Práctica 3

Fragmentación

Equipo 7
Pérez López Leonardo
Galicia Cobaxin Daniel

12 de junio de 2025

1. Conexion pre fragmentacion:

Antes de poder realizar la conexión, se tiene que crear una conexión por medio del **ODBC Data Sources**, en el cual se configura un **DSN**, el cual es una configuración que guarda los datos de conexión (host, usuario, contraseña, base de datos) para facilitar el acceso a MySQL desde aplicaciones, como en este caso *SQL Server Management*, para poder exportar los datos de la base de datos `covidHistorico`.

Se utiliza la **fragmentación horizontal**, utilizando la columna `ENTIDAD_RES` para poder repartir los datos a los diversos nodos (Norte, Sur, Centro). Se consultan mediante *consultas distribuidas*, usando `OPENQUERY` para acceder remotamente a cada nodo, donde se unifican los resultados con `UNION`.

```
-- Crear fragmentacin
CREATE DATABASE datoscovid_norte;
USE datoscovid_norte;
SELECT * INTO datoscovid_norte FROM covidHistorico.dbo.datoscovid
WHERE ENTIDAD_RES IN ('02','03','08','05','19','24','28','10','25');

CREATE DATABASE datoscovid_centro;
USE datoscovid_centro;
SELECT * INTO datoscovid_centro FROM covidHistorico.dbo.datoscovid
WHERE ENTIDAD_RES IN ('01','11','22','24','32','14','15','09','13','17','29','21');

CREATE DATABASE datoscovid_sur;
USE datoscovid_sur;
SELECT * INTO datoscovid_sur FROM covidHistorico.dbo.datoscovid
WHERE ENTIDAD_RES IN ('12','20','07','30','27','04','31','23','16','06','18');

-- En MySQL: crear base de datos y vaciar datos desde CSV
CREATE DATABASE covidHistorico;
USE covidHistorico;
-- Vaciar datos del archivo CSV

-- Consulta 3
SELECT
    (SUM(Porcentaje_Diabetes) / 3.0) AS Porcentaje_Diabetes,
    (SUM(Porcentaje_Obesidad) / 3.0) AS Porcentaje_Obesidad,
    (SUM(Porcentaje_Hipertension) / 3.0) AS Porcentaje_Hipertension
FROM (
    SELECT
        (SUM(CASE WHEN DIABETES = 1 THEN 1 ELSE 0 END) * 100.0 / COUNT(*)) AS Porcentaje_Diabetes,
        (SUM(CASE WHEN OBESIDAD = 1 THEN 1 ELSE 0 END) * 100.0 / COUNT(*)) AS Porcentaje_Obesidad,
        (SUM(CASE WHEN HIPERTENSION = 1 THEN 1 ELSE 0 END) * 100.0 / COUNT(*)) AS Porcentaje_Hipertension
    FROM OPENQUERY(SQL_Norte, 'SELECT * FROM datoscovid_norte WHERE CLASIFICACION_FINAL IN (1,2,3)')

    UNION ALL

    SELECT
        (SUM(CASE WHEN DIABETES = 1 THEN 1 ELSE 0 END) * 100.0 / COUNT(*)),
        (SUM(CASE WHEN OBESIDAD = 1 THEN 1 ELSE 0 END) * 100.0 / COUNT(*)),
        (SUM(CASE WHEN HIPERTENSION = 1 THEN 1 ELSE 0 END) * 100.0 / COUNT(*))
```

```
    FROM OPENQUERY(SQL_Centro, 'SELECT * FROM datoscovid_centro WHERE CLASIFICACION_FINAL IN (1,2,3)')

    UNION ALL

    SELECT
        (SUM(CASE WHEN DIABETES = 1 THEN 1 ELSE 0 END) * 100.0 / COUNT(*)),
        (SUM(CASE WHEN OBESIDAD = 1 THEN 1 ELSE 0 END) * 100.0 / COUNT(*)),
        (SUM(CASE WHEN HIPERTENSION = 1 THEN 1 ELSE 0 END) * 100.0 / COUNT(*))
    FROM OPENQUERY(MYSQL_SUR, 'SELECT * FROM covidHistorico_sur WHERE CLASIFICACION_FINAL IN (1,2,3)')
) AS SubResultados;

-- Consulta 4
SELECT DISTINCT Estado, Municipio
FROM (
    SELECT c.entidad AS Estado, d.MUNICIPIO_RES AS Municipio
    FROM OPENQUERY(SQL_Norte, 'SELECT d.ENTIDAD_RES, d.MUNICIPIO_RES FROM datoscovid_norte d WHERE NOT
         EXISTS (
        SELECT 1 FROM datoscovid_norte d2 WHERE d2.MUNICIPIO_RES = d.MUNICIPIO_RES AND d2.ENTIDAD_RES =
             d.ENTIDAD_RES
        AND d2.CLASIFICACION_FINAL IN (1,2,3) AND (d2.HIPERTENSION = 1 AND d2.OBESIDAD = 1 AND d2.DIABETES =
