ACTIVIDADES T-SQL

Orden de actividades:

Si la duración de la formación es de un mes, realizar todas las actividades del documento.

Si la formación es de una semana, realizar los ejercicios subrayados en amarillo en el índice.

Bosonit

Contenido

1. Queries Simples

Ejercicios Select

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorksLT2008.

1. Escriba una declaración SELECT sobre la tabla “SalesLT.customer” que nos proporcione el número de identificación del cliente, apellidos, nombres y nombres de compañías.

2. Escriba una declaración SELECT que nos proporcione el nombre, número de producto y color de cada producto de la tabla “SalesLT.product”

3. Escriba una instrucción SELECT que nos proporcione los números de ID de cliente y los números de ID de pedido de ventas de la tabla SalesLT.SalesOrderHeader

4. Responda esta pregunta: ¿Por qué debería especificar nombres de columna en lugar de un asterisco al escribir la lista SELECT? Da al menos dos razones.

Ejercicios de filtrado de datos.

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Escriba una consulta mediante una cláusula WHERE que muestre todos los empleados enumerados en la tabla HumanResources.Employee que tienen el título de trabajo “Research and Development Engineer”. Muestre el número de ID de entidad comercial, el ID de inicio de sesión y el título de cada uno.

2. Escriba una consulta mediante una cláusula WHERE que muestre todos los nombres de Person.Person con el nombre intermedio J. Muestre los nombres, el apellido y los nombres intermedios junto con los números de identificador.

3. Escriba una consulta que muestre todas las columnas de la tabla Production.ProductCostHistory de las filas que se modificaron el 17 de junio de 2003. Asegúrese de usar una de las características de SQL Server Management StudioSQL Server Management Studio para ayudarle a escribir esta consulta. En SQL Server Management StudioSQL Server Management Studio, expanda la base de datos AdventureWorks2008. Expanda Tablas. Haga clic con el botón derecho en la tabla Production.ProductCostHistory y elija "Seleccionar tabla como." Seleccione "Seleccionar en" y Nueva ventana del Editor de consultas. A continuación, escriba la cláusula WHERE.

4. Vuelva a escribir la consulta que escribió en la pregunta 1, cambiándola para que se muestren los empleados que no tienen el título “Research and Development Engineer”.

5. Escriba una consulta que muestre todas las filas de la tabla Person.Person donde se

modificaron las filas después del 29 de diciembre de 2000. Mostrar el número de ID de entidad comercial, las columnas de nombre y la fecha de modificación.

6. Vuelva a escribir la última consulta para que se muestren las filas que no se modificaron el 29 de diciembre de 2000.

7. Vuelva a escribir la consulta de la pregunta 5 para que muestre las filas modificadas durante diciembre de 2000.

8. Vuelva a escribir la consulta de la pregunta 5 para que muestre las filas modificadas durante diciembre de 2000.

9. Explicar por qué se debe usar una cláusula WHERE en muchas de las consultas de T-SQL.

Ejercicios de filtrado de datos con comodines.

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Escriba una consulta que muestre el identificador del producto y el nombre de cada producto de la tabla Production.Product con el nombre que comienza con “chain”.
2. Escriba una consulta como la de la pregunta 1 que muestra los productos con casco **“helmet”** en el nombre.
3. Cambie la última consulta para que se muestren los productos sin casco “**helmet**” en el nombre.
4. Escriba una consulta que muestre el número de ID de entidad de negocio, el nombre, el segundo nombre y el apellido de la tabla Person.Person solo para las filas que tienen E o B almacenadas en la columna de nombre intermedio.
5. Explicar la diferencia entre las dos consultas siguientes:

```
SELECT FirstName
FROM
Person.Person
WHERE LastName LIKE 'Ja%es';

SELECT FirstName
FROM
Person.Person
WHERE LastName LIKE 'Ja_es';
```

Filtrado con múltiples predicados

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Escriba una consulta que muestre el identificador de pedido, la fecha de pedido y el total de vencimiento de la tabla Sales.SalesOrderHeader. Recupere solo las filas en las que se realizó el pedido durante el mes de septiembre de 2001 y el total adeudado superó los 1.000 $.
2. Cambie la consulta en la pregunta 1 para que solo se recuperen las fechas del 1 al 3 de septiembre de 2001. Vea si puede encontrar tres maneras diferentes de escribir esta consulta.
3. Escriba una consulta que muestre los pedidos de ventas en los que el total adeudado supera los 1.000 USD. Recupere solo las filas donde el identificador de vendedor es 279 o el identificador de territorio es 6.
4. Cambie la consulta en la pregunta 3 para que se incluya el territorio 4.
5. Explicar cuándo tiene sentido utilizar el operador IN.

Trabajando con vacíos y nulos

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Escriba una consulta que muestre las columnas ProductID, Name y Color de las filas de la tabla Production.Product. Muestre solo las filas en las que no se ha asignado ningún color.
2. Escriba una consulta que muestre las columnas ProductID, Name y Color de las filas de la tabla Production.Product. Muestre solo las filas en las que el color no es azul. Aquí hay dos posibles soluciones.
3. Escriba una consulta que muestre ProductID, Name, Style, Size y Color de la tabla Production.Product. Incluya solo las filas en las que al menos una de las columnas Style, Size o Color contiene un valor.

Realización de una búsqueda de texto completo

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Escriba una consulta mediante la tabla Production.ProductReview. Utilice CONTAINS para buscar todas las filas que tienen la palabra calcetines “socks” en la columna Comments. Devuelve las columnas ProductID y Comments.

