MODELOS Y BASES DE DATOS

Consultas SQL Básico

2020-1

Laboratorio 2/6

Integrantes:

* Juan Andrés Pico
* Ana Gabriela Silva

**PARTE UNO. Refactorizacion**

1. **Modelo conceptual**
2. El mayor cambio que hubo en nuestro modelo conceptual fue el de quitar los compartimientos en los atributos y las operaciones ya que no eran necesarias, además le agregamos colores a las relaciones.
3. Los grandes conceptos que fueron tomados son los de Product y Customer.
4. **Modelo lógico**
5. No hubo ningún cambio.
6. Los grandes CRUDS que se tomaron fue Product y Customer.

**PARTE DOS. División por ciclos.**

1. **Astah**
2. **Astah**
3. **Astah**

**PARTE TRES. Ciclo Uno.**

1. **Modelo conceptual. Consultas. (¿qué ofrece?)**
2. **SQLZOO**

**Show the first name and the email address of customer with CompanyName 'Bike World'**

SELECT FirstName, EmailAddress v

FROM Customer

WHERE CompanyName = "Bike World"

**Show the CompanyName for all customers with an address in City 'Dallas'.**

SELECT CompanyName FROM Customer

JOIN CustomerAddress ON Customer.CustomerID = CustomerAddress.CustomerID

JOIN Address ON CustomerAddress.AddressID = Address.AddressID

WHERE Address.City = "Dallas"

1. **Nuevas consultas**

**El número de clientes en cada una de las ciudades**

SELECT COUNT(CustomerID) AS num, City

FROM Customer JOIN Address ON(CustomerID = AddressID)

GROUP BY City

ORDERED BY CustomerID

**Los clientes con más de dos direcciones registradas**

SELECT CustomerID

FROM Customer

JOIN Address ON(CustomerID = AddressID)

WHERE AddressLine2 IS NOT NULL

ORDER BY AddressID

**Los clientes ubicados en más de una ciudad**

SELECT COUNT(FirstName) AS x Customers, CompanyName, AddressType, City From Customer x

JOIN CustomerAddress y ON(x.CustomerID = y.CustomerID)

JOIN Address z ON(y.AddressID = z.AddressID)

GROUP BY City

ORDER BY CompanyName

**Los clientes que no han modificado su dirección de cobro.**

SELECT CustomerID, City

FROM CustomerAddres x JOIN Address y

ON x.AddressId = y.AddressID

WHERE (SELECT COUNT(BillToAddressID) FROM SalesOrderHeader z

WHERE x.CustomerID = z. CustomerID) = 1

ORDER BY City

**Los clientes que han recibido descuentos de más del 10% de sus compras.**

No se logró realizar.

1. **Mejor consulta**

**Conocer dirección de entrega y cobro de la mercancía.**

SELECT BillToAddress, ShipToAddress FROM SalesOrderHeader

WHERE BillToAddress IS NOT NULL AND ShipToAddress IS NOT NULL

**PARTE CUATRO. Ciclo dos.**

1. **Modelo conceptual. Consultas. (¿qué ofrece?)**
2. **SQLZOO**

**How many items with ListPrice more than $1000 have been sold?**

SELECT COUNT(\*) AS Total

FROM SalesOrderDetail

JOIN Product ON SalesOrderDetail.ProductID = Product.ProductID

WHERE Product.ListPrice > 1000

**A "Single Item Order" is a customer order where only one item is ordered. Show the SalesOrderID and the UnitPrice for every Single Item Order.**

SELECT SalesOrderID, UnitPrice

FROM SalesOrderDetail

WHERE OrderQty = 1

1. **Nuevas consultas**

**Total de ventas por cliente**

**[identificador, nombre de la compañía y valor total] (Orden alfabético)**

SELECT Customer.CustomerID, Customer.CompanyName , SUM(SalesOrderHeader.Subtotal+SalesOrderHeader.TaxAmt + SalesOrderHeader.Freight) AS Total FROM Customer

JOIN SalesOrderHeader ON SalesOrderHeader.CustomerID = Customer.CustomerID

GROUP BY CompanyName ASC

**Total de ventas por categoría**

**[categoría de productos, total de unidades y total de ventas] (Ordenado por total de ventas)**

SELECT ProductCategory.Name ,COUNT(Product.ProductID),SUM(SalesOrderHeader.OrderDate) Product.Name FROM ProductCategory

JOIN Product ON ProductCategory.ProductCategoryID = Product.ProductCategoryID

JOIN SalesOrderDetail ON Product.ProductID LIKE SalesOrderDetail.ProductID

JOIN SalesOrderHeader ON SalesOrderHeader.SalesOrderID LIKE SalesOrderDetail.SalesOrderID

**El producto más vendido a cada uno de los clientes.**

**[identificador, nombre de la compañía, nombre producto y unidades] (Orden alfabético por producto y unidades**

**de mayor a menor)**

SELECT w.ProductID, CompanyName , Name , z.OrderQty

FROM Customer x JOIN SalesOrderHeader y Join SalesOrderDetail z JOIN Product w ON

x.CustomerId = y.CustomerID AND y.SalesOrderID = z.SalesOrderID AND z.ProductID=w.ProductID

WHERE z.OrderQty = (SELECT MAX(OrderQty) FROM SalesOrderDetail v

WHERE v.SalesOrderID = y.SalesOrderID)

ORDER BY Name, OrderQty DESC

**Los productos que más han cambiado su precio de venta unitario.**

**[identificador, nombre, precio lista precio unitario menor, precio unitario mayor, diferencia] (Ordenado de mayor**

**a menor por diferencia).**

SELECT DISTINCT x.ProductID, Name, ListPrice, Pmin AS 'Min Price', Pmax AS 'Max Price', Pmax-Pmin AS DIFERENCE

FROM Product x JOIN SalesOrderDetail y

JOIN (SELECT MAX(UnitPrice) AS Pmax, ProductID FROM SalesOrderDetail GROUP BY ProductID) t1

JOIN (SELECT MIN(UnitPrice) AS Pmin, ProductID FROM SalesOrderDetail GROUP BY ProductID) t2

ON x.ProductID=y.ProductID AND x.ProductID=t1.ProductID AND x.ProductID=t2.ProductID

WHERE Pmax-Pmin>0

ORDER BY Diference DESC

1. **Mejor consulta**

**La Lista de nombres de los productos con su respectivo precio en lista y precio por unidad organizado ascendentemente por el nombre**

SELECT DISTINCT Name, ListPrice, UnitPrice FROM Product

JOIN SalesOrderDetail ON Product.ProductID = SalesOrderDetail.ProductID

ORDER BY Name ASC

**Retrospectiva**

1. El tiempo invertido en el avance ha sido de 24 horas. 2 integrantes.
2. Falta una consulta que no se logró realizar.
3. Aprender a diferencia un gran concepto a un concepto y un CRUD en un modelo lógico.
4. Generar algunas consultas pedidas.
5. Trabajar en equipo, recolectar ideas. Nos comprometemos a ser más eficientes en el trabajo durante un tiempo determinado.