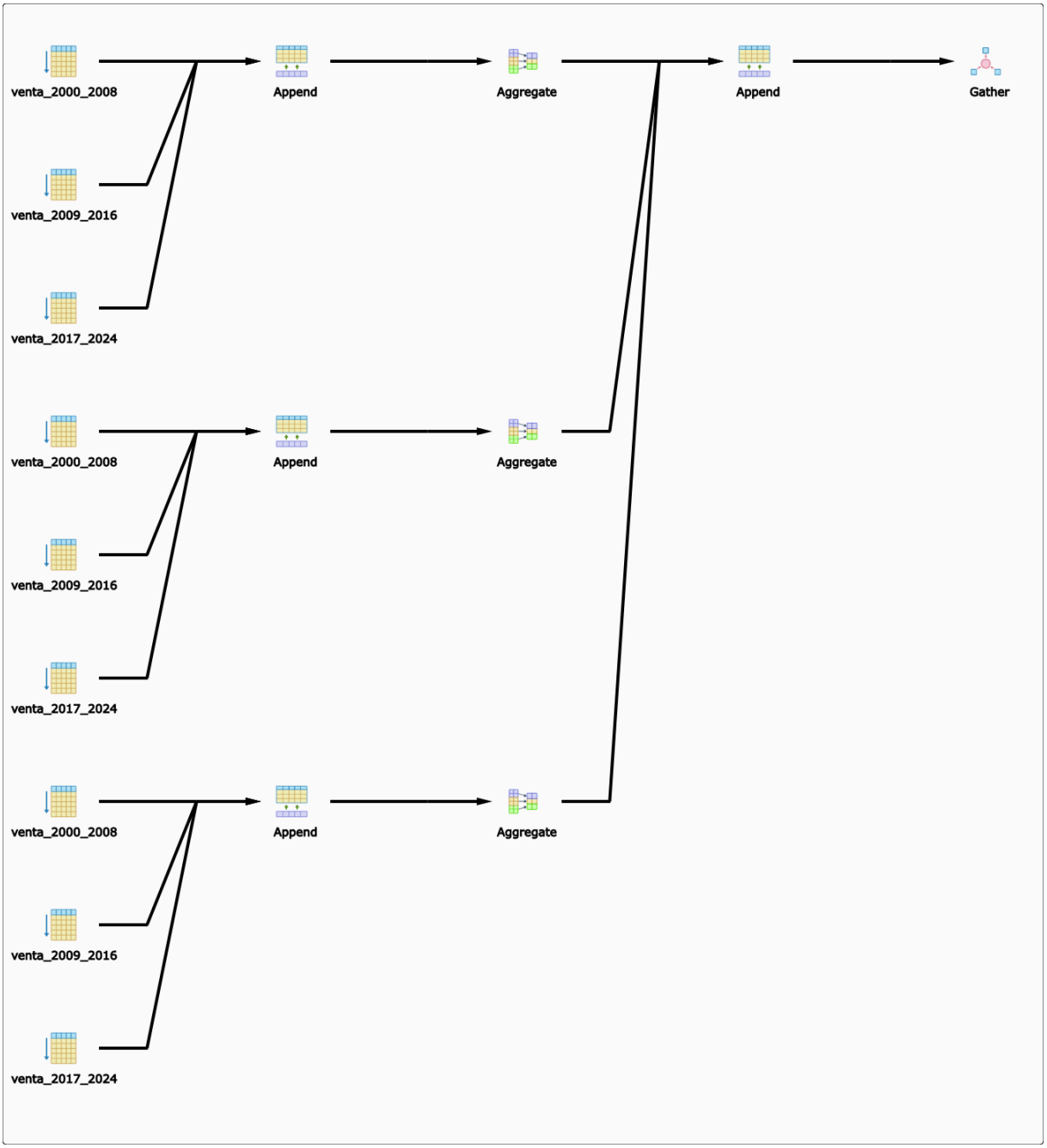

Figure 4: Grafico plan de ejecución en la consulta 2

2.3 Consulta 3

El atributo `CodLocal` se generó con los valores LOC01 - LOC12, por lo que se dividió en 3 grupos de 4 valores cada uno: [LOC01 a LOC04], [LOC05 a LOC08], [LOC09 a LOC12]

