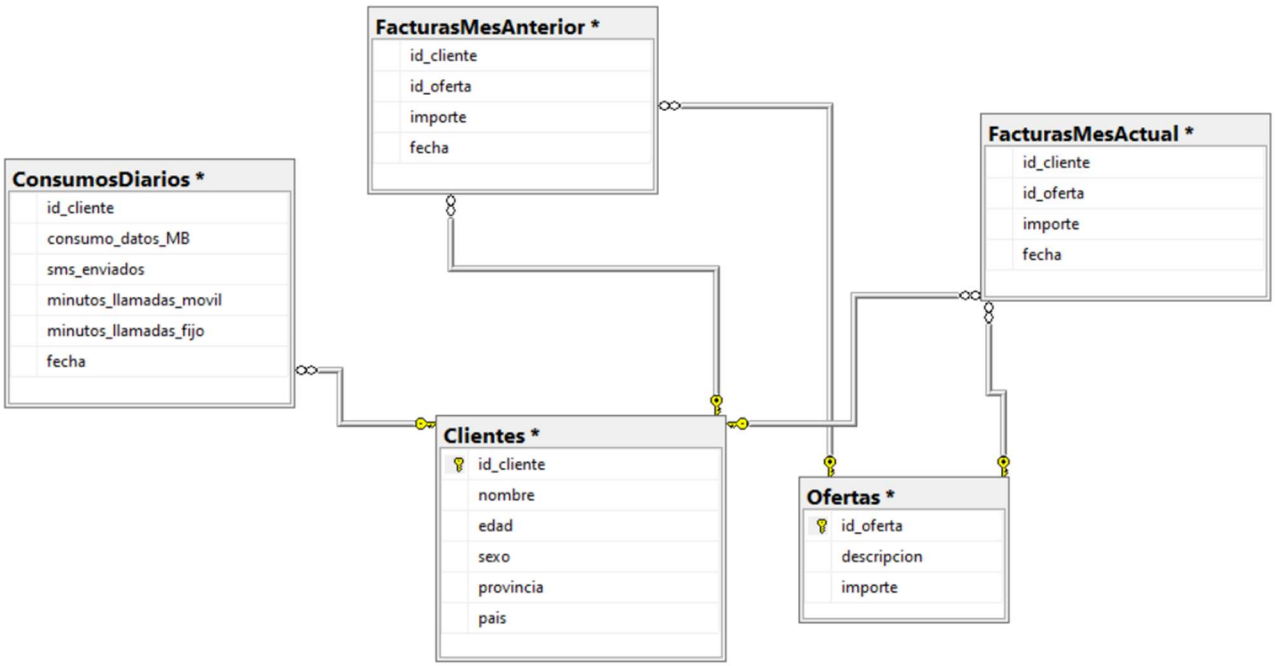
Descripción corta del challenge

A partir de una BD de clientes de una empresa de telefonía, se realizan una serie de consultas con lenguaje SQL (SQL Server).

Diagrama Entidad Relación de la BD



Tablas de la BD

A continuación se muestran las tablas que componen la BD

- Clientes

Results | Messages

	id_cliente	nombre	edad	sexo	provincia	pais
1	1	Pablo Perez	26	M	Madrid	España
2	2	Eduardo Redondo	58	M	Bogota	Colombia
3	3	Roberto Salazar	68	M	Monterrey	Mexico
4	4	Pedro Conde	32	M	Madrid	España
5	5	Ana Robles	41	F	Valladolid	España
6	6	David Roldan	74	M	Guadalajara	Mexico
7	7	Carmen Arauzo	19	F	Medellin	Colombia
8	8	Silvia Rodriguez	49	F	Barcelona	España
9	9	David Cardoso	24	M	Cali	Colombia
10	10	Pablo Lopez	40	M	Sevilla	España
11	11	Luis Rodrigues	20	M	Bogota	Colombia
12	12	Alvaro Monroy	75	M	Monterrey	Mexico
13	13	Victor Ruiz	53	M	Valladolid	España
14	14	Melisa Aguado	30	F	Bogota	Colombia

Query executed successfully.