2. Escriba una consulta mediante la tabla Production.Document. Utilice CONTAINS para buscar todas las filas que tienen la palabra reflector en cualquier columna que esté indizada con Búsqueda de texto completo. Mostrar las columnas Title y FileName.

3. Cambie la consulta en la pregunta 2 para que las filas que contienen el asiento “seat” no se devuelvan en los resultados.

4. Responda a esta pregunta: Al buscar una columna VARBINARY(MAX) que contiene documentos de Word, se puede utilizar una búsqueda LIKE, pero el rendimiento será peor. ¿Verdadero o falso?

Ordenando los datos.

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Escriba una consulta que devuelva las columnas de ID y nombre de la entidad comercial de Person.Person. Ordene los resultados por LastName, FirstName y MiddleName.

2. Modifique la consulta escrita en la pregunta 1 para que los datos se devuelvan en el orden opuesto.

Pensando sobre rendimiento.

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

Asegúrese de activar la opción Incluir plan de ejecución real antes de comenzar. Escriba el código siguiente en la ventana de consulta y, a continuación, complete cada pregunta.

```
USE
AdventureWorks2008;
GO

--1
SELECT LastName
FROM
Person.Person
WHERE LastName = 'Smith';

--2
SELECT LastName
FROM
Person.Person
WHERE LastName LIKE 'Sm%';

--3
SELECT LastName
FROM
Person.Person
WHERE LastName LIKE '%mith';

--4
SELECT
ModifiedDate FROM
Person.Person
WHERE ModifiedDate BETWEEN '2000-01-01' and '2000-01-31';
```

1. Resalte y ejecute las consultas 1 y 2. Explicar por qué no hay diferencia en el rendimiento entre las dos consultas.

2. Resalte y ejecute las consultas 2 y 3. Determine qué consulta funciona mejor y explique por qué cree que ese es el caso.

3. Resalte y ejecute las consultas 3 y 4. Determine qué consulta funciona mejor y explique por qué cree que este es el caso.

2. Uso de funciones y expresiones

Escribir expresiones con operadores

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Escriba una consulta que se muestre en una sola columna el "AddressLine1, City y PostalCode" de la tabla Person.Address.

2. Escriba una consulta mediante la tabla Production.Product que muestra las columnas de ID de producto, color y nombre. Si la columna de color contiene un valor NULL, reemplace el color por Sin color.

3. Modifique la consulta escrita en la pregunta 2 para que la descripción del producto se muestre en el formato "name: Color" (Ej "Gorra : Roja"). Asegúrese de que todas las filas muestran un valor incluso si falta el valor Color (Remplazar por espacio en blanco).

4. Escriba una consulta con la tabla Production.Product que muestre una descripción con el formato "ProductID: Name". Sugerencia: Tendrá que usar una función para escribir esta consulta.

5. Explicar la diferencia entre las funciones ISNULL y COALESCE.

Uso de operadores matemáticos.

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Escriba una consulta mediante la tabla Sales.SpecialOffer. Muestre la diferencia entre las columnas MinQty y MaxQty junto con las columnas SpecialOfferID y Description.

2. Escriba una consulta mediante la tabla Sales.SpecialOffer. Multiplique la columna MinQty por la columna DiscountPct. Incluya las columnas SpecialOfferID y Description en los resultados.

3. Escriba una consulta mediante la tabla Sales.SpecialOffer que multiplica la columna MaxQty por la columna DiscountPCT. Si el valor MaxQty es null, sustitúylo por el valor 10. Incluya las columnas SpecialOfferID y Description en los resultados.

4. Describa la diferencia entre división y módulo.

Uso de funciones de cadena.

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Escriba una consulta que muestre los primeros 10 caracteres de la columna AddressLine1 en la tabla Person.Address.

+

2. Escriba una consulta que muestre los caracteres 10 a 15 de la columna AddressLine1 en la tabla Person.Address

3. Escriba una consulta que muestre el nombre y el apellido de la tabla Person.Person en mayúsculas.

4. Escriba una consulta que utilice la función SUBSTRING y la función CHARINDEX para mostrar los caracteres del número de producto que sigue al guion. Nota: también hay un segundo guion en muchas de las filas; ignorar el segundo guion para esta pregunta. Sugerencia: Intente escribir esta instrucción en dos pasos, el primero con la función CHARINDEX y el segundo agregando la función SUBSTRING.

Uso de funciones de fecha.

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Escriba una consulta que calcule el número de días entre la fecha en que se realizó un pedido y la fecha en que se envió mediante la tabla Sales.SalesOrderHeader. Incluya las columnas SalesOrderID, OrderDate y ShipDate.

2. Escriba una consulta que muestre solo la fecha, no la hora, para la fecha de pedido y la fecha de envío en la tabla Sales.SalesOrderHeader.

3. Escriba una consulta que agregue seis meses a cada fecha de pedido en la tabla Sales.SalesOrderHeader. Incluya las columnas SalesOrderID y OrderDate.

4. Escriba una consulta que muestre el año de cada fecha de pedido y el mes numérico de cada fecha de pedido en columnas separadas en los resultados. Incluir las columnas SalesOrderID y OrderDate

5. Cambie la consulta escrita en la pregunta 4 para mostrar el nombre del mes en su lugar.

Funciones matemáticas.

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Escriba una consulta mediante la tabla Sales.SalesOrderHeader que muestra el SubTotal redondeado a dos decimales. Incluya la columna SalesOrderID en los resultados.

