Capacitación SQL Nivel inicial

Ejercicios Resueltos

Índice general

Unidad 1:

- Importar Base de datos
- Tablas y registros

Unidad 2:

- Consultas básicas SQL
- Sintaxis de Select
- Operadores aritméticos, lógicos y de comparación

Unidad 3:

- Tipos de Joins y alternativas de sintaxis

Unidad 4:

- Comandos para la Gestión de Base de datos:
  - Create
  - Insert
  - Delete
  - Update

Unidad 5:

- Funciones integradas
  - Funciones para cadenas de caracteres
  - Funciones datetime
- Agregación (group by - having)
- Funciones de Ranking
- Detección de Duplicados
- Conversión de Tipo de Datos

Unidad 6:

- Subconsultas
- Combinación de consultas SQL
- Union

Unidad 7:

- Generación de vistas

Unidad 8:

- Procedimientos Almacenados

Unidad 1:

Importar base de datos

1. Pasos para importar una la base de datos descargada en el manual de instalación
    a. Abrir Sql Server Managment studio y conectar a la instancia instalada
    b. Clic derecho en la carpeta Databases y elegir la opción *Restore Database*
    c. Marcar la opción *Device* y darle clic al botón **...**
    d. Dar clic en el botón *Add* y buscar la ubicación del archivo descargado 'AdventureWorksDW2017.bak' (Si no se lo encuentra en el explorador de archivos, mover a manualmente a la ruta C:\Program Files\Microsoft SQL Server\MSSQL15.SQLEXPRESS\MSSQL\Backup)
    e. Dar Ok y restaurar

Unidad 2:

Sintaxis de Select

1. Continuando con el pedido anterior, el equipo de Recursos Humanos, necesita saber cuáles de los empleados que están en estado activo. Para esto hay que listar todos los registros y todos los campos la tabla **DimEmployee** donde el campo **Status** sea igual 'Current'

    SELECT * FROM [dbo].[DimEmployee] WHERE Status = 'Current'

2. De la misma lista de empleados que solicita el equipo de Recursos Humanos, deberán figurar los empleados que tengan menor antigüedad, por lo tanto se solicita como parámetro traer los registros la tabla **DimEmployee** donde el campo EmployeeKey sea mayor a 10

    SELECT * FROM [dbo].[DimEmployee] WHERE EmployeeKey > 10

Operadores aritméticos, lógicos y de comparación

1. Continuando con el pedido solicitado por el equipo de Recursos Humanos, para poder realizar un subgrupo por edad de empleados en estado activo. Para esto listar todos los registros y todos los campos de la tabla **DimEmployee** donde el campo **Status** sea igual 'Current', el campo **EmployeeKey** sea mayor a 10 y que la fecha de nacimiento de esos empleados sea entre '1987-01-01' y '1990-01-01'

   SELECT * FROM [dbo].[DimEmployee] WHERE EmployeeKey > 10 and Status = 'Current' and BirthDate BETWEEN '1987-01-01' AND '1990-01-01'

2. El equipo de Recursos Humanos necesita realizar un listado de empleados para un grupo de mails y luego enviar comunicados a los mismos, donde estén todos los empleados menos 'Alejandro', 'Simon' y 'Fred'. Para esto hay que listar todos los registros y todos los campos de la tabla **DimEmployee** donde el campo **FirstName** no tenga los textos 'Alejandro', 'Simon' y 'Fred' además se solicitó **ParentEmployeeKey** sea igual a 112

   SELECT * FROM [dbo].[DimEmployee] WHERE FirstName NOT IN ('Alejandro','Simon','Fred') AND ParentEmployeeKey = 112

Unidad 3:

Tipos de Joins y alternativas de sintaxis

1. El equipo de Administración y Finanzas necesita realizar un análisis de tasa de cambio, para esto es necesario desarrollar una consulta que devuelva todo los campos de las tablas **DimDate**, **FactCurrencyRate**, partiendo de la tabla DimDate utilizando **LEFT JOIN**, filtrando que el campo DateKey de la tabla DimDate sea igual a 20141231. (Fijarse que si no existen registros con esas características los campos de la tabla **FactCurrencyRate** van a venir nulos, elegir otra fecha para ver la diferencia).