- Ofertas

Results | Messages

	id_oferta	descripcion	importe
1	1	Fibra Optica 300MB + Fijo ilimitado + Television	70.00
2	2	Fibra Optica 600MB + Fijo ilimitado + Television	78.50
3	3	Fibra Optica 300MB + Movil llamadas y datos ilimi...	85.00
4	4	Fibra Optica 600MB + Movil llamadas y datos ilimi...	95.00
5	5	Fibra Optica 300MB + Fijo ilimitado + Movil llama...	102.50
6	6	Fibra Optica 600MB + Fijo ilimitado + Movil llama...	124.50
7	7	Fibra Optica 300MB + Movil llamadas ilimitadas +...	80.50
8	8	Fibra Optica 300MB + Movil datos ilimitados + Te...	84.00
9	9	Fijo llamadas ilimitadas + Movil llamadas y datos il...	42.50
10	10	Fijo llamadas ilimitadas + Movil llamadas ilimitadas	31.99

- Facturas mes anterior

Results | Messages

	id_cliente	id_oferta	importe	fecha
1	1	6	124.50	2020-07-31
2	1	6	118.60	2020-07-31
3	1	9	36.50	2020-07-31
4	2	4	95.00	2020-07-31
5	3	5	102.50	2020-07-31
6	5	7	80.50	2020-07-31
7	5	9	40.50	2020-07-31
8	6	8	84.00	2020-07-31
9	7	4	95.00	2020-07-31
10	8	5	102.50	2020-07-31
11	10	6	124.50	2020-07-31
12	11	3	80.00	2020-07-31
13	12	1	70.00	2020-07-31
14	13	4	95.00	2020-07-31

Query executed successfully.

- Facturas mes actual

Results | Messages

	id_cliente	id_oferta	importe	fecha
1	1	6	124.50	2020-08-31
2	1	6	118.60	2020-08-31
3	2	4	95.00	2020-08-31
4	3	5	102.50	2020-08-31
5	4	1	70.00	2020-08-31
6	5	7	80.50	2020-08-31
7	5	9	40.50	2020-08-31
8	6	8	84.00	2020-08-31
9	7	4	95.00	2020-08-31
10	8	5	100.50	2020-08-31
11	9	7	78.50	2020-08-31
12	9	10	29.99	2020-08-31
13	10	6	124.50	2020-08-31
14	11	3	83.00	2020-08-31

Query executed successfully.

- Consumos diarios

Results | Messages

	id_cliente	consumo_datos_MB	sms_enviados	minutos_llamadas_movil	minutos_llamadas_fijo	fecha
1	1	664	3	25	0	2020-08-01
2	1	943	0	12	5	2020-08-02
3	1	1035	1	46	10	2020-08-03
4	1	760	3	17	0	2020-08-04
5	1	1409	1	31	4	2020-08-05
6	1	1627	0	20	0	2020-08-06
7	1	739	3	21	0	2020-08-07
8	1	760	0	12	0	2020-08-08
9	1	928	0	0	24	2020-08-09
10	1	829	2	46	0	2020-08-10
11	1	1810	3	0	0	2020-08-11
12	1	760	0	25	0	2020-08-12
13	1	603	0	41	0	2020-08-13
14	1	732	0	34	102	2020-08-14

Query executed successfully.

Consultas

Consulta 1

Mostrar por pantalla el número TOTAL de clientes que tenían en el mes anterior más de un contrato con la compañía

```sql
with recuento as(
    select
        id_cliente,
        recuento=count(*)
    from facturasmesactual
    group by id_cliente)

select count(*) as NumeroClientes
from recuento
where recuento.recuento > 1
```

