



**INSTITUTO POLITÉCNICO NACIONAL**



**UNIDAD PROFESIONAL INTERDISCIPLINARIA EN  
INGENIERÍA Y TECNOLOGÍAS AVANZADAS**

**UPIITA**

**PROFESOR:** Carlos De La Cruz Sosa

**ASIGNATURA:** Bases de Datos Distribuidas

**GRUPO:** 3TM3

**Práctica III**

**EQUIPO 5:**

- Bernal Aguilar Yuvia Abigail
- Contreras Jimenez Mariana Montserrat
- Medina Gómez Jimena Zarahí

## Introducción:

El ámbito de las bases de datos distribuidas se ha vuelto crucial en la gestión de volúmenes de datos cada vez mayores, y la información relacionada con la pandemia de COVID-19 no es una excepción. La fragmentación, que implica dividir una base de datos grande en partes más pequeñas y manejables, es una estrategia clave para mejorar el rendimiento, la escalabilidad y la disponibilidad de los datos. Al distribuir estos fragmentos en múltiples nodos, podemos optimizar el acceso a la información y garantizar la resiliencia del sistema.

En esta práctica, explicaremos la aplicación de la fragmentación regional a una base de datos histórica de COVID-19 en México. La diversidad geográfica, económica y política del país ofrece una oportunidad única para diseñar un esquema de fragmentación que no solo mejore la eficiencia de las consultas, sino que también refleje la realidad sociodemográfica. El desafío radica en identificar clasificaciones regionales pertinentes y en implementar una infraestructura distribuida que permita el acceso transparente a los datos desde cualquier punto de la red.

La interconexión de nodos heterogéneos, incluyendo servidores MySQL y SQL Server, planteará consideraciones sobre la interoperabilidad y la gestión de transacciones distribuidas. Al adaptar consultas existentes para operar sobre estos fragmentos distribuidos, profundizaremos en los principios de las consultas distribuidas y en las técnicas para optimizar su ejecución en un entorno fragmentado. Esta práctica no solo busca la implementación técnica de una solución distribuida, sino también la comprensión crítica de las implicaciones de diseño y rendimiento en sistemas de bases de datos de gran escala.

## Desarrollo:

1. Para la distribución de los fragmentos se consideran 4 nodos, donde cada uno se hará por la fragmentación por regiones las cuales se dividieron de la siguiente manera:

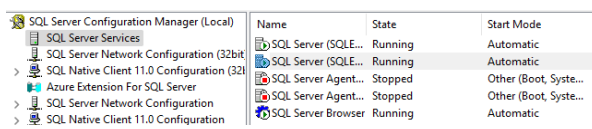
Norte	Baja California, Baja California Sur, Chihuahua, Coahuila, Durango, Sonora, Sinaloa, Nuevo León
Centro-Norte	Aguascalientes, Zacatecas, San Luis Potosí, Guanajuato, Querétaro, Jalisco, Nayarit, Colima
Centro-Sur	Ciudad de México, Estado de México, Morelos, Puebla, Tlaxcala, Hidalgo, Michoacán, Guerrero
Sur	Veracruz, Oaxaca, Chiapas, Tabasco, Campeche, Yucatán, Quintana Roo, Tamaulipas

2. Estos nodos se conectan a una base de datos diferente, esta distribución queda de la siguiente manera:

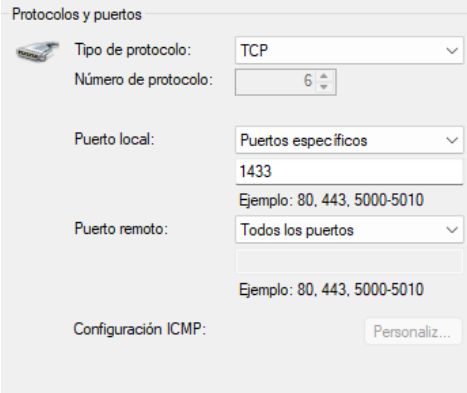
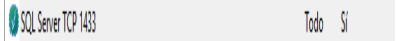
- Norte ==> SQL Server Management Studio (Yuvia)
- Centro\_Norte ==> SQL Server Management Studio (Monse)
- Centro\_Sur ==> SQL Server Management Studio (Jimena)
- Sur ==> MySql

a) Procedimiento general para la conexión remota de todos los nodos que pertenecen a SQL Server Management Studio.