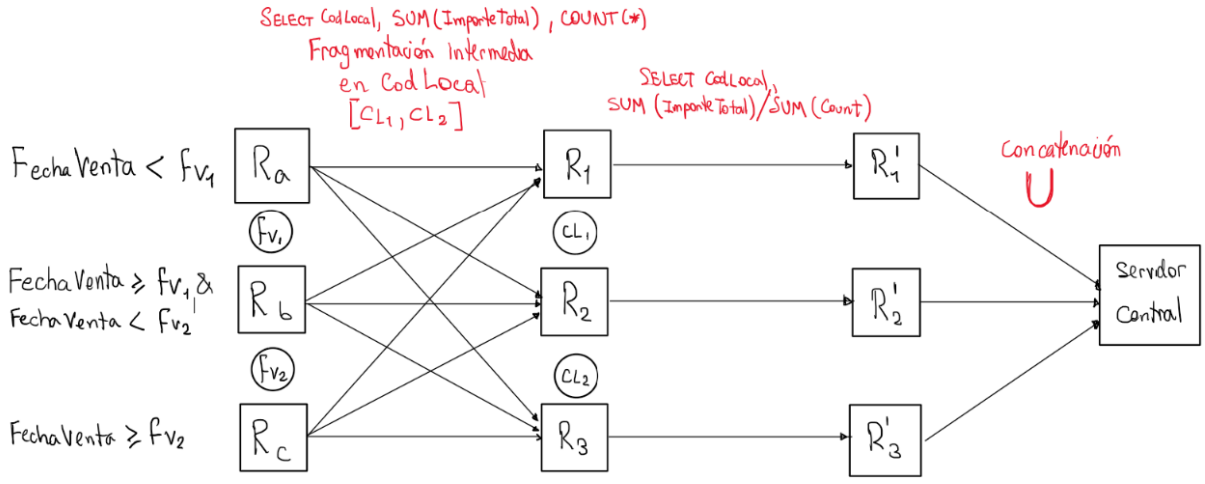

Figure 5: Optimización de la consulta 3 distribuida

2.3.1 Sentencia SQL de la implementación del algoritmo distribuido optimizado

```sql
SELECT *
FROM (
    SELECT CodLocal, SUM(SumaImporte) / SUM(CantVentas) AS
       PromedioImporte
    FROM (
        SELECT CodLocal, SUM(ImporteTotal) AS SumaImporte, COUNT(*) AS
            CantVentas
        FROM (
            SELECT V.CodLocal, V.ImporteTotal
            FROM venta_2000_2008 V
            WHERE V.CodLocal IN ('LOC01', 'LOC02', 'LOC03', 'LOC04')
            UNION ALL
            SELECT V.CodLocal, V.ImporteTotal
            FROM venta_2009_2016 V
            WHERE V.CodLocal IN ('LOC01', 'LOC02', 'LOC03', 'LOC04')
            UNION ALL
            SELECT V.CodLocal, V.ImporteTotal
            FROM venta_2017_2024 V
            WHERE V.CodLocal IN ('LOC01', 'LOC02', 'LOC03', 'LOC04')
        ) AS temp_venta1
        GROUP BY CodLocal
    ) AS aggregated_temp_venta1
    GROUP BY CodLocal

    UNION ALL

    SELECT CodLocal, SUM(SumaImporte) / SUM(CantVentas) AS
       PromedioImporte
    FROM (
        SELECT CodLocal, SUM(ImporteTotal) AS SumaImporte, COUNT(*) AS
            CantVentas
        FROM (
            SELECT V.CodLocal, V.ImporteTotal
            FROM venta_2000_2008 V
            WHERE V.CodLocal IN ('LOC05', 'LOC06', 'LOC07', 'LOC08')
```

```
            UNION ALL
            SELECT V.CodLocal, V.ImporteTotal
            FROM venta_2009_2016 V
            WHERE V.CodLocal IN ('LOC05', 'LOC06', 'LOC07', 'LOC08')
            UNION ALL
            SELECT V.CodLocal, V.ImporteTotal
            FROM venta_2017_2024 V
            WHERE V.CodLocal IN ('LOC05', 'LOC06', 'LOC07', 'LOC08')
        ) AS temp_venta2
        GROUP BY CodLocal
    ) AS aggregated_temp_venta2
    GROUP BY CodLocal

    UNION ALL

    SELECT CodLocal, SUM(SumaImporte) / SUM(CantVentas) AS
       PromedioImporte
    FROM (
        SELECT CodLocal, SUM(ImporteTotal) AS SumaImporte, COUNT(*) AS
            CantVentas
        FROM (
            SELECT V.CodLocal, V.ImporteTotal
            FROM venta_2000_2008 V
            WHERE V.CodLocal IN ('LOC09', 'LOC10', 'LOC11', 'LOC12')
            UNION ALL
            SELECT V.CodLocal, V.ImporteTotal
            FROM venta_2009_2016 V
            WHERE V.CodLocal IN ('LOC09', 'LOC10', 'LOC11', 'LOC12')
            UNION ALL
            SELECT V.CodLocal, V.ImporteTotal
            FROM venta_2017_2024 V
            WHERE V.CodLocal IN ('LOC09', 'LOC10', 'LOC11', 'LOC12')
        ) AS temp_venta3
        GROUP BY CodLocal
    ) AS aggregated_temp_venta3
    GROUP BY CodLocal
) AS final_result;
```

2.3.2 Gráfico del plan de ejecución resultante

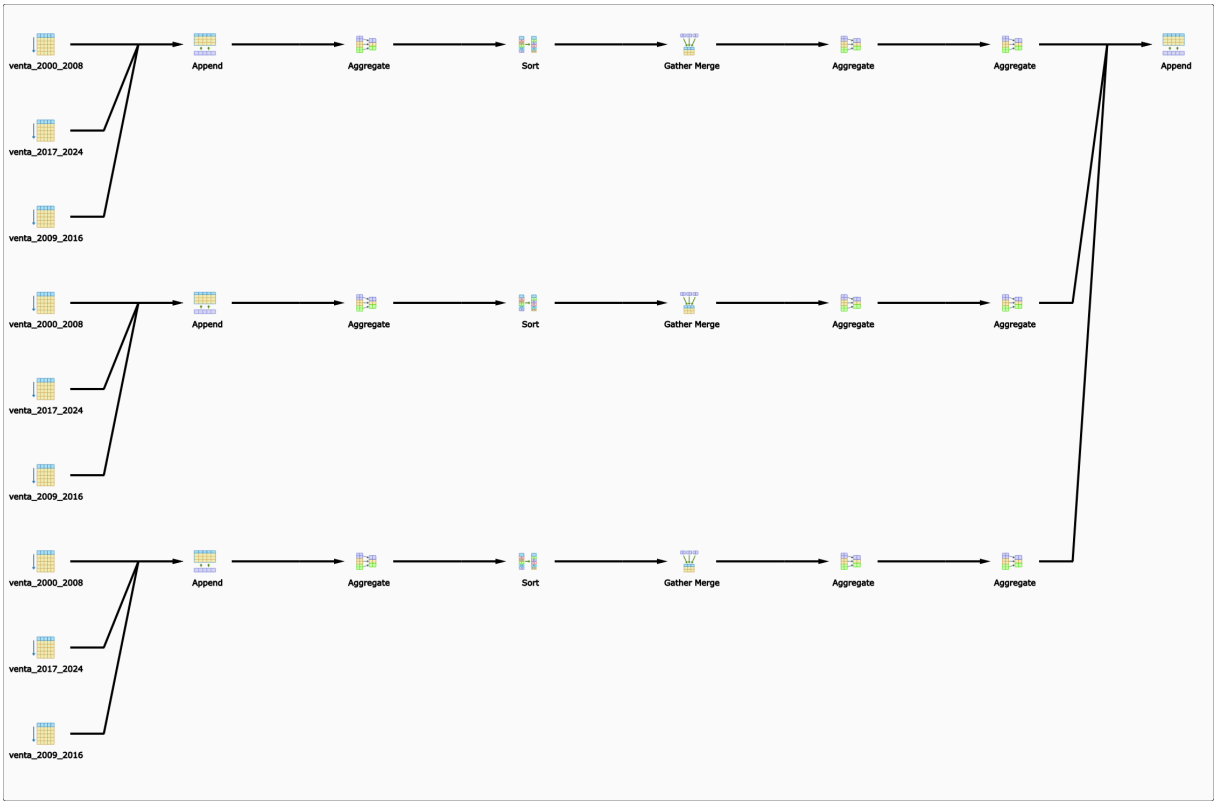

Figure 6: Grafico plan de ejecución en la consulta 3

2.4 Consulta 4

El atributo `DNI_Cliente` se generó en el rango de 70000000 a 77500000, por lo que se tomó el vector de partición [72500000, 75000000] en ese atributo.