Results | Messages

	NumeroClientes
1	8

Consulta 2

Generar una nueva tabla de facturas del mes actual que asigne un 7% de descuento a todos los contratos de clientes que ya existían en el mes anterior (los contratos de los clientes nuevos seguirán con el mismo importe). Mantener la columna "importe" y crear una nueva columna "importe_dto" con el nuevo importe, casteada a 2 decimales. Mostrar además el resultado ordenado por los campos id_cliente ascendente, importe descendente.

```sql
with ClientesAnt as(
    select distinct id_cliente
    from facturasmesanterior)

select
    facturasmesactual.id_cliente,
    facturasmesactual.id_oferta,
    facturasmesactual.importe,
    FacturasMesActual.fecha,
    importe_dto = case
                    when clientesant.id_cliente is not null then importe*0.93
                    else importe
                end

from facturasmesactual
    left join ClientesAnt
        on facturasmesactual.id_cliente = ClientesAnt.id_cliente
```

Results | Messages

	id_cliente	id_oferta	importe	fecha	importe_dto
1	1	6	124.50	2020-08-31	115.7850
2	1	6	118.60	2020-08-31	110.2980
3	2	4	95.00	2020-08-31	88.3500
4	3	5	102.50	2020-08-31	95.3250
5	4	1	70.00	2020-08-31	70.0000
6	5	7	80.50	2020-08-31	74.8650
7	5	9	40.50	2020-08-31	37.6650
8	6	8	84.00	2020-08-31	78.1200
9	7	4	95.00	2020-08-31	88.3500
10	8	5	100.50	2020-08-31	93.4650
11	9	7	78.50	2020-08-31	78.5000
12	9	10	29.99	2020-08-31	29.9900
13	10	6	124.50	2020-08-31	115.7850
14	11	3	83.00	2020-08-31	77.1900
15	12	1	70.00	2020-08-31	65.1000

Consulta 3

Por problemas de saturación en la red debido al incremento en el uso de datos por el confinamiento debido a la COVID-19, se decide limitar el uso de datos este mes subiendo la tarifa de todas las ofertas que incluyan "datos ilimitados" en un 15%. Obtener una tabla con las mismas columnas que el DF de facturas del mes actual y a mayores la columna "importe_dto".

```sql
with Ilimitados as(
    select id_oferta
    from Ofertas
    where descripcion like '%datos ilimitados%')

select
    facturasmesactual.id_cliente,
    facturasmesactual.id_oferta,
    facturasmesactual.importe,
    FacturasMesActual.fecha,
    importe_dto = case
                    when Ilimitados.id_oferta is not null then convert(decimal(10,2),FacturasMesActual.importe*1.15,2)
                    else FacturasMesActual.importe
                end
from FacturasMesActual
    left join Ilimitados
        on FacturasMesActual.id_oferta=Ilimitados.id_oferta
```

Results | Messages

	id_cliente	id_oferta	importe	fecha	importe_dto
1	1	6	124.50	2020-08-31	143.18
2	1	6	118.60	2020-08-31	136.39
3	2	4	95.00	2020-08-31	109.25
4	3	5	102.50	2020-08-31	117.88
5	4	1	70.00	2020-08-31	70.00
6	5	7	80.50	2020-08-31	80.50
7	5	9	40.50	2020-08-31	46.58
8	6	8	84.00	2020-08-31	96.60
9	7	4	95.00	2020-08-31	109.25
10	8	5	100.50	2020-08-31	115.58
11	9	7	78.50	2020-08-31	78.50
12	9	10	29.99	2020-08-31	29.99
13	10	6	124.50	2020-08-31	143.18
14	11	3	83.00	2020-08-31	95.45