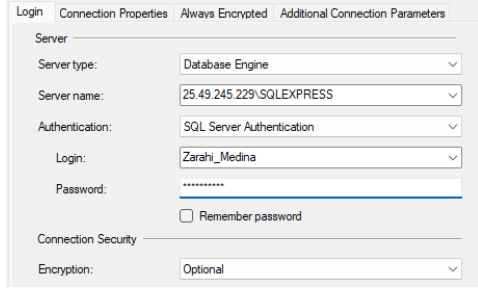
1. Verificar en SQL Server Manager 16.msc esté activo la siguiente:

	<table border="1"> <thead> <tr> <th>Protocol Name</th><th>Status</th></tr> </thead> <tbody> <tr> <td>Shared Memory</td><td>Enabled</td></tr> <tr> <td>Named Pipes</td><td>Enabled</td></tr> <tr> <td>TCP/IP</td><td>Enabled</td></tr> </tbody> </table>	Protocol Name	Status	Shared Memory	Enabled	Named Pipes	Enabled	TCP/IP	Enabled
Protocol Name	Status								
Shared Memory	Enabled								
Named Pipes	Enabled								
TCP/IP	Enabled								
<p>El servidor este corriendo de forma adecuada y automática así como los puertos bien configurados tanto para el servidor que será la parte de entrada y el browser que será la parte que recibe.</p>	<p>El protocolo este corriendo y activo sobretodo para el protocolo TCP.</p>								

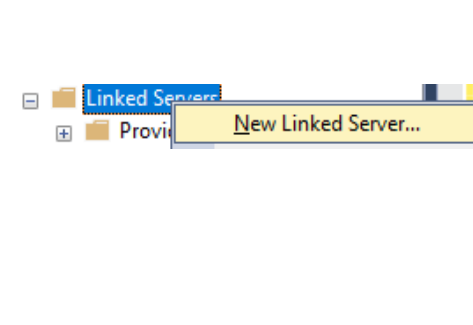
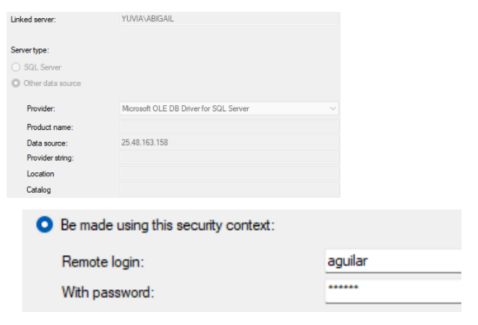
2. Verificar la entrada del puerto en el Firewall de Windows y si no existe crearlo. En nuestro caso particular dos se generaron y el otro solo se comprobó.

	
<p>Se creó una entrada la cual lleva la configuración con el puerto 1433, Se le permite la conexión tanto para recibir como enviar de todos los lados y los puertos.</p>	<p>Aquí se comprueba que esté habilitado y permite la conexión.</p>

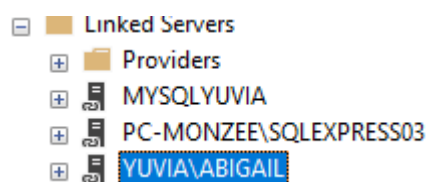
- Una vez creado los puertos (si es el caso) o verificado que estén habilitados y bien configurados, se procede a descargar Hamachi Para que la red se vuelva una sola con diferentes IP'S y se prueba la conexión de la base de datos con esa ip de manera local.

	
Todos los equipos conectados a una sola red de IP'S y que esten activos.	Verificando la red IP en la base de datos, conectando con el usuario administrador (tener todos los permisos habilitados.)

- Por último se procede a hacer la conexión remota a cada usuario en la base de datos, en la sección de Linked Server.

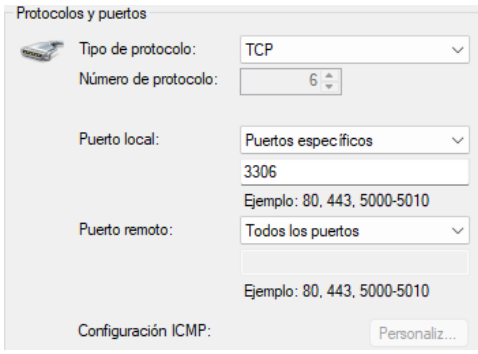
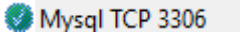
	
Se presiona clic derecho y se accede a la sección amarilla.	En la pestaña que se abrirá en la sección general se coloca en <u>Linked server. Provider y Data source</u> . Después se va a la sección de <u>security</u> hasta abajo y se coloca la opción señalada junto con el usuario y contraseña que nos queremos conectar.

- Una vez hecho este proceso se verifica que aparezca la conexión en la sección de Linked server.