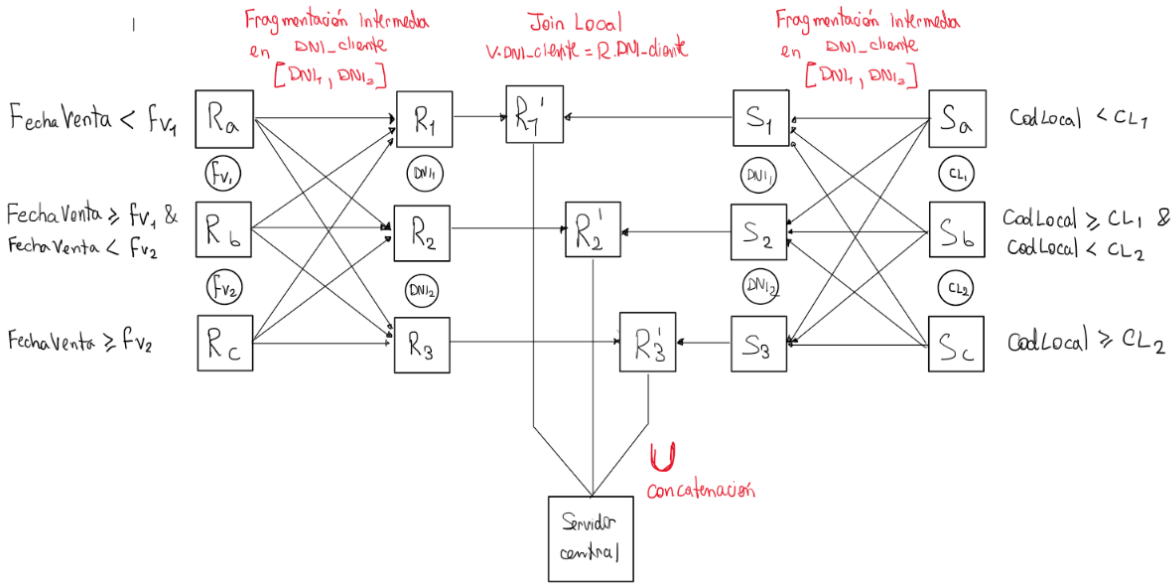

Figure 7: Optimización de la consulta 4 distribuida

2.4.1 Sentencia SQL de la implementación del algoritmo distribuido optimizado

```sql
SELECT *
FROM (
    SELECT *
    FROM (
        SELECT R.*
        FROM reclamo_loc1 R
        WHERE R.DNI_Cliente < 72500000
        UNION ALL
        SELECT R.*
        FROM reclamo_loc2 R
        WHERE R.DNI_Cliente < 72500000
        UNION ALL
        SELECT R.*
        FROM reclamo_loc3 R
        WHERE R.DNI_Cliente < 72500000
    ) AS temp_reclamo1
    JOIN (
        SELECT DNI_Cliente
        FROM venta_2000_2008 V
        WHERE V.DNI_Cliente < 72500000
        UNION ALL
        SELECT DNI_Cliente
        FROM venta_2009_2016 V
        WHERE V.DNI_Cliente < 72500000
        UNION ALL
        SELECT DNI_Cliente
        FROM venta_2017_2024 V
        WHERE V.DNI_Cliente < 72500000
    ) AS temp_venta1
    ON temp_reclamo1.DNI_Cliente = temp_venta1.DNI_Cliente

    UNION ALL

    SELECT *
    FROM (
        SELECT R.*
        FROM reclamo_loc1 R
        WHERE R.DNI_Cliente >= 72500000 AND R.DNI_Cliente < 75000000
        UNION ALL
        SELECT R.*
        FROM reclamo_loc2 R
        WHERE R.DNI_Cliente >= 72500000 AND R.DNI_Cliente < 75000000
        UNION ALL
        SELECT R.*
        FROM reclamo_loc3 R
        WHERE R.DNI_Cliente >= 72500000 AND R.DNI_Cliente < 75000000
    ) AS temp_reclamo2
    JOIN (
        SELECT DNI_Cliente
        FROM venta_2000_2008 V
        WHERE V.DNI_Cliente >= 72500000 AND V.DNI_Cliente < 75000000
        UNION ALL
        SELECT DNI_Cliente
        FROM venta_2009_2016 V
```

```
        WHERE V.DNI_Cliente >= 72500000 AND V.DNI_Cliente < 75000000
        UNION ALL
        SELECT DNI_Cliente
        FROM venta_2017_2024 V
        WHERE V.DNI_Cliente >= 72500000 AND V.DNI_Cliente < 75000000
    ) AS temp_venta2
    ON temp_reclamo2.DNI_Cliente = temp_venta2.DNI_Cliente

    UNION ALL

    SELECT *
    FROM (
        SELECT R.*
        FROM reclamo_loc1 R
        WHERE R.DNI_Cliente >= 75000000
        UNION ALL
        SELECT R.*
        FROM reclamo_loc2 R
        WHERE R.DNI_Cliente >= 75000000
        UNION ALL
        SELECT R.*
        FROM reclamo_loc3 R
        WHERE R.DNI_Cliente >= 75000000
    ) AS temp_reclamo3
    JOIN (
        SELECT DNI_Cliente
        FROM venta_2000_2008 V
        WHERE V.DNI_Cliente >= 75000000
        UNION ALL
        SELECT DNI_Cliente
        FROM venta_2009_2016 V
        WHERE V.DNI_Cliente >= 75000000
        UNION ALL
        SELECT DNI_Cliente
        FROM venta_2017_2024 V
        WHERE V.DNI_Cliente >= 75000000
    ) AS temp_venta3
    ON temp_reclamo3.DNI_Cliente = temp_venta3.DNI_Cliente
) AS final_result;
```