Consulta 4

Crear una nueva variable "grupo_edad" que agrupe a los clientes, tanto del mes_actual como del mes_anterior en 4 rangos según su edad asignando valores del 1 - 4 segun el rango al que pertenezcan (18-25 (1), 26-40 (2), 41-65 (3), >65 (4)). Obtener una tabla resumen que extraiga para cada para cada uno de los 4 grupos identificados la MEDIA de consumo de datos, sms enviados, minutos_movil, minutos_fijo, con todos los campos casteados a 2 decimales y ordenar el DF por grupo_edad ascendente. Extraer conclusiones.

```sql
with GruposEdad as (
    select
        id_cliente,
        grupo_edad =
                    case
                        when edad between 18 and 25 then 1
                        when edad between 26 and 40 then 2
                        when edad between 41 and 65 then 3
                        else 4
                    end
    from Clientes
    ),

    ConsumosTotales as (
    select
        grupo_edad
        ,CD.id_cliente
        , Media_consumo_datos_MB = sum([consumo_datos_MB])
        , Media_sms_enviados = sum([sms_enviados])
        , Media_minutos_llamadas_movil = sum([minutos_llamadas_movil])
        , Media_minutos_llamadas_fijo = sum([minutos_llamadas_fijo])
    from ConsumosDiarios CD
        left join GruposEdad GE
            on GE.id_cliente=CD.id_cliente
    group by grupo_edad, CD.id_cliente
    )

select
    grupo_edad
    , Media_consumo_datos_MB = avg(Media_consumo_datos_MB)
    , Media_sms_enviados = avg(Media_sms_enviados)
    , Media_minutos_llamadas_movil = avg(Media_minutos_llamadas_movil)
    , Media_minutos_llamadas_fijo = avg(Media_minutos_llamadas_fijo)
from ConsumosTotales
group by grupo_edad
```

Results | Messages

	grupo_edad	Media_consumo_datos_MB	Media_sms_enviados	Media_minutos_llamadas_movil	Media_minutos_llamadas_fijo
1	1	41485	16	588	336
2	2	33569	35	1081	674
3	3	14672	162	1005	819
4	4	5307	191	560	1108

Consulta 5

Se quiere realizar un estudio por Sexo para analizar si son las mujeres o los hombres quiénes consumen más datos durante el fin de semana y hacen más llamadas desde el móvil. Para ello se deberá, sin ayuda de un calendario, extraer el día de la semana al que corresponde cada una de las fechas del mes de Agosto para saber cuáles son fin de semana (se consideran días de fin de semana el viernes, sábado y domingo). </br> El DF a obtener deberá tener 2 registros con las columnas: sexo, total_mins_movil_finde, total_datos_moviles_finde. </br> Extraer conclusiones tras presentar el DF resultante.

```
select
    sexo
    ,total_mins_movil_finde = convert(decimal(10,2),sum(minutos_llamadas_movil)*1.0/count(*)*3)
    ,total_datos_moviles_finde = convert(decimal(10,2),sum(consumo_datos_mb)*1.0/count(*)*3)
from ConsumosDiarios CD
    join (select id_cliente, sexo from Clientes) SX
        on CD.id_cliente=SX.id_cliente
where datepart(weekday, convert(date,fecha)) >= 5
group by SX.sexo
```

Results | Messages

	sexo	total_mins_movil_finde	total_datos_moviles_finde
1	F	119.04	2439.71
2	M	52.94	2131.64

Consulta 6

Obtener un DF que contenga 4 registros, que serán el cliente de cada grupo edad que más datos móviles ha consumido durante los 15 primeros días del mes de Agosto (día 15 incluido en el cálculo). El DF deberá contener las columnas: nombre, edad, grupo_edad, datos_moviles_total_15, max_sms_enviados_15 </br> (max_sms_enviados_15 contiene el máximo de sms enviados en un día por el cliente con más datos_moviles consumidos de cada grupo_edad durante esos 15 primeros días del mes). </br> Extraer conclusiones en cuanto a datos consumidos y sms enviados por cada grupo_edad.

```sql
with GruposEdad as (
    select
        id_cliente,
        grupo_edad =
                    case
                        when edad between 18 and 25 then 1
                        when edad between 26 and 40 then 2
                        when edad between 41 and 65 then 3
                        else 4
                    end
    from Clientes
    )

, Agregados as(
    select
        grupo_edad
        ,CD.id_cliente
        ,datos_totales_15 = sum(consumo_datos_mb)
        ,sms_max_15 = max(sms_enviados)
    from ConsumosDiarios CD
        join GruposEdad GE
            on CD.id_cliente=GE.id_cliente
    where day(convert(date,fecha)) <= 15
    group by grupo_edad, CD.id_cliente)

, Rankings as(
    select
        *
        , ranking_datos = rank() over(partition by grupo_edad order by datos_totales_15 desc)
    from Agregados)

select
    nombre
    ,edad
    ,grupo_edad
    ,datos_totales_15
    ,sms_max_15
from Clientes
    join Rankings
        on Clientes.id_cliente=Rankings.id_cliente
where Rankings.ranking_datos=1
order by grupo_edad
```