             1 AND d2.TABAQUISMO = 1))') d
    JOIN cat_entidades c ON d.ENTIDAD_RES = c.clave

    UNION

    SELECT c.entidad, d.MUNICIPIO_RES
    FROM OPENQUERY(SQL_Centro, 'SELECT d.ENTIDAD_RES, d.MUNICIPIO_RES FROM datoscovid_centro d WHERE NOT
         EXISTS (
        SELECT 1 FROM datoscovid_centro d2 WHERE d2.MUNICIPIO_RES = d.MUNICIPIO_RES AND d2.ENTIDAD_RES =
             d.ENTIDAD_RES
        AND d2.CLASIFICACION_FINAL IN (1,2,3) AND (d2.HIPERTENSION = 1 AND d2.OBESIDAD = 1 AND d2.DIABETES =
             1 AND d2.TABAQUISMO = 1))') d
    JOIN cat_entidades c ON d.ENTIDAD_RES = c.clave

    UNION

    SELECT c.entidad, d.MUNICIPIO_RES
    FROM OPENQUERY(MYSQL_SUR, 'SELECT d.ENTIDAD_RES, d.MUNICIPIO_RES FROM covidHistorico_sur d WHERE NOT
         EXISTS (
        SELECT 1 FROM covidHistorico_sur d2 WHERE d2.MUNICIPIO_RES = d.MUNICIPIO_RES AND d2.ENTIDAD_RES =
             d.ENTIDAD_RES
        AND d2.CLASIFICACION_FINAL IN (1,2,3) AND (d2.HIPERTENSION = 1 AND d2.OBESIDAD = 1 AND d2.DIABETES =
             1 AND d2.TABAQUISMO = 1))') d
    JOIN cat_entidades c ON d.ENTIDAD_RES = c.clave
) AS Resultados
ORDER BY Estado, Municipio;

-- Consulta 5
SELECT Estado, SUM(Total) AS Total_Casos_Recuperados_Neumonia
FROM (
    SELECT c.entidad AS Estado, COUNT(*) AS Total
    FROM OPENQUERY(SQL_Norte, 'SELECT d.ENTIDAD_RES FROM datoscovid_norte d WHERE d.CLASIFICACION_FINAL = 3
         AND d.NEUMONIA = 1') d
    JOIN cat_entidades c ON d.ENTIDAD_RES = c.clave
    GROUP BY c.entidad

    UNION ALL

    SELECT c.entidad, COUNT(*)
    FROM OPENQUERY(SQL_Centro, 'SELECT d.ENTIDAD_RES FROM datoscovid_centro d WHERE d.CLASIFICACION_FINAL =
         3 AND d.NEUMONIA = 1') d
    JOIN cat_entidades c ON d.ENTIDAD_RES = c.clave
    GROUP BY c.entidad

    UNION ALL

    SELECT c.entidad, COUNT(*)
    FROM OPENQUERY(MYSQL_SUR, 'SELECT d.ENTIDAD_RES FROM covidHistorico_sur d WHERE d.CLASIFICACION_FINAL =
         3 AND d.NEUMONIA = 1') d
    JOIN cat_entidades c ON d.ENTIDAD_RES = c.clave
    GROUP BY c.entidad
) AS Resultados
GROUP BY Estado
```

```sql
ORDER BY Total_Casos_Recuperados_Neumonia DESC;

-- Consulta 7
WITH CasosUnificados AS (
    SELECT
        YEAR(d.FECHA_INGRESO) AS Ao,
        MONTH(d.FECHA_INGRESO) AS Mes,
        c.entidad AS Estado,
        COUNT(*) AS Total_Casos
    FROM (
        SELECT * FROM OPENQUERY(SQL_Norte, 'SELECT FECHA_INGRESO, ENTIDAD_RES, CLASIFICACION_FINAL FROM
            datoscovid_norte WHERE YEAR(FECHA_INGRESO) IN (2020,2021)')
        UNION ALL
        SELECT * FROM OPENQUERY(SQL_Centro, 'SELECT FECHA_INGRESO, ENTIDAD_RES, CLASIFICACION_FINAL FROM
            datoscovid_centro WHERE YEAR(FECHA_INGRESO) IN (2020,2021)')
        UNION ALL
        SELECT * FROM OPENQUERY(MYSQL_SUR, 'SELECT FECHA_INGRESO, ENTIDAD_RES, CLASIFICACION_FINAL FROM
            covidHistorico_sur WHERE YEAR(FECHA_INGRESO) IN (2020,2021)')
    ) d
    JOIN cat_entidades c ON d.ENTIDAD_RES = c.clave
    WHERE d.CLASIFICACION_FINAL IN (1,2,3,6)
    GROUP BY YEAR(d.FECHA_INGRESO), MONTH(d.FECHA_INGRESO), c.entidad
),
RankingMensual AS (
    SELECT *, RANK() OVER (PARTITION BY Ao, Estado ORDER BY Total_Casos DESC) AS Ranking
    FROM CasosUnificados
)
SELECT Ao, Estado, Mes, Total_Casos
FROM RankingMensual
WHERE Ranking = 1
ORDER BY Ao, Estado;

-- Nombres de los linked servers:
-- MYSQL_SUR (hacia el nodo MySQL)
-- SQL_Centro (hacia el nodo SQL Server del centro)
-- SQL_Norte (hacia el nodo SQL Server del norte)
```