2. Modifique la consulta en la pregunta 1 para que el SubTotal se redondee al dólar más cercano, pero todavía muestra dos ceros a la derecha de la posición decimal.

3. Escriba una consulta que calcule la raíz cuadrada del valor SalesOrderID de la tabla Sales.SalesOrderHeader.

4. Escriba una instrucción que genere un número aleatorio entre 1 y 10 cada vez que se ejecute.

Uso de funciones de sistema.

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Escriba una consulta mediante la tabla HumanResources.Employee para mostrar la columna BusinessEntityID. Incluya también una instrucción CASE que muestre "Even" cuando el valor BusinessEntityID es un número par o "Odd" cuando es impar. Sugerencia: Utilice el operador modulo '%'.

2. Escriba una consulta utilizando la tabla Sales.SalesOrderDetail para mostrar un valor ("Under 10" o "10–19" o "20–29" o "30–39" o "40 y más") en función del valor OrderQty mediante la función CASE. Incluya las columnas SalesOrderID y OrderQty en los resultados.

3. Con la tabla Person.Person, cree los nombres completos con las columnas Title, FirstName, MiddleName, LastName y Suffix. Compruebe la definición de tabla para ver qué columnas permiten valores NULL y utilice la función COALESCE en las columnas adecuadas.

4. Busque la función SERVERPROPERTY. Escriba una instrucción que muestre la edición, el nombre de instancia y el nombre del equipo mediante esta función.

Uso de funciones en WHERE y clausulas en el ORDER BY

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Escriba una consulta mediante la tabla Sales.SalesOrderHeader para mostrar los pedidos realizados durante 2001 mediante una función. Incluya las columnas SalesOrderID y OrderDate en los resultados.

2. Escriba una consulta mediante la tabla Sales.SalesOrderHeader que enumera las ventas en orden del mes en que se realizó el pedido y, a continuación, el año en que se realizó el pedido. Incluya las columnas SalesOrderID y OrderDate en los resultados.

3. Escriba una consulta que muestre las columnas PersonType y name de la tabla Person.Person. Ordene los resultados para que las filas con un PersonType de IN, SP o SC ordenen por LastName. Las otras filas deben ordenar por FirstName. Sugerencia: Utilice la función CASE.

Pensando sobre rendimiento.

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

Asegúrese de activar la opción Incluir plan de ejecución real antes de comenzar.

1. Escriba y ejecute el siguiente código. Vea los planes de ejecución una vez completada la ejecución de la consulta y explique si una consulta funciona mejor que la otra y por qué.

```
USE
AdventureWorks2008;
GO

--1
SELECT Name
FROM
Production.Product
WHERE Name LIKE 'B%';

--2
SELECT Name
FROM Production.Product
WHERE CHARINDEX('B',Name) = 1;
```

2. Escriba y ejecute el siguiente código. Vea los planes de ejecución una vez completada la ejecución de la consulta y explique si una consulta funciona mejor que la otra y por qué.

```
USE
AdventureWorks2008;
GO

--1
SELECT LastName
FROM
Person.Person
WHERE LastName LIKE '%i%';

--2
SELECT LastName
FROM
Person.Person
WHERE CHARINDEX('i',LastName) > 0;
```

3. Queries sobre múltiples tablas.

Uso de Inner Joins

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. La tabla HumanResources.Employee no contiene los nombres de los empleados. Unir esa tabla a la Person.Person usando la columna BusinessEntityID. Mostrar el título del trabajo, la fecha de nacimiento, el nombre y el apellido.

2. Los nombres de los clientes también aparecen en la tabla Person.Person. Unir la tabla Sales.Customer a la tabla Person.Person. La columna BusinessEntityID de la tabla Person.Person coincide con la columna PersonID de la tabla Sales.Customer. Mostrar las columnas CustomerID, StoreID, y TerritoryID junto con las columnas de nombre.

3. Extienda la consulta escrita en la pregunta 2 para incluir la tabla Sales.SalesOrderHeader. Mostrar la columna SalesOrderID junto con las columnas ya especificadas. La tabla Sales.SalesOrderHeader combina la tabla Sales.Customer por la columna CustomerID.

4. **Escriba una consulta que una la tabla Sales.SalesOrderHeader a la tabla Sales.SalesPerson. Unir usando la columna BusinessEntityID de la tabla Sales.SalesPerson a la columna SalesPersonID de la tabla Sales.SalesOrderHeader. Muestre SalesOrderID junto con SalesQuota y Bonus.**

5. Agregue las columnas name a la consulta escrita en la pregunta 4 uniéndose en la tabla Person.Person. Vea si puede averiguar qué columnas se usarán para escribir la combinación.

   Puede unir la tabla Person.Person en la tabla SalesOrderHeader o la tabla Sales.SalesPerson.

6. La descripción del catálogo para cada producto se almacena en la tabla Production.ProductModel. Muestre las columnas que describen el producto de la tabla Production.Product, como el color y el tamaño junto con la descripción del catálogo para cada producto.

7. Escriba una consulta que muestre los nombres de los clientes junto con los nombres de productos que han comprado. Sugerencia: ¡Se necesitarán cinco tablas para escribir esta consulta!

Uso de Outer Joins

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Escriba una consulta que muestre todos los productos junto con SalesOrderID incluso si nunca se ha realizado un pedido para ese producto. cruzar con la tabla sales.SalesOrderDetail mediante la columna ProductID.

2. Cambie la consulta escrita en la pregunta 1 para que solo aparezcan en la consulta los productos que no se han pedido.