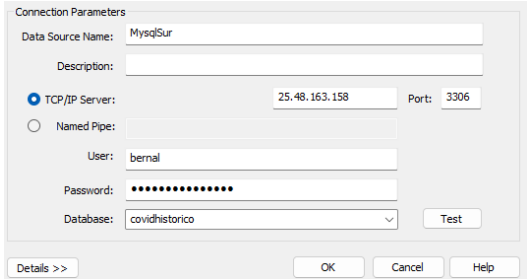
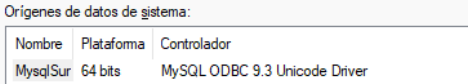


b) Procedimiento general para la conexión remota del último nodo (Sur) que pertenece a Mysql..

1. Abrimos el Firewall y creamos una nueva entrada para Mysql con el puerto 3306.

	
<p>Aquí se agrega la entrada nueva en el puerto TCP para MySQL para recibir y mandar.</p>	<p>Aquí se comprueba que el puerto para MySQL quedó habilitado</p>

2. Abrimos el ODBC de 64 bits, ingresamos a DSN sistema Y agregamos uno nuevo. En el caso que no aparezca la sección de MySQL ODBC 9.3 Unicode Driver se instala y luego se programa de la siguiente manera.

	
<p>Se le pone un nombre del data source como nosotros lo identifiquemos, seleccionamos TCP y colocamos la IP de la maquina virtual el puerto sera 3306, usuario es el que esta registrado en MySQL despues de eso aparecera automaticamente la base de datos y se hace el test. Terminando este proceso debe salir una leyenda que fue conectado exitosamente</p>	<p>Una vez realizado lo anterior, se verifica que este conectado en esta sección.</p>

Después se repetirán los pasos a) 3.4 & 5. para la base de datos.

c) Modificaciones de las consultas 3,4,5 y 7 de la Práctica 1 para transformarlas en consultas distribuidas.

1. Consulta 3) Porcentaje de casos confirmados en diabetes, obesidad e hipertensión.

<b>No. Consulta</b>	3
<b>Descripción</b>	Porcentaje de casos confirmados de diabetes, obesidad e hipertensión.
<b>Significado de los valores de los catálogos</b>	<ul style="list-style-type: none"> <li>- Diabetes: 1= El paciente tiene diabetes / 2= El paciente no tiene diabetes</li> <li>- Obesidad: 1= El paciente tiene obesidad / 2= El paciente no tiene obesidad</li> <li>- Hipertension: 1= El paciente tiene hipertension / 2= El paciente no tiene hipertension</li> <li>- CLASIFICACION_FINAL = 3 : Casos confirmados por laboratorio</li> </ul>
<b>Responsable</b>	<b>Bernal Aguilar Yuvia Abigail</b>
<b>Comentarios</b>	<ul style="list-style-type: none"> <li>- SUM: Usado para la suma de todos los casos y la poblacion.</li> <li>- CASE WHEN: Determina que tipo de caso es, si DIABETES, OBESIDAD E HIPERTENSION son positivos y ademas, la CALIFICACION_FINAL es igual a 3 , entonces suma a 1 si no, suma 0.</li> <li>- UNION ALL: Lo usamos para combinar dos o más o consultas en un solo conjunto de resultados.</li> <li>- GROUP BY: Sirve para agrupar filas con mismos valores.</li> </ul>

Consulta:

```

SELECT
    morbilidad, |
    SUM(casos) AS casos_totales,
    SUM(poblacion) AS poblacion_total,
    (SUM(casos) * 100.0 / SUM(poblacion)) AS porcentaje
FROM (
    SELECT
        'Diabetes' AS morbilidad,
        SUM(CASE WHEN DIABETES = 1 AND CLASIFICACION_FINAL = 3 THEN 1 ELSE 0 END) AS casos,
        SUM(CASE WHEN CLASIFICACION_FINAL = 3 THEN 1 ELSE 0 END) AS poblacion
    FROM [PC-MONZEE\SQLEXPRESS03].[covidHistorico].[dbo].[datoscovid centro norte]
    UNION ALL
    SELECT
        'Obesidad' AS morbilidad,
        SUM(CASE WHEN OBESIDAD = 1 AND CLASIFICACION_FINAL = 3 THEN 1 ELSE 0 END),
        SUM(CASE WHEN CLASIFICACION_FINAL = 3 THEN 1 ELSE 0 END)
    FROM [PC-MONZEE\SQLEXPRESS03].[covidHistorico].[dbo].[datoscovid centro norte]
    UNION ALL
    SELECT
        'Hipertensión' AS morbilidad,
        SUM(CASE WHEN HIPERTENSION = 1 AND CLASIFICACION_FINAL = 3 THEN 1 ELSE 0 END),
        SUM(CASE WHEN CLASIFICACION_FINAL = 3 THEN 1 ELSE 0 END)
    FROM [PC-MONZEE\SQLEXPRESS03].[covidHistorico].[dbo].[datoscovid centro norte]
    UNION ALL
    SELECT
        'Diabetes' AS morbilidad,
        SUM(CASE WHEN DIABETES = 1 AND CLASIFICACION_FINAL = 3 THEN 1 ELSE 0 END),
        SUM(CASE WHEN CLASIFICACION_FINAL = 3 THEN 1 ELSE 0 END)
    FROM [25.49.245.229].[covidHistorico].[dbo].[datoscovid centro sur]
    UNION ALL