Results | Messages

	nombre	edad	grupo_edad	datos_totales_15	sms_max_15
1	Clara Suarez	21	1	27665	2
2	Romina Verde	29	2	23603	1
3	Walter Ramos	54	3	11892	18
4	Marta Rodrigo	67	4	4657	16

Consulta 7

Nos interesa averiguar los minutos de llamadas de los clientes que son nuevos este mes para realizar un estudio del impacto que tendrán en caso de producirse un pico de volumen de llamadas en la red. </br> Obtener un DF que contenga sólo a los clientes nuevos de este mes con 4 columnas: nombre_cliente_nuevo, edad, importe_total_mes_actual, total_minutos. </br> Se ordenará el DF resultante por la columna "total_minutos" que contiene el total de minutos de llamadas fijas y móviles durante este mes para cada cliente, ordenándolos de manera que el primer cliente nuevo que aparezca sea el que menos minutos de llamadas ha realizado en el mes de Agosto.</br> NOTAS: Si hay solo 2 clientes nuevos, el DF a obtener tendrá sólo 2 registros. </br> Tener presente que a la hora de hacer un join, la clave de unión debe ser única en la tabla de la derecha o de otra manera se multiplicarán los registros resultantes de manera descontrolada. </br> Tener en cuenta que si un cliente nuevo tiene 2 ofertas contratadas, la columna importe_mes_actual deberá contener la suma de los importes de ambas ofertas.

```sql
with NuevosClientes as(
    select distinct act.id_cliente from FacturasMesActual ACT
        left join (select distinct id_cliente from FacturasMesAnterior) ANT
            on act.id_cliente=ant.id_cliente
    where ant.id_cliente is null)
    ,
    DatosClientes as(
    select id_cliente, nombre, edad
    from Clientes)
    ,
    Importes as(
    select id_cliente, mensual=sum(importe)
    from FacturasMesActual
    group by id_cliente)


select
    nombre
    ,edad
    ,importe_total_mes = mensual
    ,total_minutos = sum(minutos_llamadas_movil)+sum(minutos_llamadas_fijo)
from ConsumosDiarios
    inner join NuevosClientes
        on ConsumosDiarios.id_cliente = NuevosClientes.id_cliente
    inner join DatosClientes
        on ConsumosDiarios.id_cliente = DatosClientes.id_cliente
    inner join Importes
        on ConsumosDiarios.id_cliente = Importes.id_cliente
group by nombre, edad, mensual
order by total_minutos
```

Results | Messages

	nombre	edad	importe_total_mes	total_minutos
1	David Cardoso	24	108.49	368
2	Walter Ramos	54	102.50	528
3	Juan Carlos Iglesias	38	85.00	923
4	Pedro Conde	32	70.00	1939
5	Fatima Cuevas	29	124.50	2366

Consulta 8

Obtener un DF que contenga, para los clientes que ya existían en el mes anterior y siguen dados de alta este mes, 3 columnas: nombre, edad, n_dias_sin_sms. La última columna se refiere a obtener el número de veces que cada cliente NO ha enviado ningún SMS en el mes de Agosto. Forzar a que el comando "show" muestre 30 valores. </br> (Si en todo el mes el cliente "A" no envío SMS 3 días en todo el mes, esta columna deberá contener el valor 3. Si hubiera algún cliente que hubiese enviado al menos 1 sms todos los dias del mes, tendrá valor 0 en esta columna).

```sql
with TablaSms as(
    select id_cliente, dias_sin_sms=count(*)
    from ConsumosDiarios
    where sms_enviados=0
    group by id_cliente
    ),
    ClientesJulioAgosto as(
    select distinct FacturasMesActual.id_cliente
    from FacturasMesActual
        join FacturasMesAnterior
            on FacturasMesActual.id_cliente=FacturasMesAnterior.id_cliente
    )

select
    nombre
    ,edad
    ,dias_sin_sms
from Clientes
    inner join ClientesJulioAgosto
        on Clientes.id_cliente=ClientesJulioAgosto.id_cliente
    inner join TablaSms
        on Clientes.id_cliente=TablaSms.id_cliente
order by nombre
```