3. Escriba una consulta que devuelva todas las filas de la tabla Sales.SalesPerson unida a la tabla Sales.SalesOrderHeader cruzando por la columna SalesOrderID con BusinessEntityID, se deben obtener los datos de Sales.SalesPerson aunque no cruce con Sales.SalesOrderHeader. Incluya las columnas SalesOrderID, SalesPersonID y SalesYTD en los resultados.

4. Cambie la consulta escrita en la pregunta 3 para que el nombre del vendedor también se muestre desde la tabla Person.Person.

5. La tabla Sales.SalesOrderHeader contiene claves externas para las tablas Sales.CurrencyRate y Purchasing.ShipMethod. Escriba una consulta que una las tres tablas, asegurándose de que contiene todas las filas de Sales.SalesOrderHeader. Incluya las columnas CurrencyRateID, AverageRate, SalesOrderID y ShipBase.

6. Escriba una consulta que devuelva la columna BusinessEntityID de la tabla Sales.SalesPerson junto con cada ProductID de la tabla Production.Product.

Subqueries

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Mediante una subconsulta, muestre los nombres de producto y los números de ID de producto de la tabla Production.Product que se han pedido (que se encuentran en la tabla **Sales.SalesOrderDetail**).

2. Cambie la consulta escrita en la pregunta 1 para mostrar los productos que no se han pedido.

3. Si la tabla Production.ProductColor no forma parte de la base de datos AdventureWorks2008, ejecute el código listado a continuación para crearlo. Escriba una consulta mediante una subconsulta que devuelva las filas de la tabla Production.ProductColor que no se usan en la tabla Production.Product.

   (Script para la creación de la tabla ProductColor)

```
USE AdventureWorks2008;
GO
IF OBJECT_ID('Production.ProductColor') IS NOT NULL BEGIN DROP TABLE
Production.ProductColor;
END
CREATE table Production.ProductColor
(Color nvarchar(15) NOT NULL PRIMARY KEY)
GO
--Insert most of the existing colors INSERT INTO Production.ProductColor SELECT
DISTINCT COLOR
FROM Production.Product
WHERE Color IS NOT NULL and Color <> 'Silver'
--Insert some additional colors INSERT INTO Production.ProductColor
VALUES ('Green'),('Orange'),('Purple');
```

4. Escriba una consulta que muestre los colores utilizados en la tabla Production.Product que no aparecen en la tabla Production.ProductColor mediante una subconsulta. Utilice la palabra clave DISTINCT antes del nombre de columna para devolver cada color solo una vez.

5. Escriba una consulta UNION que combine ModifiedDate de Person.Person y HireDate de HumanResources.Employee.

Exploración de tablas derivadas y expresiones de tablas comunes

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Con una tabla derivada, una la tabla Sales.SalesOrderHeader con la tabla Sales.SalesOrderDetail. Mostrar las columnas SalesOrderID, OrderDate y ProductID en los resultados. La tabla Sales.SalesOrderDetail debe estar dentro de la consulta de tabla derivada.

2. Vuelva a escribir la consulta en la pregunta 1 con una expresión de tabla común (CTE).

3. Escriba una consulta que muestre a todos los clientes junto con los pedidos realizados en 2001. Use una expresión de tabla común para escribir la consulta e incluir las columnas CustomerID, SalesOrderID y OrderDate en los resultados.

Pensando sobre rendimiento.

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

Ejecute el código siguiente para agregar y rellenar una nueva columna, OrderID, a la tabla Sales.SalesOrderDetail. Después de ejecutar el código, la nueva columna contendrá los mismos datos que la columna SalesOrderID.

```
USE
AdventureWorks2008;
GO
ALTER TABLE Sales.SalesOrderDetail ADD OrderID INT
NULL; GO
UPDATE Sales.SalesOrderDetail SET OrderID = SalesOrderID;
```

1. Asegúrese de activar la opción Incluir plan de ejecución real antes de comenzar. Vea los planes de ejecución y explique por qué una consulta funciona mejor que la otra.

```
--1
SELECT
o.SalesOrderID, d.SalesOrderDetailID FROM
Sales.SalesOrderHeader AS o
INNER JOIN Sales.SalesOrderDetail AS d ON o.SalesOrderID = d.SalesOrderID;
```

```
--2
SELECT
o.SalesOrderID, d.SalesOrderDetailID FROM
Sales.SalesOrderHeader AS o
INNER JOIN Sales.SalesOrderDetail AS
    d ON o.SalesOrderID = d.OrderID;
```

2. Compare los planes de ejecución del ejemplo de tabla derivada y el ejemplo de CTE. Explicar por qué el rendimiento de la consulta es el mismo o por qué una consulta funciona mejor que la otra.

```
USE AdventureWorks2008;
GO
SELECT c.CustomerID, s.SalesOrderID FROM Sales.Customer AS c
INNER JOIN (SELECT SalesOrderID, CustomerID
FROM Sales.SalesOrderHeader) AS s ON c.CustomerID = s.CustomerID;

USE AdventureWorks2008;
GO
WITH orders AS (
SELECT SalesOrderID, CustomerID FROM Sales.SalesOrderHeader
)
SELECT c.CustomerID, orders.SalesOrderID FROM Sales.Customer AS c
INNER JOIN orders ON c.CustomerID = orders.CustomerID
```

4. Agrupación y simplificación de datos

Uso de funciones de agregado.

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Escriba una consulta para determinar el número de clientes en la tabla Sales.Customer.