2.4.2 Gráfico del plan de ejecución resultante

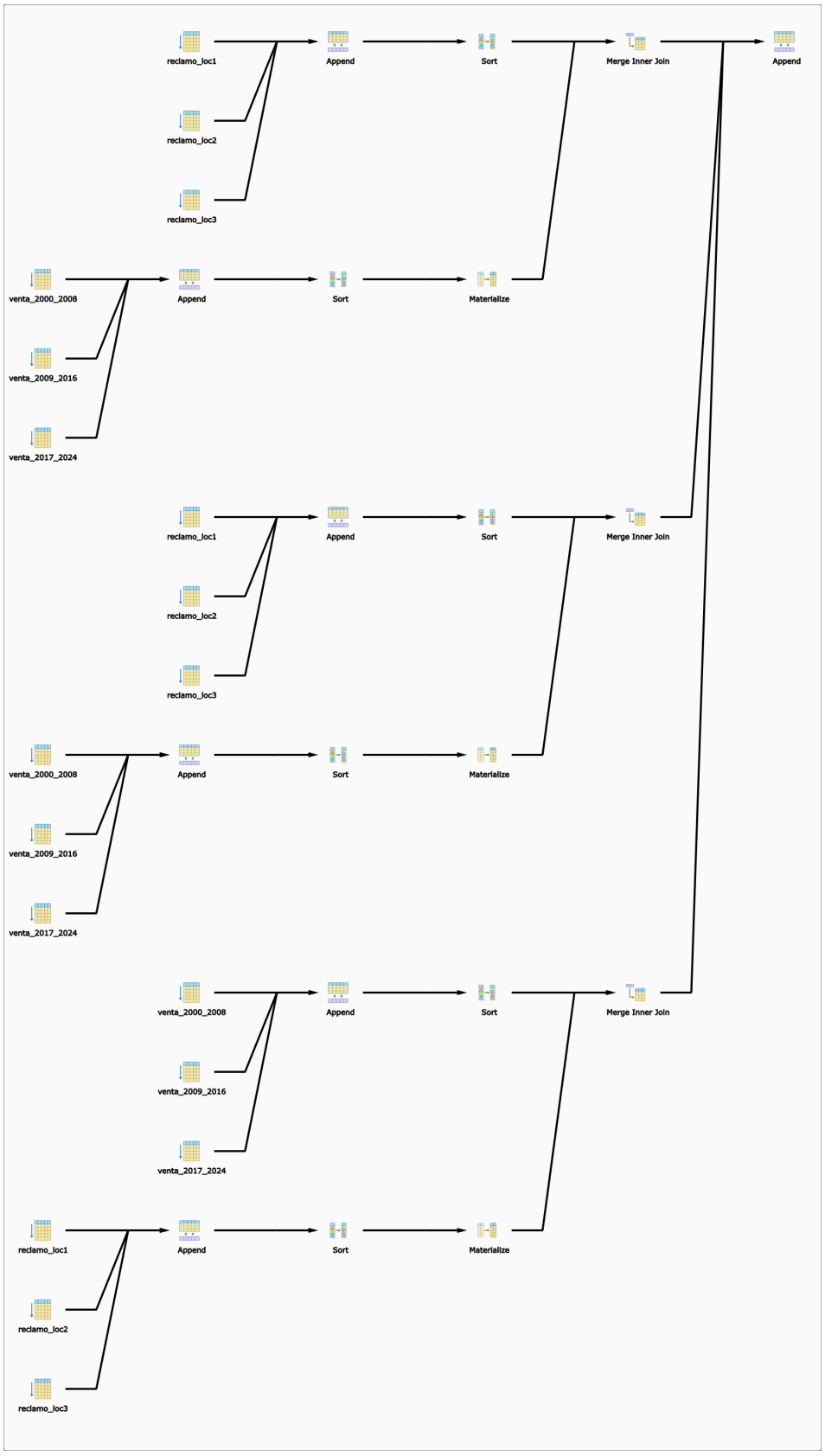

Figure 8: Grafico plan de ejecución en la consulta 4

3 P3. Algoritmos distribuidos en al menos tres servidores

Para este caso, se estan usando 3 contenedores de Docker para simular 3 servidores remotos y ejecutar todas las consultas ahí.

```yaml
version: '3.8'

services:
  postgres1:
    image: postgres:latest
    container_name: postgres1
    environment:
      POSTGRES_USER: user1
      POSTGRES_PASSWORD: password1
      POSTGRES_DB: db1
    ports:
      - "5433:5432"
    networks:
      - postgres-network

  postgres2:
    image: postgres:latest
    container_name: postgres2
    environment:
      POSTGRES_USER: user2
      POSTGRES_PASSWORD: password2
      POSTGRES_DB: db2
    ports:
      - "5434:5432"
    networks:
      - postgres-network

  postgres3:
    image: postgres:latest
    container_name: postgres3
    environment:
      POSTGRES_USER: user3
      POSTGRES_PASSWORD: password3
      POSTGRES_DB: db3
    ports:
      - "5435:5432"
    networks:
      - postgres-network

networks:
  postgres-network:
    driver: bridge
```

3.1 Consulta 1

```sql
SELECT *
FROM (
    SELECT *
    FROM (
        SELECT V.*
        FROM dblink('dbname=db1 user=user1 password=password1 host=
            localhost port=5433',
                    'SELECT * FROM venta_2000_2008 WHERE ImporteTotal
                        >= 666')
```

```
        AS V(IdVenta INT, DNI_Cliente INT, FechaVenta DATE, CodLocal
            VARCHAR, ImporteTotal FLOAT, IdEmpleado VARCHAR)
        UNION ALL
        SELECT V.*
        FROM dblink('dbname=db2 user=user2 password=password2 host=
            localhost port=5434',
                    'SELECT * FROM venta_2009_2016 WHERE ImporteTotal
                        >= 666')
        AS V(IdVenta INT, DNI_Cliente INT, FechaVenta DATE, CodLocal
            VARCHAR, ImporteTotal FLOAT, IdEmpleado VARCHAR)
        UNION ALL
        SELECT V.*
        FROM dblink('dbname=db3 user=user3 password=password3 host=
            localhost port=5435',
                    'SELECT * FROM venta_2017_2024 WHERE ImporteTotal
                        >= 666')
        AS V(IdVenta INT, DNI_Cliente INT, FechaVenta DATE, CodLocal
            VARCHAR, ImporteTotal FLOAT, IdEmpleado VARCHAR)
    ) AS temp_venta3
    ORDER BY ImporteTotal DESC
) AS temp_venta6

UNION ALL

SELECT *
FROM (
    SELECT *
    FROM (
        SELECT V.*
        FROM dblink('dbname=db1 user=user1 password=password1 host=
            localhost port=5433',
                    'SELECT * FROM venta_2000_2008 WHERE ImporteTotal
                        >= 333 AND ImporteTotal < 666')
        AS V(IdVenta INT, DNI_Cliente INT, FechaVenta DATE, CodLocal
            VARCHAR, ImporteTotal FLOAT, IdEmpleado VARCHAR)
        UNION ALL
        SELECT V.*
        FROM dblink('dbname=db2 user=user2 password=password2 host=
            localhost port=5434',
                    'SELECT * FROM venta_2009_2016 WHERE ImporteTotal
                        >= 333 AND ImporteTotal < 666')
        AS V(IdVenta INT, DNI_Cliente INT, FechaVenta DATE, CodLocal
            VARCHAR, ImporteTotal FLOAT, IdEmpleado VARCHAR)
        UNION ALL
        SELECT V.*
        FROM dblink('dbname=db3 user=user3 password=password3 host=
            localhost port=5435',
                    'SELECT * FROM venta_2017_2024 WHERE ImporteTotal
                        >= 333 AND ImporteTotal < 666')
        AS V(IdVenta INT, DNI_Cliente INT, FechaVenta DATE, CodLocal
            VARCHAR, ImporteTotal FLOAT, IdEmpleado VARCHAR)
    ) AS temp_venta2
    ORDER BY ImporteTotal DESC
) AS temp_venta5

UNION ALL
```

```
48
49 SELECT *
50 FROM (
51     SELECT *
52     FROM (
53         SELECT V.*
54         FROM dblink('dbname=db1 user=user1 password=password1 host=
               localhost port=5433',
55                     'SELECT * FROM venta_2000_2008 WHERE ImporteTotal
                          < 333')
56         AS V(IdVenta INT, DNI_Cliente INT, FechaVenta DATE, CodLocal
               VARCHAR, ImporteTotal FLOAT, IdEmpleado VARCHAR)
57         UNION ALL
58         SELECT V.*
59         FROM dblink('dbname=db2 user=user2 password=password2 host=
               localhost port=5434',
60                     'SELECT * FROM venta_2009_2016 WHERE ImporteTotal
                          < 333')
61         AS V(IdVenta INT, DNI_Cliente INT, FechaVenta DATE, CodLocal
               VARCHAR, ImporteTotal FLOAT, IdEmpleado VARCHAR)
62         UNION ALL
63         SELECT V.*
64         FROM dblink('dbname=db3 user=user3 password=password3 host=
               localhost port=5435',
65                     'SELECT * FROM venta_2017_2024 WHERE ImporteTotal
                          < 333')
66         AS V(IdVenta INT, DNI_Cliente INT, FechaVenta DATE, CodLocal
               VARCHAR, ImporteTotal FLOAT, IdEmpleado VARCHAR)
67     ) AS temp_venta1
68     ORDER BY ImporteTotal DESC
69 ) AS temp_venta4
70 ORDER BY ImporteTotal DESC;
```