```

```

UNION ALL
SELECT
    'Obesidad' AS morbilidad,
    SUM(CASE WHEN OBESIDAD = 1 AND CLASIFICACION_FINAL = 3 THEN 1 ELSE 0 END),
    SUM(CASE WHEN CLASIFICACION_FINAL = 3 THEN 1 ELSE 0 END)
FROM [25.49.245.229].[covidHistorico].[dbo].[datoscovid_centro_sur]
UNION ALL
SELECT
    'Hipertensión' AS morbilidad,
    SUM(CASE WHEN HIPERTENSION = 1 AND CLASIFICACION_FINAL = 3 THEN 1 ELSE 0 END),
    SUM(CASE WHEN CLASIFICACION_FINAL = 3 THEN 1 ELSE 0 END)
FROM [25.49.245.229].[covidHistorico].[dbo].[datoscovid_centro_sur]
UNION ALL
SELECT
    'Diabetes' AS morbilidad,
    SUM(CASE WHEN DIABETES = 1 AND CLASIFICACION_FINAL = 3 THEN 1 ELSE 0 END),
    SUM(CASE WHEN CLASIFICACION_FINAL = 3 THEN 1 ELSE 0 END)
FROM [YUVIA\ABIGAIL].[covidHistorico].[dbo].[datoscovid_norte]
UNION ALL
SELECT
    'Obesidad' AS morbilidad,
    SUM(CASE WHEN OBESIDAD = 1 AND CLASIFICACION_FINAL = 3 THEN 1 ELSE 0 END),
    SUM(CASE WHEN CLASIFICACION_FINAL = 3 THEN 1 ELSE 0 END)
FROM [YUVIA\ABIGAIL].[covidHistorico].[dbo].[datoscovid_norte]
UNION ALL
SELECT
    'Hipertensión' AS morbilidad,
    SUM(CASE WHEN HIPERTENSION = 1 AND CLASIFICACION_FINAL = 3 THEN 1 ELSE 0 END),
    SUM(CASE WHEN CLASIFICACION_FINAL = 3 THEN 1 ELSE 0 END)
FROM [YUVIA\ABIGAIL].[covidHistorico].[dbo].[datoscovid_norte]
UNION ALL
SELECT
    'Diabetes' AS morbilidad,
    SUM(CASE WHEN DIABETES = 1 AND CLASIFICACION_FINAL = 3 THEN 1 ELSE 0 END),
    SUM(CASE WHEN CLASIFICACION_FINAL = 3 THEN 1 ELSE 0 END)
FROM OPENQUERY([NODO_MYSQL], 'SELECT * FROM covidhistorico.datoscovid')
UNION ALL
SELECT
    'Obesidad' AS morbilidad,
    SUM(CASE WHEN OBESIDAD = 1 AND CLASIFICACION_FINAL = 3 THEN 1 ELSE 0 END),
    SUM(CASE WHEN CLASIFICACION_FINAL = 3 THEN 1 ELSE 0 END)
FROM OPENQUERY([NODO_MYSQL], 'SELECT * FROM covidhistorico.datoscovid')
UNION ALL
SELECT
    'Hipertensión' AS morbilidad,
    SUM(CASE WHEN HIPERTENSION = 1 AND CLASIFICACION_FINAL = 3 THEN 1 ELSE 0 END),
    SUM(CASE WHEN CLASIFICACION_FINAL = 3 THEN 1 ELSE 0 END)
FROM OPENQUERY([NODO_MYSQL], 'SELECT * FROM covidhistorico.datoscovid')
) AS combined_data
GROUP BY morbilidad;

```

## Resultado:

Results		Messages		
	morbilidad	casos_totales	poblacion_total	porcentaje
1	Diabetes	509300	5266753	9.670094648448
2	Hipertensión	678389	5266753	12.880592653576
3	Obesidad	567148	5266753	10.768456390493