2. Escriba una consulta que muestre el número total de productos pedidos. Utilice la columna OrderQty de la tabla Sales.SalesOrderDetail y la función SUM.

3. Escriba una consulta para determinar el precio del producto más caro solicitado. Utilice la columna UnitPrice de la tabla Sales.SalesOrderDetail

4. Escriba una consulta para determinar el importe medio de "freight" en la tabla Sales.SalesOrderHeader.

5. Escriba una consulta mediante la tabla Production.Product que muestra el valor mínimo, máximo y promedio de ListPrice.

Uso de la cláusula GROUP BY

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Escriba una consulta que muestre el número total de artículos pedidos para cada producto. Use la tabla Sales.SalesOrderDetail para escribir la consulta.

2. Escriba una consulta mediante la tabla Sales.SalesOrderDetail que muestra un recuento de las líneas de detalle para cada SalesOrderID.

3. Escriba una consulta mediante la tabla Production.Product que enumera un recuento de los productos de cada línea de productos.

4. Escriba una consulta que muestre el recuento de pedidos realizados por año para cada cliente mediante la tabla Sales.SalesOrderHeader.

Uso de la cláusula HAVING

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Escriba una consulta que devuelva un recuento de líneas de detalle en la tabla Sales.SalesOrderDetail

2. Escriba una consulta que cree una suma de LineTotal en la tabla Sales.SalesOrderDetail agrupada por SalesOrderID. Incluya solo las filas en las que la suma supera las 1.000.

3. Write a query that groups the products by **ProductModelID** along with a count. Display the rows that have a count that equals 1. 3. Escriba una consulta que agrupe los productos por ProductModelID junto con un recuento (count). Muestre las filas que tienen un recuento que es igual a 1.

4. Cambie la consulta en la pregunta 3 para que solo se incluyan los productos con el color azul o rojo.

Uso de DISTINCT

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Escriba una consulta mediante la tabla Sales.SalesOrderDetail para crear un recuento de valores ProductID únicos que se han pedido.

2. Escriba una consulta mediante la tabla Sales.SalesOrderHeader que devuelve el recuento de valores TerritoryID únicos por cliente.

Uso de consultas agregadas con más de una tabla

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Escriba una consulta que una a las tablas Person.Person, Sales.Customer y Sales.SalesOrderHeader para devolver una lista de los nombres de los clientes junto con un recuento de los pedidos realizados.

2. Escriba una consulta mediante las tablas Sales.SalesOrderHeader, Sales.SalesOrderDetail y Production.Product para mostrar la suma total de productos por ProductID y OrderDate.

Lógica de consulta agregada de aislación

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Escriba una consulta que una la tabla HumanResources.Employee a la tabla Person.Person para que pueda mostrar las columnas FirstName, LastName y HireDate para cada empleado. Muestre JobTitle junto con un recuento de empleados para el título. Use una tabla derivada para resolver esta consulta.

2. Vuelva a escribir la consulta de la pregunta 1 utilizando un CTE.

3. Vuelva a escribir la consulta de la pregunta 1 utilizando la cláusula OVER.

4. Muestre CustomerID, SalesOrderID y OrderDate para cada fila Sales.SalesOrderHeader siempre que el cliente haya realizado al menos cinco pedidos. Utilice cualquiera de las técnicas de esta sección para llegar a la consulta, aquí hay tres posibles soluciones.

Pensando sobre rendimiento.

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Asegúrese de que la opción Incluir plan de ejecución real esté activada antes de escribir y ejecutar el siguiente código. Compare los planes de ejecución para ver si la consulta CTE funciona mejor que la consulta de cláusula OVER.

```
USE
AdventureWorks2008;
GO
--1
WITH SumSale AS
   (SELECT SUM(TotalDue) AS
        SumTotalDue, CustomerID
    FROM
    Sales.SalesOrderHeader
    GROUP BY CustomerID)
SELECT o.CustomerID, TotalDue,
    TotalDue / SumTotalDue * 100 AS
PercentOfSales FROM SumSale INNER JOIN
Sales.SalesOrderHeader AS o ON SumSale.CustomerID
= o.CustomerID
ORDER BY CustomerID;


--2
SELECT CustomerID, TotalDue,
    TotalDue / SUM(TotalDue) OVER(PARTITION BY CustomerID) * 100 AS
PercentOfSales FROM Sales.SalesOrderHeader
ORDER BY CustomerID;
```

2. Las siguientes consultas contienen cada una dos cálculos: porcentaje de ventas por cliente y porcentaje de ventas por territorio. Escriba y ejecute el código para ver la diferencia en el rendimiento. Asegúrese de que la opción Incluir plan de ejecución real esté activada antes de ejecutar el código.

```
USE
AdventureWorks2008;
GO

--1
WITH SumSale AS
   (SELECT SUM(TotalDue) AS
        SumTotalDue, CustomerID
    FROM
    Sales.SalesOrderHeader
    GROUP BY CustomerID),
 TerrSales AS
    (SELECT SUM(TotalDue) AS SumTerritoryTotalDue,
     TerritoryID FROM Sales.SalesOrderHeader
     GROUP BY TerritoryID )
SELECT o.CustomerID,
TotalDue,
    TotalDue / SumTotalDue * 100 AS PercentOfCustSales,
    TotalDue / SumTerritoryTotalDue * 100 AS
    PercentOfTerrSales
FROM SumSale
INNER JOIN Sales.SalesOrderHeader AS o ON SumSale.CustomerID =
o.CustomerID INNER JOIN TerrSales ON TerrSales.TerritoryID =
o.TerritoryID
ORDER BY CustomerID;

--2
SELECT CustomerID, TotalDue,
    TotalDue / SUM(TotalDue) OVER(PARTITION BY CustomerID) * 100 AS
PercentOfCustSales,
    TotalDue / SUM(TotalDue) OVER(PARTITION BY TerritoryID) * 100 AS
PercentOfTerrSales
FROM
Sales.SalesOrderHeader
ORDER BY CustomerID;
```

5. Manipulación de datos.

Insertando nuevas filas.