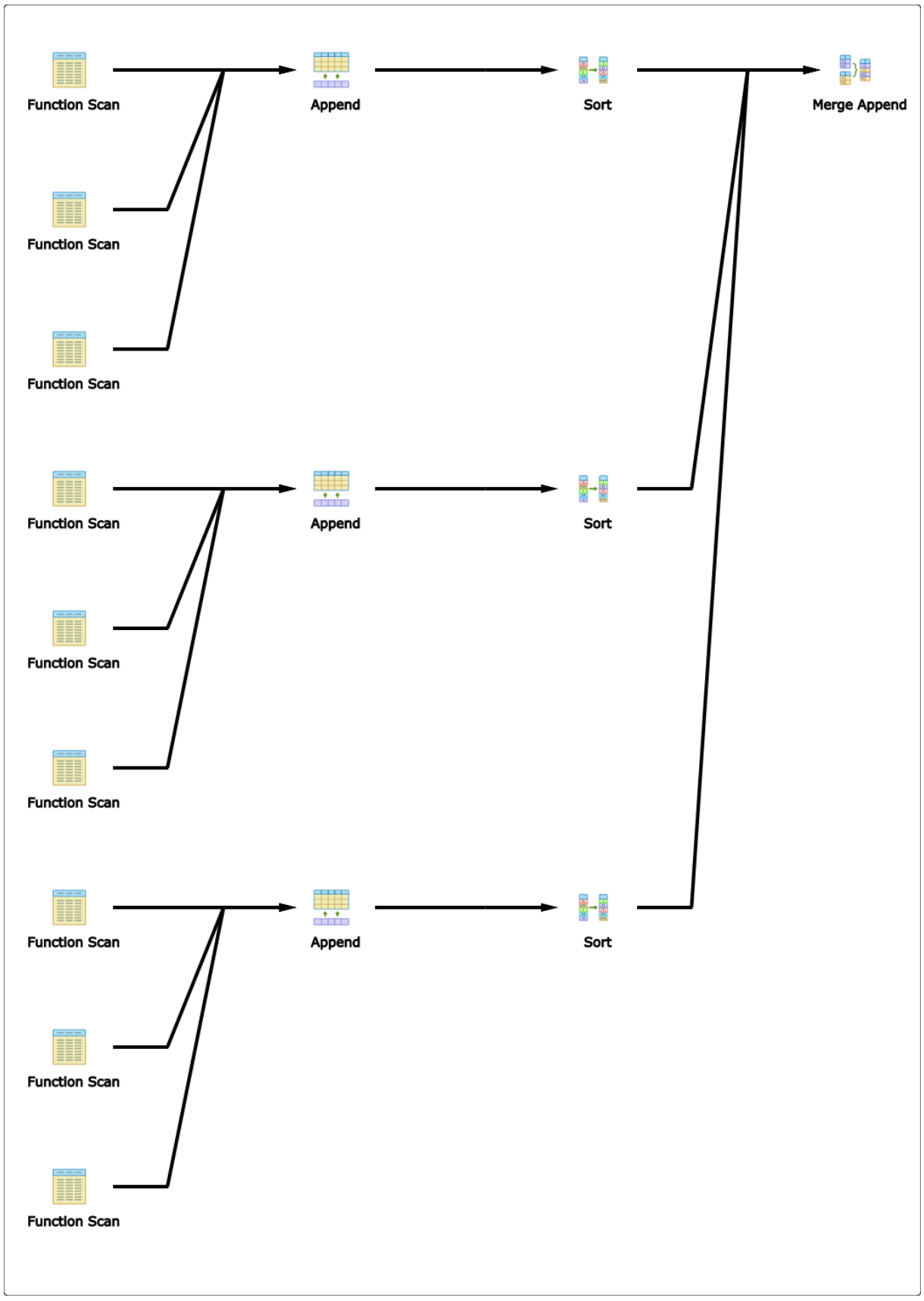

Figure 9: Optimización de la consulta 1 distribuida en diferentes servidores

3.2 Consulta 2

```
SELECT DISTINCT DNI_Cliente
FROM (
    SELECT *
    FROM (
        SELECT V.DNI_Cliente
        FROM dblink('dbname=db1 user=user1 password=password1 host=
            localhost port=5433',
                    'SELECT DNI_Cliente FROM venta_2000_2008 WHERE
                       DNI_Cliente >= 75000000')
        AS V(DNI_Cliente INT)
        UNION ALL
        SELECT V.DNI_Cliente
        FROM dblink('dbname=db2 user=user2 password=password2 host=
            localhost port=5434',
                    'SELECT DNI_Cliente FROM venta_2009_2016 WHERE
                       DNI_Cliente >= 75000000')
        AS V(DNI_Cliente INT)
        UNION ALL
        SELECT V.DNI_Cliente
        FROM dblink('dbname=db3 user=user3 password=password3 host=
            localhost port=5435',
                    'SELECT DNI_Cliente FROM venta_2017_2024 WHERE
                       DNI_Cliente >= 75000000')
        AS V(DNI_Cliente INT)
    ) AS temp_venta3
) AS temp_venta6

UNION ALL

SELECT DISTINCT DNI_Cliente
FROM (
    SELECT *
    FROM (
        SELECT V.DNI_Cliente
        FROM dblink('dbname=db1 user=user1 password=password1 host=
            localhost port=5433',
                    'SELECT DNI_Cliente FROM venta_2000_2008 WHERE
                       DNI_Cliente >= 72500000 AND DNI_Cliente <
                       75000000')
        AS V(DNI_Cliente INT)
        UNION ALL
        SELECT V.DNI_Cliente
        FROM dblink('dbname=db2 user=user2 password=password2 host=
            localhost port=5434',
                    'SELECT DNI_Cliente FROM venta_2009_2016 WHERE
                       DNI_Cliente >= 72500000 AND DNI_Cliente <
                       75000000')
        AS V(DNI_Cliente INT)
        UNION ALL
        SELECT V.DNI_Cliente
        FROM dblink('dbname=db3 user=user3 password=password3 host=
            localhost port=5435',
                    'SELECT DNI_Cliente FROM venta_2017_2024 WHERE
                       DNI_Cliente >= 72500000 AND DNI_Cliente <
```

```
                        75000000')
        AS V(DNI_Cliente INT)
    ) AS temp_venta2
) AS temp_venta5

UNION ALL

SELECT DISTINCT DNI_Cliente
FROM (
    SELECT *
    FROM (
        SELECT V.DNI_Cliente
        FROM dblink('dbname=db1 user=user1 password=password1 host=
            localhost port=5433',
                    'SELECT DNI_Cliente FROM venta_2000_2008 WHERE
                        DNI_Cliente < 72500000')
        AS V(DNI_Cliente INT)
        UNION ALL
        SELECT V.DNI_Cliente
        FROM dblink('dbname=db2 user=user2 password=password2 host=
            localhost port=5434',
                    'SELECT DNI_Cliente FROM venta_2009_2016 WHERE
                        DNI_Cliente < 72500000')
        AS V(DNI_Cliente INT)
        UNION ALL
        SELECT V.DNI_Cliente
        FROM dblink('dbname=db3 user=user3 password=password3 host=
            localhost port=5435',
                    'SELECT DNI_Cliente FROM venta_2017_2024 WHERE
                        DNI_Cliente < 72500000')
        AS V(DNI_Cliente INT)
    ) AS temp_venta1
) AS temp_venta4;
```