2. Consulta 4) Municipios que no tienen casos confirmados en Hipertensión, obesidad, diabetes y tabaquismo.

<b>No. Consulta</b>	4
<b>Descripción</b>	Municipios que no tienen casos confirmados en Hipertensión, obesidad, diabetes y tabaquismo.
<b>Significado de los valores de los catálogos</b>	<ul style="list-style-type: none"> <li>- MUNICIPIO_RES: Municipios que no tienen casos confirmados con las enfermedades pedidas.</li> <li>- CLASIFICACION_FINAL = 3 : Ese valor son casos negativos o no confirmados.</li> <li>- DIABETES = 1, OBESIDAD = 1 &amp; TABAQUISMO = 1: Este valor representa que si tenga esa enfermedad.</li> </ul>
<b>Responsable</b>	Medina Gómez Jimena Zarahí
<b>Comentarios</b>	<ul style="list-style-type: none"> <li>- UNION: Se usa este ya que nos ayuda a que no se repita ningún valor.</li> <li>- NOT IN: Es un filtro para encontrar municipios (si es el caso) que no estén incluidos en las comorbilidades.</li> <li>- DISTINCT: De igual manera se evita la repetición de municipios.</li> <li>- OPENQUERY: Esto nos ayudará a la conexión con el servidor de MySQL y permita que accedemos a los datos desde SQL Server.</li> <li>- INTO #temporal_municipios: Aquí se guardaran los datos temporalmente en una tabla para reducir el tiempo de ejecución.</li> <li>- DROP TABLE: Aquí se elimina la tabla temporal una vez recuperados los datos, para mandar a la tabla de los resultados.</li> </ul>

Consulta:

```

SQLQuery6.sql - 25...Zarahi_Medina (61))" * X SQLQuery13.sql - 2...Zarahi_Medina (59))" SQLQuery12.sql - 2...Zarahi_Medina (60))" SQLQuery11.
--Consulta 4: Municipios que no tienen casos confirmados en Hipertensión, obesidad, diabetes y tabaquismo.
SELECT DISTINCT MUNICIPIO_RES INTO #temporal_municipios
FROM (
    SELECT MUNICIPIO_RES
    FROM [YUVIA\ABIGAIL].[covidHistorico].[dbo].[datoscovid_norte]
    WHERE CLASIFICACION_FINAL = 3 AND (DIABETES = 1 OR OBESIDAD = 1 OR TABAQUISMO = 1)
    UNION
    SELECT MUNICIPIO_RES
    FROM [PC-MONZEE\SQLEXPRESS03].[covidHistorico].[dbo].[datoscovid_centro_norte]
    WHERE CLASIFICACION_FINAL = 3 AND (DIABETES = 1 OR OBESIDAD = 1 OR TABAQUISMO = 1)
    UNION
    SELECT MUNICIPIO_RES
    FROM covidHistorico.dbo.[datoscovid_centro_sur]
    WHERE CLASIFICACION_FINAL = 3 AND (DIABETES = 1 OR OBESIDAD = 1 OR TABAQUISMO = 1)
    UNION
    SELECT MUNICIPIO_RES
    FROM OPENQUERY([MYSQLYUVIA], '
        SELECT MUNICIPIO_RES
        FROM datoscovid
        WHERE CLASIFICACION_FINAL = 3
        AND (DIABETES = 1 OR OBESIDAD = 1 OR TABAQUISMO = 1)
    ')
) AS datos;

SQLQuery6.sql - 25...Zarahi_Medina (61))" * X SQLQuery13.sql - 2...Zarahi_Medina (59))" SQLQuery12.sql - 2...Zarahi_Medina (60))" SQLQuery
SELECT DISTINCT MUNICIPIO_RES
FROM (
    SELECT MUNICIPIO_RES FROM [YUVIA\ABIGAIL].[covidHistorico].[dbo].[datoscovid_norte]
    UNION
    SELECT MUNICIPIO_RES FROM [PC-MONZEE\SQLEXPRESS03].[covidHistorico].[dbo].[datoscovid_centro_norte]
    UNION
    SELECT MUNICIPIO_RES FROM covidHistorico.dbo.[datoscovid_centro_sur]
    UNION
    SELECT MUNICIPIO_RES FROM OPENQUERY([MYSQLYUVIA], 'SELECT MUNICIPIO_RES FROM datoscovid')
) AS todos_municipios
WHERE MUNICIPIO_RES NOT IN (SELECT MUNICIPIO_RES FROM #temporal_municipios)
ORDER BY MUNICIPIO_RES;
DROP TABLE #temporal_municipios;

```



Resultado:

MUNICIPIO_RES		MUNICIPIO_RES		MUNICIPIO_RES	
1	218	21	311	41	416
2	222	22	313	42	423
3	223	23	317	43	430
4	225	24	322	44	436
5	228	25	323	45	440
6	229	26	328	46	444
7	234	27	331	47	448
8	236	28	332	48	451
9	240	29	335	49	454
10	245	30	354	50	465
11	260	31	357	51	473
12	264	32	359	52	476
13	267	33	361	53	477
14	268	34	371	54	478
15	274	35	374	55	479
16	275	36	383	56	481
17	281	37	394	57	490
18	282	38	396	58	491
19	284	39	402	59	493
20	288	40	408	60	503
21	311	41	416	61	504
				62	506
				63	511
				64	512
				65	514
				66	521
				67	522
				68	527
				69	529
				70	541
				71	543
				72	564
				73	566