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorksLT2008. (Atento al cambio de BDD)

Ejecuta el siguiente código para crear las tablas requeridas:

```
USE
AdventureWorksLT2008;
GO
IF EXISTS (SELECT * FROM sys.objects
            WHERE object_id =
               OBJECT_ID(N'[dbo].[demoProduct]') AND type in
               (N'U'))
DROP TABLE
[dbo].[demoProduct] GO

CREATE TABLE [dbo].[demoProduct](
    [ProductID] [INT] NOT NULL PRIMARY
    KEY,
    [Name] [dbo].[Name] NOT
    NULL, [Color]
    [NVARCHAR](15) NULL,
    [StandardCost] [MONEY] NOT
    NULL, [ListPrice] [MONEY] NOT
    NULL, [Size] [NVARCHAR](5)
    NULL, [Weight] [DECIMAL](8, 2)
    NULL,
);
IF EXISTS (SELECT * FROM sys.objects
            WHERE object_id =
                OBJECT_ID(N'[dbo].[demoSalesOrderHeader]') AND type in
                (N'U'))
DROP TABLE
[dbo].[demoSalesOrderHeader] GO

CREATE TABLE [dbo].[demoSalesOrderHeader](
    [SalesOrderID] [INT] NOT NULL PRIMARY
    KEY, [SalesID] [INT] NOT NULL IDENTITY,
    [OrderDate] [DATETIME] NOT NULL,
    [CustomerID] [INT] NOT NULL,
    [SubTotal] [MONEY] NOT NULL,
```

```
    [TaxAmt] [MONEY] NOT NULL,
    [Freight] [MONEY] NOT NULL,



    [DateEntered] [DATETIME],
    [TotalDue] AS (ISNULL(([SubTotal]+[TaxAmt])+[Freight],(0))),
    [RV] ROWVERSION NOT NULL);
GO

ALTER TABLE [dbo].[demoSalesOrderHeader] ADD CONSTRAINT
[DF_demoSalesOrderHeader_DateEntered]
DEFAULT (GETDATE()) FOR [DateEntered];

GO
IF EXISTS (SELECT * FROM sys.objects
    WHERE object_id =
    OBJECT_ID(N'[dbo].[demoAddress]') AND type in
    (N'U'))
DROP TABLE
[dbo].[demoAddress] GO

CREATE TABLE [dbo].[demoAddress](
    [AddressID] [INT] NOT NULL IDENTITY PRIMARY
    KEY, [AddressLine1] [NVARCHAR](60) NOT NULL,
    [AddressLine2] [NVARCHAR](60) NULL,
    [City] [NVARCHAR](30) NOT NULL,
    [StateProvince]   [dbo].[Name]   NOT
    NULL,   [CountryRegion]   [dbo].[Name]
    NOT NULL, [PostalCode] [NVARCHAR](15)
    NOT NULL
);
```

1. Escriba una instrucción SELECT para recuperar datos de la tabla SalesLT.Product. Utilice estos valores para insertar cinco filas en la tabla dbo.demoProduct utilizando valores literales. Escriba cinco instrucciones INSERT individuales. Las filas que elijas para insertar deben variar.

2. Inserte cinco filas más en la tabla dbo.demoProduct. Esta vez usando una sola instrucción INSERT, las filas que elijas deben variar.

3. Escriba una instrucción INSERT que inserte todas las filas en la tabla dbo.demoSalesOrderHeader de la tabla SalesLT.SalesOrderHeader. Sugerencia: Preste mucha atención a las propiedades de las columnas de la tabla dbo.demoSalesOrderHeader. No inserte un valor en las columnas SalesID, DateEntered y RV.

4. Escriba una instrucción SELECT INTO que cree una tabla, dbo.tempCustomerSales, que muestre cada CustomerID de SalesLT.Customer junto con un recuento de los pedidos realizados y el importe total adeudado para cada cliente.

5. Escriba una instrucción INSERT que inserte todos los productos en la tabla dbo.demoProduct de la tabla SalesLT.Product que aún no se hayan insertado. No especifique valores ProductID literales en la instrucción.

6. Escriba una instrucción INSERT que inserte todas las direcciones en la tabla dbo.demoAddress de la tabla SalesLT.Address. Antes de ejecutar la instrucción INSERT, escriba y ejecute el comando para que pueda insertar valores en la columna AddressID.

Bosonit

Borrando filas.