Figure 10: Optimización de la consulta 2 distribuida en diferentes servidores

3.3 Consulta 3

```
SELECT CodLocal, SUM(SumaImporte) / SUM(CantVentas) AS PromedioImporte
FROM (
    SELECT CodLocal, SUM(ImporteTotal) AS SumaImporte, COUNT(*) AS
       CantVentas
    FROM (
        SELECT V.CodLocal, V.ImporteTotal
        FROM dblink('dbname=db1 user=user1 password=password1 host=
           localhost port=5433',
                    'SELECT CodLocal, ImporteTotal FROM
                       venta_2000_2008 WHERE CodLocal IN (''LOC01'', '
                       'LOC02'', ''LOC03'', ''LOC04'')')
        AS V(CodLocal VARCHAR, ImporteTotal FLOAT)
        UNION ALL
        SELECT V.CodLocal, V.ImporteTotal
        FROM dblink('dbname=db2 user=user2 password=password2 host=
           localhost port=5434',
                    'SELECT CodLocal, ImporteTotal FROM
                       venta_2009_2016 WHERE CodLocal IN (''LOC01'', '
                       'LOC02'', ''LOC03'', ''LOC04'')')
        AS V(CodLocal VARCHAR, ImporteTotal FLOAT)
        UNION ALL
        SELECT V.CodLocal, V.ImporteTotal
        FROM dblink('dbname=db3 user=user3 password=password3 host=
           localhost port=5435',
                    'SELECT CodLocal, ImporteTotal FROM
                       venta_2017_2024 WHERE CodLocal IN (''LOC01'', '
                       'LOC02'', ''LOC03'', ''LOC04'')')
        AS V(CodLocal VARCHAR, ImporteTotal FLOAT)
    ) AS temp_venta1
    GROUP BY CodLocal
) AS aggregated_temp_venta1
GROUP BY CodLocal

UNION ALL

SELECT CodLocal, SUM(SumaImporte) / SUM(CantVentas) AS PromedioImporte
FROM (
    SELECT CodLocal, SUM(ImporteTotal) AS SumaImporte, COUNT(*) AS
       CantVentas
    FROM (
        SELECT V.CodLocal, V.ImporteTotal
        FROM dblink('dbname=db1 user=user1 password=password1 host=
           localhost port=5433',
                    'SELECT CodLocal, ImporteTotal FROM
                       venta_2000_2008 WHERE CodLocal IN (''LOC05'', '
                       'LOC06'', ''LOC07'', ''LOC08'')')
        AS V(CodLocal VARCHAR, ImporteTotal FLOAT)
        UNION ALL
        SELECT V.CodLocal, V.ImporteTotal
        FROM dblink('dbname=db2 user=user2 password=password2 host=
           localhost port=5434',
                    'SELECT CodLocal, ImporteTotal FROM
                       venta_2009_2016 WHERE CodLocal IN (''LOC05'', '
                       'LOC06'', ''LOC07'', ''LOC08'')')
```

```
        AS V(CodLocal VARCHAR, ImporteTotal FLOAT)
        UNION ALL
        SELECT V.CodLocal, V.ImporteTotal
        FROM dblink('dbname=db3 user=user3 password=password3 host=
           localhost port=5435',
                    'SELECT CodLocal, ImporteTotal FROM
                       venta_2017_2024 WHERE CodLocal IN (''LOC05'', '
                       'LOC06'', ''LOC07'', ''LOC08'')')
        AS V(CodLocal VARCHAR, ImporteTotal FLOAT)
    ) AS temp_venta2
    GROUP BY CodLocal
) AS aggregated_temp_venta2
GROUP BY CodLocal

UNION ALL

SELECT CodLocal, SUM(SumaImporte) / SUM(CantVentas) AS PromedioImporte
FROM (
    SELECT CodLocal, SUM(ImporteTotal) AS SumaImporte, COUNT(*) AS
       CantVentas
    FROM (
        SELECT V.CodLocal, V.ImporteTotal
        FROM dblink('dbname=db1 user=user1 password=password1 host=
           localhost port=5433',
                    'SELECT CodLocal, ImporteTotal FROM
                       venta_2000_2008 WHERE CodLocal IN (''LOC09'', '
                       'LOC10'', ''LOC11'', ''LOC12'')')
        AS V(CodLocal VARCHAR, ImporteTotal FLOAT)
        UNION ALL
        SELECT V.CodLocal, V.ImporteTotal
        FROM dblink('dbname=db2 user=user2 password=password2 host=
           localhost port=5434',
                    'SELECT CodLocal, ImporteTotal FROM
                       venta_2009_2016 WHERE CodLocal IN (''LOC09'', '
                       'LOC10'', ''LOC11'', ''LOC12'')')
        AS V(CodLocal VARCHAR, ImporteTotal FLOAT)
        UNION ALL
        SELECT V.CodLocal, V.ImporteTotal
        FROM dblink('dbname=db3 user=user3 password=password3 host=
           localhost port=5435',
                    'SELECT CodLocal, ImporteTotal FROM
                       venta_2017_2024 WHERE CodLocal IN (''LOC09'', '
                       'LOC10'', ''LOC11'', ''LOC12'')')
        AS V(CodLocal VARCHAR, ImporteTotal FLOAT)
    ) AS temp_venta3
    GROUP BY CodLocal
) AS aggregated_temp_venta3
GROUP BY CodLocal;
```