### 3. Consulta 5) Estados con más casos recuperados de Neumonía.

<b>No. Consulta</b>	5
<b>Descripción</b>	Estados con más casos recuperados con neumonía.
<b>Significado de los valores de los catálogos</b>	<ul style="list-style-type: none"> <li>• ENTIDAD_UM - Estado de nacimiento de una persona, su valor es un código asociado a una entidad federativa.</li> <li>• NEUMONIA - Indica si el paciente tiene neumonía, en dado caso que sí = 1, 2 = No.</li> <li>• CLASIFICACIÓN_FINAL - El estado del caso = 3, significa "caso confirmado por prueba de laboratorio".</li> <li>• FECHA_DEF - Si es NULL, se asume que el paciente no falleció, es decir, posible recuperado.</li> </ul>
<b>Responsable</b>	<b>Contreras Jiménez Mariana Montserrat</b>
<b>Comentarios</b>	<ul style="list-style-type: none"> <li>• Estrategia - Ya que no hay una columna clara para poder identificar recuperados, se hace una inferencia: se asume que si el paciente tuvo neumonía (NEUMONIA = 1) y no ha fallecido (FECHA_DEF = 9-99-99'), es probable que se haya recuperado.</li> <li>• '9999-99-99' en lugar de NULL - Convención que indica que el paciente no ha fallecido.</li> <li>• UNION ALL - Se utiliza para juntar los datos de los 4</li> </ul>

	<p>fragmentos (3 en SQL Server y 1 en MySQL). No se ocupa solamente UNION para evitar duplicados automáticamente.</p> <ul style="list-style-type: none"> <li>● COUNT(*) y GROUP BY ENTIDAD_UM - Sirve para agrupar los datos por estado (ENTIDAD_UM) y contar cuántos casos “recuperados con neumonía” hay en cada uno.</li> <li>● ORDER BY - Sirve para que aparezcan los estados con mayor número de casos en la parte superior.</li> <li>● OPENQUERY - Se utilizó para conectar con MySQL mediante un linked server.</li> </ul>
--	--

Consulta:

```

----- Consulta 5 -----
SELECT ENTIDAD_UM, COUNT(*) AS Casos_recuperados
FROM (
  -- Nodo centro_norte (local)
  SELECT ENTIDAD_UM
  FROM [PC-MONZEE\SQLEXPRESS03].[covidHistorico].[dbo].[datoscovid_centro_norte]
  WHERE NEUMONIA = 1 AND FECHA_DEF = '9999-99-99'
  UNION ALL
  -- Nodo norte
  SELECT ENTIDAD_UM
  FROM [YUVIA].[covidHistorico].[dbo].[datoscovid_norte]
  WHERE NEUMONIA = 1 AND FECHA_DEF = '9999-99-99'
  UNION ALL
  -- Nodo centro_sur
  SELECT ENTIDAD_UM
  FROM [25.49.245.229\SQLEXPRESS].[covidHistorico].[dbo].[datoscovid_centro_sur]
  WHERE NEUMONIA = 1 AND FECHA_DEF = '9999-99-99'
  UNION ALL
  -- Nodo CENTRO MYSQLYUVIA
  SELECT ENTIDAD_UM
  FROM OPENQUERY(MYSQLYUVIA, '
    SELECT ENTIDAD_UM
    FROM covidhistorico.datoscovid
    WHERE NEUMONIA = 1 AND FECHA_DEF = ''9999-99-99''
  ')
) AS datos_unidos
GROUP BY ENTIDAD_UM
ORDER BY Casos_recuperados DESC;

```

Resultados:

100 %	Results	Messages
	ENTIDAD_UM	Casos_recuperados
1	09	105372
2	15	59452
3	21	27802
4	11	18923
5	30	18532
6	14	17308
7	02	15457
8	19	15237
9	08	13458
10	13	13204
11	16	12830
12	25	12386
13	26	12325
14	22	10976
15	27	10449
16	24	10326

100 %	Results	Messages
	ENTIDAD_UM	Casos_recuperados
17	12	10110
18	17	9304
19	05	9253
20	28	9012
21	04	6511
22	23	6265
23	31	6112
24	20	6043
25	29	5886
26	01	5076
27	03	4621
28	32	4121
29	18	3439
30	06	3326
31	07	2822
32	10	2716

4. Consulta 7) Para el año 2020 y 2021 cual fue el mes con más casos registrados, confirmados, sospechosos, por estado registrado en la base de datos.