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorksLT2008. (Atento al cambio de BDD)

Antes de empezar los ejercicios ejecutar el siguiente código:

```
USE AdventureWorksLT2008;
GO

IF EXISTS (SELECT * FROM sys.objects
WHERE object_id = OBJECT_ID(N'[dbo].[demoProduct]') AND type in (N'U'))
DROP TABLE [dbo].[demoProduct]; GO
SELECT * INTO dbo.demoProduct FROM SalesLT.Product; IF EXISTS (SELECT * FROM sys.objects
WHERE object_id = OBJECT_ID(N'[dbo].[demoCustomer]')
AND type in (N'U')) DROP TABLE [dbo].[demoCustomer]; GO

SELECT * INTO dbo.demoCustomer FROM SalesLT.Customer; IF EXISTS (SELECT * FROM
sys.objects
WHERE object_id = OBJECT_ID(N'[dbo].[demoAddress]') AND type in (N'U'))
DROP TABLE [dbo].[demoAddress]; GO

SELECT * INTO dbo.demoAddress FROM SalesLT.Address; IF EXISTS (SELECT * FROM sys.objects
WHERE object_id = OBJECT_ID(N'[dbo].[demoSalesOrderHeader]') AND type in (N'U'))
DROP TABLE [dbo].[demoSalesOrderHeader]; GO
SELECT * INTO dbo.demoSalesOrderHeader FROM SalesLT.SalesOrderHeader; IF EXISTS (SELECT
* FROM sys.objects
WHERE object_id = OBJECT_ID(N'[dbo].[demoSalesOrderDetail]')
AND type in (N'U'))
DROP TABLE [dbo].[demoSalesOrderDetail]; GO

SELECT * INTO dbo.demoSalesOrderDetail FROM SalesLT.SalesOrderDetail;
```

1. Escriba una consulta que elimine las filas de la tabla dbo.demoCustomer donde los valores LastName comienzan con la letra S.

2. Elimine las filas de la tabla dbo.demoCustomer si el cliente no ha realizado un pedido o si la suma de TotalDue de la tabla dbo.demoSalesOrderHeader para el cliente es menor de $1,000.

3. Elimine las filas de la tabla dbo.demoProduct que nunca se han pedido.

Actualizar filas existentes.

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorksLT2008. (Atento al cambio de BDD)

Antes de empezar los ejercicios vuelve a ejecutar el script del punto anterior para recrear las tablas.

1. Escriba una instrucción UPDATE que cambie todos los valores NULL de la columna AddressLine2 de la tabla dbo.demoAddress a N/A.

2. Escriba una instrucción UPDATE que aumente el ListPrice de cada producto de la tabla dbo.demoProduct en un 10 por ciento.

3. Escriba una instrucción UPDATE que corrija el valor de UnitPrice con el de ListPrice de cada fila de la tabla dbo.demoSalesOrderDetail cruzando con la tabla dbo.demoProduct.

4. Escriba una instrucción UPDATE que actualice la columna SubTotal de cada fila de la tabla dbo.demoSalesOrderHeader con la suma de la columna LineTotal de la tabla dbo.demoSalesOrderDemo.

Uso de transacciones.

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorksLT2008. (Atento al cambio de BDD)

Antes de empezar los ejercicios ejecutar el siguiente código:

```
IF OBJECT_ID('dbo.Demo') IS NOT NULL BEGIN
    DROP TABLE dbo.Demo;
EN
D;
GO
CREATE TABLE dbo.Demo(ID INT PRIMARY KEY, Name VARCHAR(25));
```

1. Escriba una transacción que incluya dos instrucciones insert para agregar dos filas a la tabla dbo.demo.

2. Escriba una transacción que incluya dos instrucciones insert para agregar dos filas más a la tabla dbo.demo. Intente insertar una letra en lugar de un número en la columna ID de una de las instrucciones. Seleccione los datos de la tabla demo para ver qué filas llegaron a la tabla.

6. Comprender la lógica de programación de T-SQL

Uso de variables.

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Escriba un script que declare una variable entera llamada @myInt. Asigne 10 a la variable y, a continuación, imprímalo.

2. Escriba un script que declare una variable VARCHAR(20) denominada @myString. Asignar “Esto es una prueba” a la variable e imprimirla.

3. Escriba un script que declare dos variables enteras denominadas @MaxID y @MinID. Utilice las variables para imprimir los valores SalesOrderID más altos y más bajos de la tabla Sales.SalesOrderHeader.

4. Escriba un script que declare una variable entera llamada @ID. Asigne el valor 70000 a la variable. Utilice la variable de una instrucción SELECT que devuelve todos los valores SalesOrderID de la tabla Sales.SalesOrderHeader que tienen un SalesOrderID mayor que el valor de la variable.

5. Escriba un script que declare tres variables, una variable de entero llamada @ID, una variable NVARCHAR(50) llamada @FirstName y una variable NVARCHAR(50) llamada @LastName. Utilice una instrucción SELECT para establecer el valor de las variables con la fila de la tabla Person.Person con BusinessEntityID nro 1. Imprima una instrucción en el formato "BusinessEntityID: FirstName LastName".

6. Escriba un script que declare una variable entera llamada @SalesCount. Establezca el valor de la variable en el recuento total de ventas en la tabla Sales.SalesOrderHeader. Utilice la variable en una instrucción SELECT que muestre la diferencia entre el @SalesCount y el recuento de ventas por cliente.

Usando los constructores IF…ELSE

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Declare una variable entera llamada @Count para guardar el recuento de todos los registros Sales.SalesOrderDetail. Agregue un bloque IF que imprima "Más de 100.000" si el valor de la variable supera los 100.000. De lo contrario, imprima "100.000 o menos."

2. Escriba un lote que contenga bloques IF anidados. El bloque exterior debe comprobar si el mes es octubre o noviembre. Si ese es el caso, imprima "El mes es" y el nombre del mes. El bloque interior debe comprobar si el año es par o impar e imprimir el resultado. Puede modificar el mes para comprobar que el bloque interno se activa

3. Escriba un lote que utilice IF EXISTS para comprobar si hay una fila en la tabla Sales.SalesOrderHeader que tenga SalesOrderID n.o 1. Imprima "Hay un SalesOrderID 1" o "No hay un SalesOrderID 1" dependiendo del resultado.