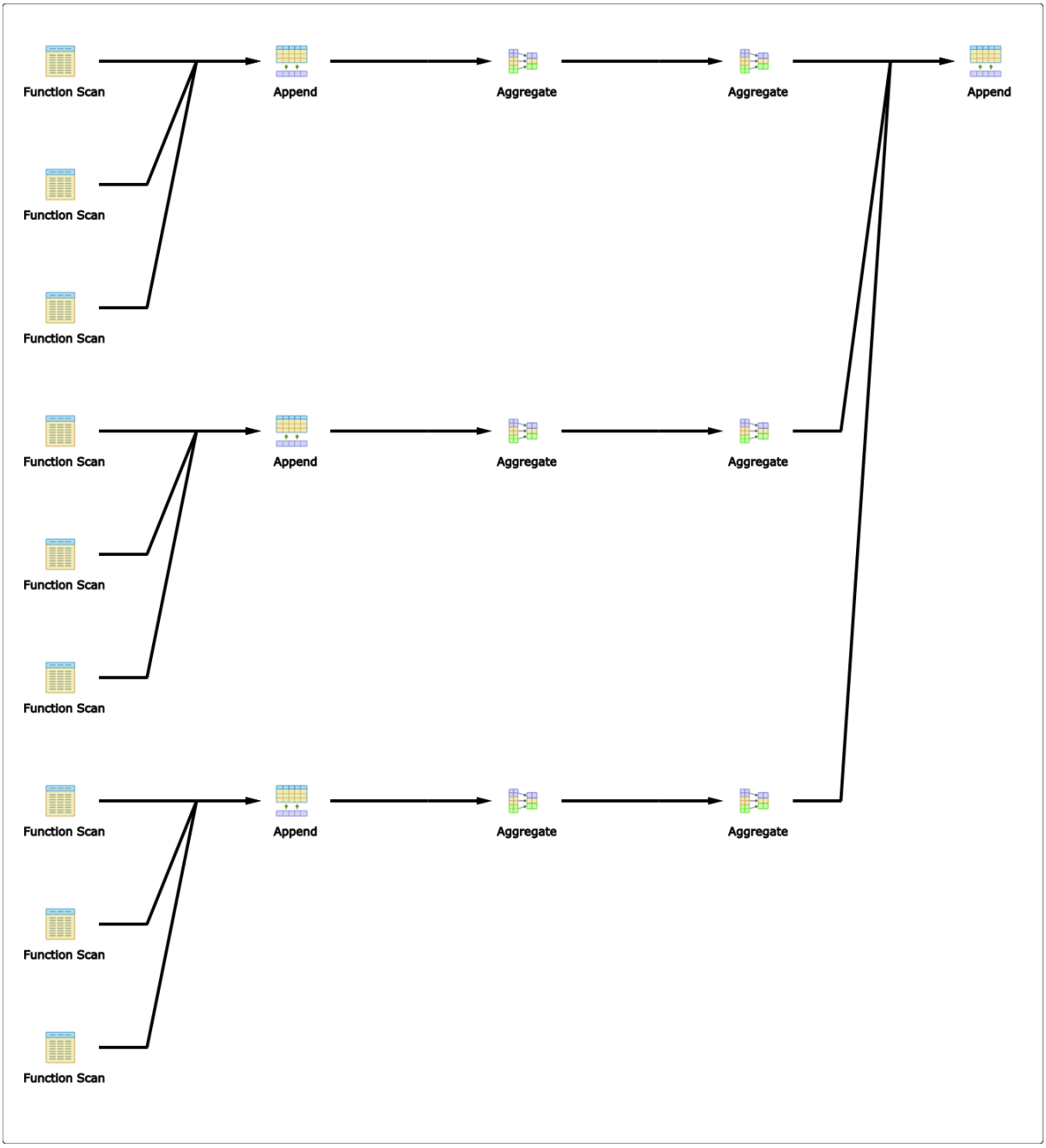

Figure 11: Optimización de la consulta 3 distribuida en diferentes servidores

3.4 Consulta 4

```
1 SELECT *
2 FROM (
3     SELECT *
4     FROM (
5         SELECT R.*
6         FROM dblink('dbname=db1 user=user1 password=password1 host=
              localhost port=5433',
7                     'SELECT * FROM reclamo_loc1_loc4 WHERE DNI_Cliente
                          < 72500000')
```

```
        AS R(IdReclamo INT, DNI_Cliente INT, FechaReclamo DATE,
            CodLocal VARCHAR, Descripcion VARCHAR, Estado VARCHAR)
        UNION ALL
        SELECT R.*
        FROM dblink('dbname=db2 user=user2 password=password2 host=
            localhost port=5434',
                    'SELECT * FROM reclamo_loc5_loc8 WHERE DNI_Cliente
                        < 72500000')
        AS R(IdReclamo INT, DNI_Cliente INT, FechaReclamo DATE,
            CodLocal VARCHAR, Descripcion VARCHAR, Estado VARCHAR)
        UNION ALL
        SELECT R.*
        FROM dblink('dbname=db3 user=user3 password=password3 host=
            localhost port=5435',
                    'SELECT * FROM reclamo_loc9_loc12 WHERE
                        DNI_Cliente < 72500000')
        AS R(IdReclamo INT, DNI_Cliente INT, FechaReclamo DATE,
            CodLocal VARCHAR, Descripcion VARCHAR, Estado VARCHAR)
    ) AS temp_reclamo1
    JOIN (
        SELECT DNI_Cliente
        FROM dblink('dbname=db1 user=user1 password=password1 host=
            localhost port=5433',
                    'SELECT DNI_Cliente FROM venta_2000_2008 WHERE
                        DNI_Cliente < 72500000')
        AS V(DNI_Cliente INT)
        UNION ALL
        SELECT DNI_Cliente
        FROM dblink('dbname=db2 user=user2 password=password2 host=
            localhost port=5434',
                    'SELECT DNI_Cliente FROM venta_2009_2016 WHERE
                        DNI_Cliente < 72500000')
        AS V(DNI_Cliente INT)
        UNION ALL
        SELECT DNI_Cliente
        FROM dblink('dbname=db3 user=user3 password=password3 host=
            localhost port=5435',
                    'SELECT DNI_Cliente FROM venta_2017_2024 WHERE
                        DNI_Cliente < 72500000')
        AS V(DNI_Cliente INT)
    ) AS temp_venta1
    ON temp_reclamo1.DNI_Cliente = temp_venta1.DNI_Cliente
) AS result1

UNION ALL

SELECT *
FROM (
    SELECT *
    FROM (
        SELECT R.*
        FROM dblink('dbname=db1 user=user1 password=password1 host=
            localhost port=5433',
                    'SELECT * FROM reclamo_loc1_loc4 WHERE DNI_Cliente
                        >= 72500000 AND DNI_Cliente < 75000000')
```

```
48          AS R(IdReclamo INT, DNI_Cliente INT, FechaReclamo DATE,
                CodLocal VARCHAR, Descripcion VARCHAR, Estado VARCHAR)
49          UNION ALL
50          SELECT R.*
51          FROM dblink('dbname=db2 user=user2 password=password2 host=
                localhost port=5434',
52                      'SELECT * FROM reclamo_loc5_loc8 WHERE DNI_Cliente
                            >= 72500000 AND DNI_Cliente < 75000000')
53          AS R(IdReclamo INT, DNI_Cliente INT, FechaReclamo DATE,
                CodLocal VARCHAR, Descripcion VARCHAR, Estado VARCHAR)
54          UNION ALL
55          SELECT R.*
56          FROM dblink('dbname=db3 user=user3 password=password3 host=
                localhost port=5435',
57                      'SELECT * FROM reclamo_loc9_loc12 WHERE
                            DNI_Cliente >= 72500000 AND DNI_Cliente <
                            75000000')
58          AS R(IdReclamo INT, DNI_Cliente INT, FechaReclamo DATE,
                CodLocal VARCHAR, Descripcion VARCHAR, Estado VARCHAR)
59      ) AS temp_reclamo2
60      JOIN (
61          SELECT DNI_Cliente
62          FROM dblink('dbname=db1 user=user1 password=password1 host=
                localhost port=5433',
63                      'SELECT DNI_Cliente FROM venta_2000_2008 WHERE
                            DNI_Cliente >= 72500000 AND DNI_Cliente <
                            75000000')
64          AS V(DNI_Cliente INT)
65          UNION ALL
66          SELECT DNI_Cliente
67          FROM dblink('dbname=db2 user=user2 password=password2 host=
                localhost port=5434',
68                      'SELECT DNI_Cliente FROM venta_2009_2016 WHERE
                            DNI_Cliente >= 72500000 AND DNI_Cliente <
                            75000000')
69          AS V(DNI_Cliente INT)
70          UNION ALL
71          SELECT DNI_Cliente
72          FROM dblink('dbname=db3 user=user3 password=password3 host=
                localhost port=5435',
73                      'SELECT DNI_Cliente FROM venta_2017_2024 WHERE
                            DNI_Cliente >= 72500000 AND DNI_Cliente <
                            75000000')
74          AS V(DNI_Cliente INT)
75      ) AS temp_venta2
76      ON temp_reclamo2.DNI_Cliente = temp_venta2.DNI_Cliente
77  ) AS result2
78
79  UNION ALL
80
81  SELECT *
82  FROM (
83      SELECT *
84      FROM (
85          SELECT R.*
```

```
86          FROM dblink('dbname=db1 user=user1 password=password1 host=
                localhost port=5433',
87                      'SELECT * FROM reclamo_loc1_loc4 WHERE DNI_Cliente
                            >= 75000000')
88          AS R(IdReclamo INT, DNI_Cliente INT, FechaReclamo DATE,
                CodLocal VARCHAR, Descripcion VARCHAR, Estado VARCHAR)
89          UNION ALL
90          SELECT R.*
91          FROM dblink('dbname=db2 user=user2 password=password2 host=
                localhost port=5434',
92                      'SELECT * FROM reclamo_loc5_loc8 WHERE DNI_Cliente
                            >= 75000000')
93          AS R(IdReclamo INT, DNI_Cliente INT, FechaReclamo DATE,
                CodLocal VARCHAR, Descripcion VARCHAR, Estado VARCHAR)
94          UNION ALL
95          SELECT R.*
96          FROM dblink('dbname=db3 user=user3 password=password3 host=
                localhost port=5435',
97                      'SELECT * FROM reclamo_loc9_loc12 WHERE
                            DNI_Cliente >= 75000000')
98          AS R(IdReclamo INT, DNI_Cliente INT, FechaReclamo DATE,
                CodLocal VARCHAR, Descripcion VARCHAR, Estado VARCHAR)
99      ) AS temp_reclamo3
100     JOIN (
101         SELECT DNI_Cliente
102         FROM dblink('dbname=db1 user=user1 password=password1 host=
                localhost port=5433',
103                     'SELECT DNI_Cliente FROM venta_2000_2008 WHERE
                            DNI_Cliente >= 75000000')
104         AS V(DNI_Cliente INT)
105         UNION ALL
106         SELECT DNI_Cliente
107         FROM dblink('dbname=db2 user=user2 password=password2 host=
                localhost port=5434',
108                     'SELECT DNI_Cliente FROM venta_2009_2016 WHERE
                            DNI_Cliente >= 75000000')
109         AS V(DNI_Cliente INT)
110         UNION ALL
111         SELECT DNI_Cliente
112         FROM dblink('dbname=db3 user=user3 password=password3 host=
                localhost port=5435',
113                     'SELECT DNI_Cliente FROM venta_2017_2024 WHERE
                            DNI_Cliente >= 75000000')
114         AS V(DNI_Cliente INT)
115     ) AS temp_venta3
116     ON temp_reclamo3.DNI_Cliente = temp_venta3.DNI_Cliente
117 ) AS result3;
```