   SELECT * FROM [dbo].[DimDate] DT
   LEFT JOIN
   [dbo].[FactCurrencyRate] CR
   ON CR.DateKey = DT.DateKey
   AND DT.DateKey = 20141231

2. Para un análisis de cierre mensual, el equipo de Administración y Finanzas en conjunto con el equipo de Marketing, necesitan realizar una consulta utilizando **RIGHT JOIN** para traerse la información de las dos tablas mencionadas en el ejercicio anterior pero esta vez sin aplicar ningún filtro (Observar como el resultado cambia)

```
SELECT * FROM [dbo].[DimDate] DT
RIGHT JOIN
[dbo].[FactCurrencyRate] CR
ON CR.DateKey = DT.DateKey
```

3. El equipo de Marketing necesita realizar una visualización donde se compara la lista de precios de los competidores vs la interna de la empresa, para esto se deberá realizar una consulta utilizando **FULL JOIN** para cruzar toda la información que contienen las tablas **DimProduct** y **DimProductSubcategory**, filtrar los resultados para que el campo **ListPrice** no sea nulo (observar el comportamiento de los datos con la consulta, full join, ¿Se puede aplicar el filtro?)

```
SELECT * FROM [dbo].[DimProduct] P
FULL JOIN
[dbo].[DimProductSubcategory] PC
ON P.ProductSubcategoryKey = PC.ProductSubcategoryKey
AND ListPrice IS NOT NULL
```

Unidad 4:

Create

1. El área de I+D para realizar una petición del área de Recursos Humanos, donde se listen las personas por nombre, apellido, edad y peso, deberá crear una tabla que se llame **DimPersona** y que contenga los siguientes campos
    a. Id (entero auto incremental)
    b. Nombre (tipo de dato texto)
    c. Apellido (tipo de dato texto)
    d. Edad (tipo de dato numérico entero)
    e. Peso (tipo de dato decimal)

(Definir el tamaño de los campos para posteriormente insertar datos, el campo peso puede puede ser nulo el resto no)

```
CREATE TABLE [dbo].[DimPersona](
    [Id] [int] IDENTITY(1,1) NOT NULL,
    [Nombre]  [nvarchar](255) NOT NULL,
    [Apellido] [nvarchar](255) NOT NULL,
    [Edad] [int] NOT NULL,
    [Peso] [decimal](18,2)  NULL
    ) ON [PRIMARY]
```

2. El equipo de I+D tiene que crear dos tablas de respaldo (backUp) de la tabla Dim_Account para realizar unas pruebas de carga de datos. Para esto, realizar una consulta SQL que cree dos tablas de respaldo con datos llamadas Dim_Account_Bkp1 y Dim_Account_Bkp2 a partir de la tabla Dim_Account
(Tienen que tener la misma estructura las dos tablas)

```
SELECT * INTO [dbo].[DimAccount_Bkp1] FROM [dbo].[DimAccount]
SELECT * INTO [dbo].[DimAccount_Bkp2] FROM [dbo].[DimAccount]
```

Insert

1. Se realizaron 5 nuevos ingresos de personas en el área Comercial, por lo que el equipo de Recursos Humanos tiene que hacer el ingreso de los mismos, en la tabla creada con anterioridad DimPersona. Para esto de debe armar una consulta que inserte 5 registros de distintas personas a la tabla **DimPersona** creada anteriormente (*Dificultad:* Tener en cuenta los campos que no deberían ir)

```
INSERT INTO DimPersona (Nombre, Apellido, Edad, Peso) VALUES ('Juan','Perez','22',70.5)
INSERT INTO DimPersona (Nombre, Apellido, Edad) VALUES ('Pedro','Perez','22')
INSERT INTO DimPersona (Nombre, Apellido, Edad, Peso) VALUES ('Paul,'Bustos','29',80.5)
INSERT INTO DimPersona (Nombre, Apellido, Edad, Peso) VALUES ('Raul','Jerez','26',60)
INSERT INTO DimPersona (Nombre, Apellido, Edad, Peso) VALUES ('Pablo','Gomez','19')
```

Delete

1. Luego de realizar las pruebas de carga de datos, el área de I+D ya no necesita utilizar la **DimAccount_Bkp2,** por lo que solicita armar una consulta SQL que borre todos los registros de la tabla **DimAccount_Bkp2**

   DELETE FROM DimAccount_Bkp2

Update

1. Para un análisis del equipo Comercial, se solicita realizar una actualización de datos de la tabla Bkp1, creada por el área de I+D para que en el campo **CustomMemberOptions** se actualice con el valor 'N/A. Para esto, armar una consulta SQL que actualice el campo **CustomMemberOptions** de la tabla **DimAccount_Bkp1** con el valor 'N/A' para todos los registros de la misma.