<b>No. Consulta</b>	7
<b>Descripción</b>	Para el año 2020 y 2021 cual fue el mes con más casos registrados, confirmados, sospechosos, por estado registrado en la base de datos.
<b>Significado de los valores de los catálogos</b>	<ul style="list-style-type: none"> <li>• ENTIDAD_UM - Estado de nacimiento de una persona, su valor es un código asociado a una entidad federativa.</li> <li>• CLASIFICACIÓN_FINAL - El estado del caso = 3, significa "caso confirmado por prueba de laboratorio".</li> <li>• FECHA_INGRESO: Indica fecha de ingreso del paciente a la unidad médica.</li> </ul>
<b>Responsable</b>	<b>Medina Gómez Jimena Zarahí</b>
<b>Comentarios</b>	<ul style="list-style-type: none"> <li>- INTO: Creación de tablas.</li> <li>- WHERE: Filtro que muestra los datos necesarios para la consulta</li> <li>- UNION ALL: Une dos o más consultas para crear un solo conjunto de resultados.</li> <li>- OPENQUERY: Hace la consulta desde MySQL.</li> <li>- WITH: Realiza una consulta temporal.</li> <li>- ORDER BY: Ordena el resultado final de la consulta.</li> <li>- ON: Se utiliza con JOIN, este indica cómo deben unirse las tablas.</li> <li>- GROUP BY: Sirve para agrupar filas con mismos valores.</li> <li>- DROP: Elimina la tabla creada anteriormente.</li> </ul>

```

SELECT ENTIDAD_UM, FECHA_INGRESO, CLASIFICACION_FINAL
INTO #temporal_casos
FROM (

    SELECT ENTIDAD_UM, FECHA_INGRESO, CLASIFICACION_FINAL
    FROM [YUVIA\ABIGAIL].[covidHistorico].[dbo].[datos_covid_norte]
    WHERE CLASIFICACION_FINAL IN (3, 6) AND YEAR(FECHA_INGRESO) IN (2020, 2021)

    UNION ALL

    SELECT ENTIDAD_UM, FECHA_INGRESO, CLASIFICACION_FINAL
    FROM [PC-MONZEE\SQLEXPRESS03].[covidHistorico].[dbo].[datos_covid_centro_norte]
    WHERE CLASIFICACION_FINAL IN (3, 6) AND YEAR(FECHA_INGRESO) IN (2020, 2021)

    UNION ALL

    -- Centro Sur
    SELECT ENTIDAD_UM, FECHA_INGRESO, CLASIFICACION_FINAL
    FROM [25.49.245.229].[covidHistorico].[dbo].[datos_covid_centro_sur]
    WHERE CLASIFICACION_FINAL IN (3, 6) AND YEAR(FECHA_INGRESO) IN (2020, 2021)

    UNION ALL

    SELECT ENTIDAD_UM, FECHA_INGRESO, CLASIFICACION_FINAL
    FROM OPENQUERY([NODO_MYSQL], '
        SELECT ENTIDAD_UM, FECHA_INGRESO, CLASIFICACION_FINAL
        FROM datos_covid
        WHERE CLASIFICACION_FINAL IN (3, 6)
        AND YEAR(FECHA_INGRESO) IN (2020, 2021)
    ')

```

```

    ')
) AS datos_combinados;

WITH casos AS (
    SELECT
        ENTIDAD_UM AS entidad,
        YEAR(FECHA_INGRESO) AS año,
        MONTH(FECHA_INGRESO) AS mes,
        SUM(CASE WHEN CLASIFICACION_FINAL = 3 THEN 1 ELSE 0 END) AS casos_confirmados,
        SUM(CASE WHEN CLASIFICACION_FINAL = 6 THEN 1 ELSE 0 END) AS casos_sospechosos
    FROM #temporal_casos
    GROUP BY ENTIDAD_UM, YEAR(FECHA_INGRESO), MONTH(FECHA_INGRESO)
)

SELECT c.entidad, c.año, c.mes, c.casos_confirmados, c.casos_sospechosos
FROM casos c
JOIN (
    SELECT entidad, año, MAX(casos_confirmados + casos_sospechosos) AS m_casos
    FROM casos
    GROUP BY entidad, año
) maximos
ON c.entidad = maximos.entidad
AND c.año = maximos.año
AND (c.casos_confirmados + c.casos_sospechosos) = maximos.m_casos
ORDER BY c.entidad, c.año;

DROP TABLE #temporal_casos;

```

## Resultado:

	entidad	año	mes	casos_confirmados	casos_sospechosos
1	01	2020	11	4673	879
2	01	2021	1	3810	926
3	02	2020	12	7610	4705
4	02	2021	11	11549	700
5	03	2020	12	3364	32
6	03	2021	7	8619	53
7	04	2020	7	2251	1207
8	04	2021	8	5454	397
9	05	2020	11	8403	1590
10	05	2021	1	9233	884
11	06	2020	8	1831	140
12	06	2021	8	10292	162
13	07	2020	6	2616	10409
14	07	2021	8	4369	5740
15	08	2020	10	13496	360
16	08	2021	12	8548	266

✔ Query executed successfully.