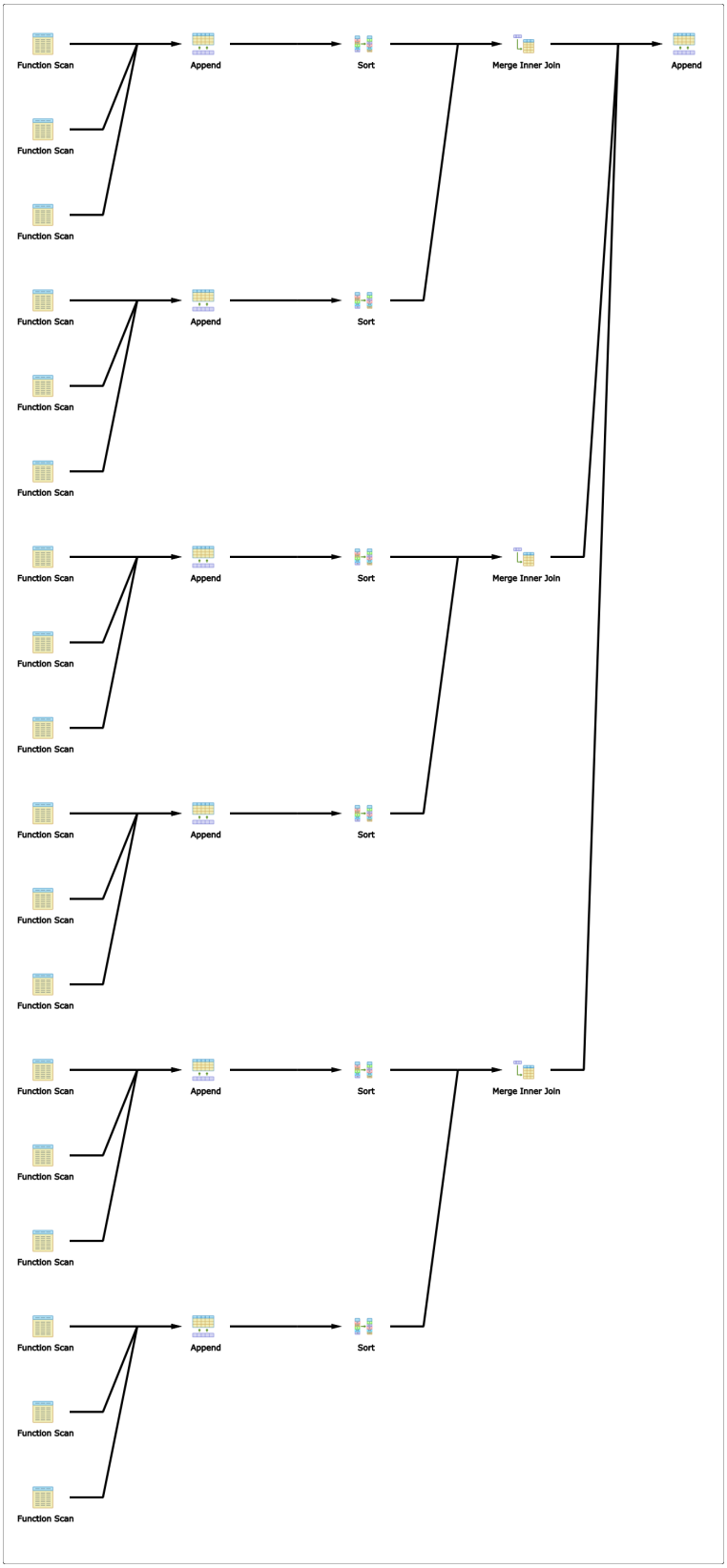

Figure 12: Optimización de la consulta 4 distribuida en diferentes servidores