Uso de WHILE

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Escriba un script que contenga un bucle WHILE que imprima las letras de la A a la Z. Utilice la función CHAR para cambiar un número a una letra. Inicie el bucle con el valor 65.

2. Escriba un script que contenga un bucle WHILE anidado dentro de otro bucle WHILE. El contador para el bucle externo debe contar de 1 a 100. El contador para el bucle interno debe contar de 1 a 5. Imprima el producto de los dos contadores dentro del bucle interior.

3. Cambie el script en la pregunta 2 para que el bucle interno finalice en lugar de imprimir cuando el contador para el bucle exterior es divisible por 5.

4. Escriba un script que contenga un bucle WHILE que cuente de 1 a 100. Imprima "Impar" o "par" dependiendo del valor del contador.

Manejo de errores

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Escriba una instrucción que intente insertar una fila duplicada en la tabla HumanResources.Department. Utilice la función @@ERROR para mostrar el error.

2. Cambie el código que escribió en la pregunta 1 para usar TRY… CATCH. Muestre el número de error, el mensaje y la gravedad.

3. Cambie el código que escribió en la pregunta 2 para generar un mensaje de error personalizado en lugar del mensaje de error real.

Creación de tablas temporales y variables de tabla

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Cree una tabla temporal denominada #CustomerInfo que contenga las columnas CustomerID, FirstName y LastName. Incluya las columnas CountOfSales y SumOfTotalDue. Rellene la tabla con una consulta mediante las tablas Sales.Customer, Person.Person y Sales.SalesOrderHeader.

2. Cambie el código escrito en la pregunta 1 para utilizar una variable de tabla en lugar de una tabla temporal.

3. Cree una variable de tabla con dos columnas enteras, una de ellas una columna INDENTITY. Utilice un bucle WHILE para rellenar la tabla con 1.000 enteros aleatorios mediante la siguiente fórmula. Utilice un segundo bucle WHILE para imprimir los valores de la variable de tabla uno por uno.

```
CAST(RND() * 10000 AS INT) + 1
```

7. Mover la lógica a la base de datos

Creando tablas.

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Cree una tabla llamada dbo.testCustomer. Incluya una columna CustomerID que sea primary key e identity. Incluya las columnas FirstName y LastName. Incluya una columna Age con una check constraint que especifique que el valor debe ser inferior a 120. Incluya una columna que su valor predeterminado sea “Y” y solo permita Y o N. Agregar algunas filas a la tabla.

2. Cree una tabla llamada dbo.testOrder. Incluya una columna CustomerID que sea una foreign key que apunte a dbo.testCustomer. Incluya una columna OrderID que sea una primary key de tipo identity. Incluya una columna OrderDate que tenga como valor predeterminado la fecha y hora actuales. Incluya una columna ROWVERSION. Agregue algunas filas a la tabla.

3. Cree una tabla llamada dbo.testOrderDetail. Incluya una columna OrderID que sea una foreign key que apunte a dbo.testOrder. Incluya una columna ItemID Integer, una columna Price y una columna Qty. La clave principal debe ser una clave compuesta de las columnas OrderID y ItemID. Cree una columna calculada denominada LineItemTotal que multiplique los precios De precio Cantidad. Agregue algunas filas a la tabla.

Creación de vistas

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Cree una vista denominada dbo.vw_Products que muestre una lista de los productos de la tabla Production.Product unidos a la tabla Production.ProductCostHistory. Incluya columnas que describan el producto y muestren el historial de costes de cada producto. Pruebe la vista creando una consulta que recupere datos de la vista.

2. Cree una vista denominada dbo.vw_CustomerTotals que muestre las ventas totales de la columna TotalDue por año y mes para cada cliente. Pruebe la vista creando una consulta que recupere datos de la vista.

Creación de funciones definidas por el usuario

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Cree una función definida por el usuario llamada dbo.fn_AddTwoNumbers que acepte dos parámetros enteros. Devuelve el valor que es la suma de los dos números. Pruebe la función.

2. Cree una función definida por el usuario llamada dbo.trim que toma un parámetro VARCHAR(250). Esta función debe recortar los espacios desde el principio y el final de una cadena. Pruebe la función.

3. Cree una función llamada dbo.fn_RemoveNumbers que elimine los caracteres numéricos de una cadena VARHCHAR(250). Pruebe la función. Sugerencia: La función ISNUMERIC comprueba si una cadena es numérica.

4. Escriba una función llamada dbo.fn_FormatPhone que tome una cadena de diez números. La función formateará la cadena en este formato de número de teléfono: “(###) ###-####.” Pruebe la función.

Creación de procedimientos almacenados

Para realizar los siguientes ejercicios debemos utilizar la base de datos AdventureWorks2008.

1. Cree un procedimiento almacenado llamado dbo.usp_CustomerTotals en lugar de la vista de la pregunta 1 en el Ejercicio de creación de vistas. Pruebe el procedimiento almacenado

2. Modifique el procedimiento almacenado creado en la pregunta 1 para incluir un parámetro @CustomerID. Utilice el parámetro de la cláusula WHERE de la consulta en el procedimiento almacenado. Pruebe el procedimiento almacenado.

3. Cree un procedimiento almacenado llamado dbo.usp_ProductSales que acepte un ProductID para un parámetro y tenga un parámetro OUTPUT que devuelva el número vendido para el producto. Pruebe el procedimiento almacenado