2. Conclusiones

Conclusión

La implementación de la fragmentación horizontal en la base de datos `covidHistorico` ha sido una experiencia para entender cómo las bases de datos distribuidas gestionan grandes volúmenes de información. Al fragmentar los datos utilizando la columna `ENTIDAD_RES`, que representa las entidades para distrubuir en los nodos Norte, Sur y Centro. Esto no solo refleja una separación lógica basada en criterios reales del dominio, sino que también facilita la administración y el acceso a los datos específicos de cada región.

Una de las ventajas más importantes de esta fragmentación es la escalabilidad que aporta al sistema. Al distribuir los datos en varios nodos, se reduce la carga de trabajo en cada servidor individual, lo que permite manejar un mayor volumen de datos sin sacrificar el rendimiento. Esto significa que, a medida que la cantidad de información crezca, el sistema podrá escalar agregando más nodos o distribuyendo los datos de manera más granular, manteniendo la eficiencia y la rapidez en las consultas.

Además, aunque los datos están físicamente distribuidos, la utilización de consultas distribuidas a través de `OPENQUERY` permite que el usuario final acceda a la información como si estuviera centralizada. Esto simplifica la interacción con la base de datos y asegura que las aplicaciones puedan trabajar con los datos de manera integrada, sin preocuparse por la ubicación física de cada fragmento.

También es importante destacar la importancia de mantener la coherencia en la estructura de los datos, ya que todos los fragmentos contienen las mismas columnas y formatos, asegurando que las consultas y operaciones puedan realizarse sin problemas en cualquier nodo.