Results | Messages

	nombre	edad	dias_sin_sms
1	Alvaro Monroy	75	3
2	Ana Robles	41	3
3	Carmen Arauzo	19	19
4	Clara Suarez	21	26
5	Cristian Cuadrado	52	5
6	David Roldan	74	1
7	Eduardo Redondo	58	7
8	Fernanda Gomez	78	21
9	Ines Barcero	29	13
10	Jorge Recio	18	22
11	Laura Luiz	44	7
12	Luis Rodrigues	20	21
13	Marta Rodrigo	67	2
14	Melisa Aguado	30	20

Consulta 9

Queremos obtener un Coeficiente de Ponderación que nos permita evaluar a cada cliente en función de su consumo para identificar los clientes más atractivos que forman parte de nuestra compañía. ESTE CÁLCULO SÓLO SE REALIZARÁ PARA LOS CLIENTES QUE TENGAN UNA SOLA OFERTA CONTRATADA CON LA COMPAÑÍA. </br> Este coeficiente se obtendrá en base a los consumos diarios, y por tanto sólo se tendrán en cuenta los clientes que existen en el mes de Agosto, ya que no tenemos datos de consumo del mes de Julio. </br> Las ponderaciones que se darán a cada uno de los consumos son las siguientes:

0.4 --> llamadas desde telefóno móvil.

0.3 --> datos_móviles_MB

0.2 --> llamadas desde teléfono fijo

0.1 --> SMS enviados </br>

Los pasos a seguir son los siguientes:

Obtener la suma de todos los días de cada uno de los 4 consumos para cada cliente.

Obtener el máximo del cálculo anterior de todos los clientes para poder obtener un valor entre 0 y 1 para cada uno de los 4 consumos (recordar que sólo aplica para los clientes con UNA SOLA OFERTA CONTRATADA en el mes de Agosto. El cliente con mayor consumo de datos tendrá un valor de 1 en la columna 'datos_móviles_0_1')

Multiplicar estas columnas obtenidas con valor entre 0 y 1 por su ponderación correspondiente (por ejemplo la columna "datos_moviles_0_1" se multiplicará por la ponderación de datos móviles 0.4).

Por último se calculará la suma de las 4 ponderaciones obtenidas para calcular el Coeficiente de ponderación buscado (coeficiente_cliente), casteado a 3 decimales.

Mostrar DF resultante ordenado por este coeficiente en sentido descendente, de manera que el primer registro corresponderá al cliente más atractivo. El DF resultante tendrá 3 columnas: nombre, edad, coeficiente_cliente.

```sql
with GruposEdad as (
    select
        id_cliente,
        grupo_edad =
                    case
                        when edad between 18 and 25 then 1
                        when edad between 26 and 40 then 2
                        when edad between 41 and 65 then 3
                        else 4
                    end
    from Clientes)

, DatosTotales as (
    select grupo_edad, ConsumosDiarios.id_cliente, datos_totales = sum(consumo_datos_mb)
    from ConsumosDiarios
        inner join GruposEdad
            on ConsumosDiarios.id_cliente = GruposEdad.id_cliente
    group by grupo_edad, ConsumosDiarios.id_cliente)

, RankingDatos as (
    select *, ranking = rank() over(partition by grupo_edad order by datos_totales desc)
    from DatosTotales)

, Top3Consumos as (
    select grupo_edad, id_cliente, datos_totales
    from RankingDatos
    where ranking <= 3)

, DatosGrupoMes as (
    select grupo_edad, datos_grupo_mes = sum(datos_totales)
    from Top3Consumos
    group by grupo_edad)

, ConsumosAcumulados as(
    select distinct grupo_edad, fecha, consumo_acumulado = sum(consumo_datos_MB) over(partition by grupo_edad order by fecha)
    from ConsumosDiarios
        join Top3Consumos
            on ConsumosDiarios.id_cliente = Top3Consumos.id_cliente)
```