## **Conclusiones:**

- Bernal Aguilar Yuvia Abigail

Al querer hacer una consulta, muchas veces podemos llegar a toparnos con dificultades, como la falta de información que puede encontrarse en otra base de datos. Para eso, es necesario la vinculación de servidores, los cuales, al estar conectados, podemos llegar a realizar consultas desde cualquier otra parte solo con la condición de compartir la misma red de conexión.

La práctica realizada nos permitió no solamente conectar servidores de SQL Server si no también ampliar nuestras fronteras hasta MySQL, la cual, albergaba una fragmentación de la base de datos madre, CovidHistorico. Esto nos llevó a nuevos descubrimientos, conocer cada uno de los detalles necesarios para una conexión exitosa y al igual, consultas exitosas.

Si bien, es una gran herramienta, esta puede llegar a ser algo complicada ya que las consultas pueden llegar a tardar en ejecutarse pero esto puede ser solucionado al usar funciones correctas al igual que una mejor organización en cómo hacemos las consultas.

- Medina Gómez Jimena Zarahí

La implementación de consultas distribuidas en esta práctica nos permitió comprender de manera práctica cómo operar con bases de datos fragmentadas y alojadas en distintos servidores, tanto en SQL Server como en MySQL. También comprendimos los procedimientos generales para las conexiones remotas entre los nodos de SQL Server Management Studio y con el nodo MySQL, incluyendo la verificación de protocolos activos (como TCP/IP en el puerto 1433 para SQL Server y 3306 para MySQL), la configuración de reglas de firewall, y la configuración de servidores vinculados (Linked Servers).

Al modificar las consultas de la práctica 1 note que había una optimización para cada consulta y que facilita su manejo siempre y cuando se cuente con una buena conexión remota. Su implementación puede ser compleja y puede resultar en tiempos de ejecución de consultas más largos. Sin embargo, la vinculación exitosa de servidores y la ejecución de consultas a través de SQL Server y MySQL nos dan grandes beneficios para acceder a datos fragmentados y simultáneos.

- Contreras Jiménez Mariana Montserrat

Esta práctica fue más que una implementación técnica; fue una experiencia que nos llevó a entender cómo la fragmentación de datos en múltiples nodos distribuidos (incluso en motores distintos como SQL Server y MySQL) exige no solo conocimiento en consultas, sino también en interoperabilidad y arquitectura de red. Nos enfrentamos a desafíos que normalmente no se ven en bases de datos centralizadas: desde errores por comillas mal escapadas hasta problemas de latencia por enlaces remotos. Estos detalles nos enseñaron que una consulta distribuida no solo se diseña bien, sino que también se piensa estratégicamente, considerando tanto la estructura de datos como la infraestructura detrás.

También aprendimos a transformar limitaciones en oportunidades: la ausencia de una columna que indicara directamente si un paciente se había recuperado nos llevó a plantear una lógica propia, razonando con los valores disponibles. Esa capacidad de inferencia es quizá el aprendizaje más valioso de todos: no siempre los datos nos lo dicen todo, a veces tenemos que saber cómo escucharlos.

En definitiva, trabajar con bases de datos distribuidas no es solo cuestión de enlazar servidores, sino de aprender a conversar con los datos que están lejos, dispersos, pero que juntos forman una sola historia.

## **Conclusión General:**

La práctica de bases de datos distribuidas nos permitió ir más allá del simple diseño de consultas: nos llevó a explorar cómo interactúan sistemas heterogéneos en un entorno real, donde los datos están fragmentados entre servidores SQL Server y MySQL, y sin embargo, deben comportarse como una sola unidad lógica. Este proceso implicó no sólo dominar las instrucciones SQL, sino también configurar adecuadamente conexiones remotas, protocolos, servidores vinculados y resolver errores derivados de diferencias sintácticas entre motores.

Al enfrentarnos con desafíos como la ausencia de indicadores directos de recuperación o tiempos de respuesta prolongados, aprendimos a ser ingeniosos en el diseño de nuestras consultas y pacientes en la interpretación de los resultados. La fragmentación no fue una barrera, sino una oportunidad para optimizar, inferir y reorganizar el acceso a la información.

En conjunto, esta experiencia nos enseñó que trabajar con bases distribuidas no es solo cuestión de tecnología, sino de pensamiento estratégico, análisis crítico y adaptación. El verdadero valor está en saber unir piezas dispersas para descubrir patrones, responder preguntas complejas y, sobre todo, diseñar soluciones funcionales en entornos reales y distribuidos.