   UPDATE DimAccount_Bkp1 SET CustomMemberOptions = 'N/A'

2. El equipo de Comercial nuevamente necesita una actualización de datos en la tabla armada con anterioridad de Bkp1, en el cual se actualice el campo **CustomMemberOptions** con el valor 'Cuenta' y el campo **CustomMembers** con el valor 'Usuario' de la tabla **DimAccount_Bkp1** con la condición de que el campo ParentAccountKey tiene que ser igual a 17. Realice una consulta de SQL para lograr dicha actualización.
   UPDATE DimAccount_Bkp1 SET CustomMemberOptions = 'Cuenta', CustomMembers = 'Usuario' WHERE ParentAccountKey = 17

Unidad 5:

Funciones para cadenas de caracteres

1. El equipo de Recursos humanos necesita crear un nuevo campo desde la tabla **DimEmployee** que se llame Nombre_Apellido que contenga el nombre y apellido de los empleados y un espacio en el medio entre el nombre y apellido, el mismo debe contener todos los valores en mayúsculas para luego utilizar esa información en un tablero de

presentismo y vacaciones, listar además el campo **Title** y reemplazar la letra **a** por un **@,** listar todos los valores del campo **VacationHours** y cambiar el tipo de dato a decimal con un tamaño de (18,2), por último eliminar los espacios de la izquierda del campo

```
SELECT
UPPER(CONCAT(FirstName,' ' ,LastName)) AS Nombre_Apellido
,REPLACE(Title,'a','@') AS Title
,CAST(VacationHours AS  DECIMAL(18,2)) AS VacationHours
,LTRIM(EmergencyContactName) AS EmergencyContactName
FROM [dbo].[DimEmployee]
```

Funciones datetime

1. El equipo de Recursos Humanos necesita realizar además del tablero de presentismo y vacaciones, un tablero en el que se visualice el tiempo de permanencia de cada empleado dentro de la empresa. Continuando con el pedido anterior, crear desde la tabla **DimEmployee** un campo llamado Nombre y Apellido, que contenga el nombre y apellido de los empleados, el mismo debe contener todos los valores en minúsculas, listar el campo **HireDate** y renombrarlo a 'Fecha_Ingreso', crear los campos Año de Ingreso y Mes de Ingreso a partir del campo **HireDate**. Calcular el primer aniversario del empleado y traerlo en un campo, calcular además la cantidad de días siendo empleado de la empresa.

```
SELECT
LOWER(CONCAT(FirstName,' ' ,LastName)) AS Nombre_Apellido
,HireDate AS Fecha_Ingreso
,DATEPART(YEAR,HireDate) AS Anio
,DATEPART(MONTH,HireDate) AS Mes
,DATEADD(YEAR, 1, HireDate) AS Primer_Aniversario
,DATEDIFF(DAY,HireDate,GETDATE()) AS Antiguedad
FROM [dbo].[DimEmployee]
```

Agregación (group by - having)

1. El equipo de Comercial en conjunto con el equipo de Administración y Finanzas, necesitan armar un reporte de productos por ventas para evaluar el evolutivo de las mismas, que muestre las ventas de los productos por nombre y por fecha de orden con la condición que tienen que ser productos del mes de Enero de 2011 y que la cantidad de productos vendidos sea menor a 300. Para esto se utilizan las tablas **DimProducto**, **FactInternetSales**, y los campos **EnglishProductName**, **ProductKey, OrderDateKey**, **TotalProductCost**, **OrderDateKey**

```
select
EnglishProductName,
OrderDateKey,
TotalProductCost,
count(*) as Cantidad
from [dbo].[FactInternetSales] FC
JOIN
[dbo].[DimProduct] PD
ON FC.ProductKey = PD.ProductKey
WHERE  OrderDateKey/100 = 201101
group by EnglishProductName,OrderDateKey,TotalProductCost
HAVING count(*) < 300
```

Las funciones SQL MIN () y MAX ()

1. El equipo de Comercial necesita visualizar la venta con el menor valor, para la toma de decisiones frente al cierre del periodo. Para esto se necesita desarrollar una consulta de SQL que muestre los campos SalesOrderNumber y UnitePrice con la condicion que traiga el registro con el menor valor de una venta de la tabla **FactInternetSales**

```
SELECT SalesOrderNumber, min(UnitPrice) PrecioUnitario FROM [FactInternetSales]
group by SalesOrderNumber
```