Results | Messages

	nombre	edad	coeficiente_cliente
1	Romina Verde	29	0.740
2	Fatima Cuevas	29	0.687
3	Silvia Rodriguez	49	0.653
4	Celia Castro	47	0.595
5	Melisa Aguado	30	0.552
6	Pedro Conde	32	0.517
7	Carmen Arauzo	19	0.486
8	Eduardo Redondo	58	0.440
9	Fernanda Gomez	78	0.410
10	Juan Carlos Iglesias	38	0.385
11	Roberta Varado	64	0.383
12	David Roldan	74	0.375
13	Jorge Recio	18	0.356
14	Walter Ramos	54	0.339

Consulta 10

Queremos averiguar la fecha en la que entre los 3 clientes que más datos consumen de cada grupo_edad llegan a un consumo de 20 GB de datos móviles. En caso de que algún grupo_edad no llegue entre los 3 clientes a 20 GB en todo el mes, se asignará "null" como valor de esta columna (Recordar que 1 GB = 1024 MB). </br> Obtener un DF que contiene 4 registros, uno para cada grupo_edad y 3 columnas: grupo_edad, fecha_20_GB, datos_moviles_total_grupo_3_clientes. (datos_moviles_total_grupo_3_clientes representa el total de datos consumidos en MB por los 3 clientes del grupo hasta final de mes)

```sql
with GruposEdad as (
    select
        id_cliente,
        grupo_edad =
                    case
                        when edad between 18 and 25 then 1
                        when edad between 26 and 40 then 2
                        when edad between 41 and 65 then 3
                        else 4
                    end
    from Clientes)

, DatosTotales as (
    select grupo_edad, ConsumosDiarios.id_cliente, datos_totales = sum(consumo_datos_mb)
    from ConsumosDiarios
        inner join GruposEdad
            on ConsumosDiarios.id_cliente = GruposEdad.id_cliente
    group by grupo_edad, ConsumosDiarios.id_cliente)

, RankingDatos as (
    select *, ranking = rank() over(partition by grupo_edad order by datos_totales desc)
    from DatosTotales)

, Top3Consumos as (
    select grupo_edad, id_cliente, datos_totales
    from RankingDatos
    where ranking <= 3)

, DatosGrupoMes as (
    select grupo_edad, datos_grupo_mes = sum(datos_totales)
    from Top3Consumos
    group by grupo_edad)

, ConsumosAcumulados as(
    select distinct grupo_edad, fecha, consumo_acumulado = sum(consumo_datos_MB) over(partition by grupo_edad order by fecha)
    from ConsumosDiarios
        join Top3Consumos
            on ConsumosDiarios.id_cliente = Top3Consumos.id_cliente)
```

```sql
, ConsumosFiltrados as (
    select *, sobrepasa_20GB=rank() over(partition by grupo_edad order by consumo_acumulado)
    from ConsumosAcumulados
    where consumo_acumulado > 20000)

, Fechas_20GB as (
    select *
    from ConsumosFiltrados
    where sobrepasa_20GB = 1)

select DatosGrupoMes.grupo_edad, datos_grupo_mes, fecha
from DatosGrupoMes
    left join Fechas_20GB
        on DatosGrupoMes.grupo_edad=Fechas_20GB.grupo_edad
```

Results | Messages

	grupo_edad	datos_grupo_mes	fecha
1	1	139687	2020-08-04
2	2	136137	2020-08-05
3	3	63100	2020-08-10
4	4	18995	NULL