2. A su vez, el equipo de Comercial solicita visualizar el mismo pedido de ventas pero esta vez visualizando el máximo valor en otro periodo determinado. Para esto traer la misma consulta con el máximo valor solo para el mes de Febrero de 2011.

```
SELECT SalesOrderNumber, max(UnitPrice) PrecioUnitario FROM [FactInternetSales]
```

```
WHERE  OrderDateKey/100 = 201102
group by SalesOrderNumber
```

Las funciones SQL COUNT (), AVG () y SUM ()

1. De cara al cierre mensual, el equipo de Administración y Finanzas necesita realizar un reporte de balance, para el cual solicitaron los montos y cantidad total de ventas del mes de enero del 2011. Para esto se deberá armar una consulta SQL que calcule el monto y cantidad total de las ventas de los productos por nombre y por fecha de orden con la condición que tienen que ser productos del mes de Enero de 2011 y que la cantidad de productos vendidos sea menor a 300, se deberán utilizar las tablas **DimProducto**, **FactInternetSales**, y los campos **EnglishProductName**, **ProductKey, OrderDateKey**, **TotalProductCost**, **OrderDateKey**

```
SELECT
EnglishProductName,
OrderDateKey,
sum(TotalProductCost) as Total,
count(*) as Cantidad
from [dbo].[FactInternetSales] FC
JOIN
[dbo].[DimProduct] PD
ON FC.ProductKey = PD.ProductKey
WHERE  OrderDateKey/100 = 201101
group by EnglishProductName,OrderDateKey
HAVING count(*) < 300
```

2. El equipo de Administración para poder presentar el balance del primer trimestre del año 2012, necesita visualizar el monto promedio de ventas por productos del mes de marzo 2012. Para esto se deberá realizar una consulta SQL que devuelva el monto promedio de ventas por producto en el mes de marzo de 2012 en donde se va a utilizar la tabla **FactInternetSales**

```
SELECT SalesOrderNumber, avg(UnitPrice) PrecioUnitario FROM
[FactInternetSales]
WHERE  OrderDateKey/100 = 201203
group by SalesOrderNumber
```

Operadores Aritméticos

1. Como todos los meses, el equipo de Control y Armado de Stock solicita calcular la cantidad de productos con el código 310 y rendir el monto invertido del lote de productos al equipo de comercial, para que luego este pueda calcular la ganancia del mismo. Para esto se debe armar una consulta SQL que calcule la cantidad de todos los productos con el código 310 y esa cantidad la multiplique por el precio unitario se utilizará la tabla **FactInternetSales**

   SELECT COUNT(ProductKey) * SUM(UnitPrice) AS MontoTotal FROM [FactInternetSales]

Funciones de RANK

1. El equipo Comercial solicitó un análisis del monto total de ventas de los productos por fecha de orden para el mes de Enero de 2011. Para esto se deberá armar una consulta SQL que calcule el monto total de las ventas de los productos por nombre y por fecha de orden, con la condición que tienen que ser productos del mes de Enero de 2011, además en el caso que se repita el producto para distintas fechas del mes, ordenar con la función Rank los productos con mayor precio, se utilizará la tabla **FactInternetSales, DimProducto**

   ```
   SELECT
   EnglishProductName,
   OrderDateKey,
   sum(TotalProductCost) as Total,
   RANK () over(partition by EnglishProductName order by OrderDateKey
   desc) AS Ranking
   from [dbo].[FactInternetSales] FC
   JOIN
   [dbo].[DimProduct] PD
   ON FC.ProductKey = PD.ProductKey
   WHERE  OrderDateKey/100 = 201101
   group by EnglishProductName,OrderDateKey
   ```

2. Para el mismo pedido del equipo Comercial, se solicita armar una consulta SQL que calcule el monto total de las ventas de los productos por nombre y por fecha de orden, con la misma condición que tienen

que ser productos del mes de Enero de 2011, además en el caso que se repita el producto para distintas fechas del mes de enero ordenar con la función Row_Number los productos con menor precio, se utilizará la tabla **FactInternetSales, DimProducto**

```
SELECT
EnglishProductName,
OrderDateKey,
sum(TotalProductCost) as Total,
RANK () over(partition by EnglishProductName order by OrderDateKey
desc) AS Ranking
from [dbo].[FactInternetSales] FC
JOIN
[dbo].[DimProduct] PD
ON FC.ProductKey = PD.ProductKey
WHERE  OrderDateKey/100 = 201101
group by EnglishProductName,OrderDateKey
```

Conversión de Tipo de Datos

1. El equipo de Control y Armado de Stock solicita cambiar los valores del campo estado de cada producto para poder luego plasmar en un tablero los productos agrupados por status. Para esto, realizar una consulta SQL que liste la columna **Status** de la tabla **DimProduct** y si encuentra registros vacíos en ese campo los reemplace por la palabra 'Sin Status', se utilizará la tabla **DimProducto**

```
SELECT ISNULL([Status],'Sin Status') as Status  from [dbo].[DimProduct]
```

Unidad 6:

Subconsultas

1. Dentro de la petición del equipo de Marketing, se necesita en este caso, el resultado de las filas que no coinciden entre las tablas de **DimCustomer** y **ProspectiveBuyers,** por lo que se necesita usar la sentencia NOT EXISTS (funciona como lo contrario que EXISTS). La cláusula WHERE en NOT EXISTS se cumple si la subconsulta no devuelve filas. Para esto, crear una consulta SQL para saber si se encuentra filas en la tabla **DimCustomer** donde **LastName** y **BirthDate** no coinciden con ninguna entrada en la tabla **ProspectiveBuyers**.

```
SELECT a.LastName, a.BirthDate
FROM DimCustomer AS a
WHERE NOT EXISTS
(SELECT *
   FROM dbo.ProspectiveBuyer AS b
         WHERE (a.LastName = b.LastName) AND (a.BirthDate = b.BirthDate));
```

Combinación de consultas SQL

Union

1. El equipo de Atención al Cliente, necesita realizar un análisis de los compradores de ventas por internet. Para esto se deberá realizar una consulta SQL que en su conjunto de resultados incluya el contenido de las columnas **CustomerKey** de las tablas **FactInternetSales** y **DimCustomer**. Como la palabra clave ALL no se utiliza, los duplicados se excluyen de los resultados.

```
SELECT CustomerKey
FROM FactInternetSales
UNION
SELECT CustomerKey
FROM DimCustomer
ORDER BY CustomerKey;
```

2. Continuando con el pedido del equipo de Atención al Cliente y poder finalizar su reporte de análisis, se solicita, usando UNION de dos sentencias SELECT con ORDER BY, traer los datos de las columnas **CustomerKey** de las tablas **FactInternetSales** y **DimCustomer**

```
SELECT CustomerKey
FROM FactInternetSales
UNION
SELECT CustomerKey
FROM DimCustomer
ORDER BY CustomerKey;
```

Unidad 7:

Generación de vistas

1. El equipo de Administración y Finanzas en conjunto con el equipo de Marketing, en vista a que cada mes se necesita armar reportes con la cantidad de ventas y montos de cada una por producto, solicitan armar una vista que se llame ***Monto_Cantidad_Ventas_View*** cuyo objetivo es generar reportes mensuales. Para esto crear una consulta SQL que calcule el monto y cantidad total de las ventas de los productos por nombre y por fecha de orden con la condición que tienen que ser productos del mes en curso y que la cantidad de productos vendidos sea menor a 300, para esto se deben utilizar las tablas **DimProducto**, **FactInternetSales**, y los campos **EnglishProductName**, **ProductKey, OrderDateKey**, **TotalProductCost**, **OrderDateKey**

```
CREATE VIEW [dbo].[Monto_Cantidad_Ventas_View ]
AS

SELECT
EnglishProductName,
OrderDateKey,
sum(TotalProductCost) as Total,
count(*) as Cantidad
from [dbo].[FactInternetSales] FC
JOIN
[dbo].[DimProduct] PD
ON FC.ProductKey = PD.ProductKey
WHERE  OrderDateKey/100 = Month(GETDATE())
group by EnglishProductName,OrderDateKey
HAVING count(*) < 300
```

```
;

GO
```

Unidad 8:

Procedimientos Almacenados

1. Para poder crear los reportes mensuales para el área de Administración y Finanzas, se deberá crear un procedimiento almacenado que ejecute la instrucción SELECT de la vista generada con anterioridad.

```
CREATE PROCEDURE Reporte_Monto_Cantidad_Ventas
AS
BEGIN
   SELECT *
   FROM Monto_Cantidad_Ventas_View AS V
   ORDER BY OrderDateKey;
END
;
GO
```