Unit 2 SQL Procedures and Views

**Corey Crooks**

**Purdue University Global**

**IT350 – Advanced Database Concepts**

**Jason Johnson**

**February 15th, 2023**

**Problem 1:** Create a stored procedure that lists the staff's last name, first name, and email address from all the company stores. The output needs to be in alphabetical order based on the last name then by the first name. The skeleton of the stored procedure is provided below.

USE BikeStores

GO

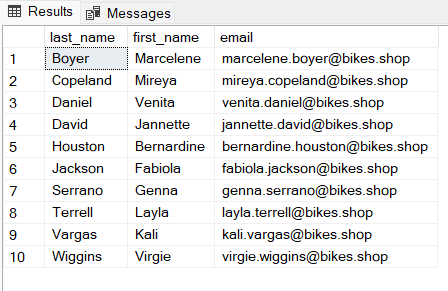
CREATE PROCEDURE GetCompanyStaffList

AS

SELECT last\_name, first\_name, email

FROM Sales.Staffs

ORDER BY last\_name, first\_name



**Problem 2:** Create a stored procedure that lists the customers from a specified zip code. The skeleton of the stored procedure is provided below.

USE BikeStores

GO

CREATE PROCEDURE GetCustomersByZipCode

@ZipCode VARCHAR(5)

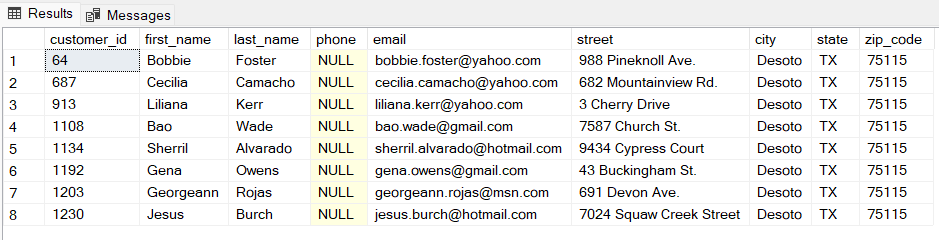
AS

SELECT \*

FROM Sales.Customers

WHERE zip\_code = @ZipCode;

EXEC GetCustomersByZipCode @ZipCode = '75115';



**Problem 3:** Create a stored procedure that lists mountain bikes for a specified brand. The output must show the bike name, model year, and list price. The results need to be presented in alphabetical order based on the bike name. The skeleton of the stored procedure is provided below.

USE BikeStores

GO

CREATE PROCEDURE GetMountainBikesByBrand

@Brand VARCHAR(255)

AS

SELECT product\_name, model\_year, list\_price

FROM Production.Products P INNER JOIN Production.Brands B

ON P.brand\_id = B.brand\_id

INNER JOIN Production.Categories C

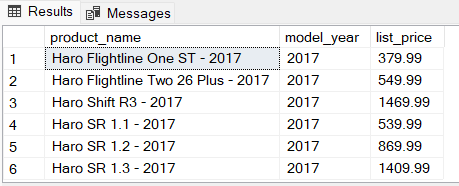
ON P.category\_id = C.category\_id

WHERE C.category\_name = 'Mountain Bikes'

AND B.brand\_name = @Brand

ORDER BY product\_name;

EXEC GetMountainBikesByBrand @Brand = 'Haro';



**Problem 4:** Create a stored procedure that lists bikes within a specified price range. The output must show the bike name, model year, and list price. The results need to be presented in alphabetical order based on the bike name. The skeleton of the stored procedure is provided below.

USE BikeStores

GO

CREATE PROCEDURE GetBikesByPriceRange

@MinPrice Decimal(10,2), @MaxPrice Decimal(10,2)

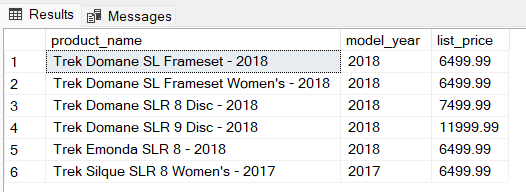
AS

SELECT product\_name, model\_year, list\_price

FROM Production.Products

WHERE list\_price BETWEEN @MinPrice AND @MaxPrice

ORDER BY product\_name;

EXEC GetBikesByPriceRange @MinPrice=6000, @MaxPrice=12000; 

**Problem 5:** Create a stored procedure that lists the top five priced bikes at a specified store. The output must show the bike name, model year, and list price. The results need to be presented in descending price order. The skeleton of the stored procedure is provided below.

USE BikeStores

GO

CREATE PROCEDURE GetTopFivePricedBikesByStoreName

@StoreName VARCHAR(255)

AS

SELECT TOP 5 product\_name, model\_year, list\_price

FROM Production.Products PR INNER JOIN Production.STocks SK

ON PR.product\_id = SK.product\_id

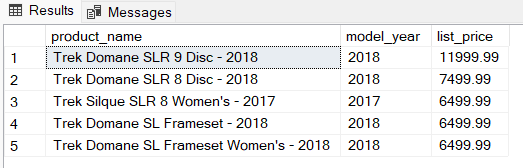
INNER JOIN Sales.Stores ST

ON SK.store\_id = ST.store\_id

WHERE store\_name = @StoreName

ORDER BY list\_price DESC;

EXEC GetTopFivePricedBikesByStoreName @StoreName='Baldwin Bikes';



**Problem 6:** Establish a database view called **CompanyStaffListView**that lists the last name, first name, and email address of the staff from all the company stores. Then execute a SELECT query to verify the view.

USE BikeStores

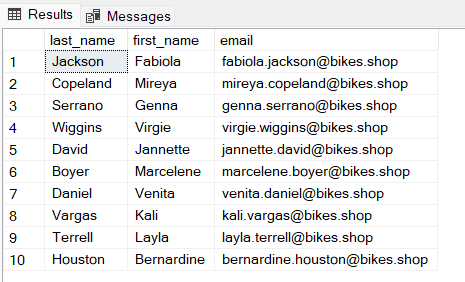
GO

CREATE VIEW CompanyStaffListView

AS

SELECT last\_name, first\_name, email

FROM Sales.Staffs;



**Problem 7:** Establish a database view called **BuffaloCustomersView**that lists the last name, first name, and email address of customers that live in the city of Buffalo, NY. Then execute a SELECT query to verify the view.

USE BikeStores

GO

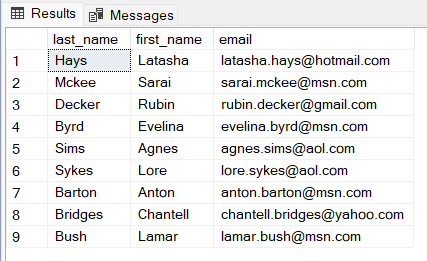
CREATE VIEW BuffaloCustomersView

AS

SELECT last\_name, first\_name, email

FROM Sales.Customers

WHERE city = 'Buffalo' AND state = 'NY';



**Problem 8:** Establish a database view called **LowRowlettBikesInventoryView**that lists the bikes with stock levels less than 2 at the Rowlett Bikes store. The view needs to contain the bike name, brand name, and stock quantity amount. Then execute a SELECT query to verify the view.

USE BikeStores

GO

CREATE VIEW LowRowlettBikesInventoryView

AS

SELECT product\_name, brand\_name, quantity

FROM Production.Products PR INNER JOIN Production.Stocks SK

ON PR.product\_id = SK.product\_id

INNER JOIN Sales.Stores ST

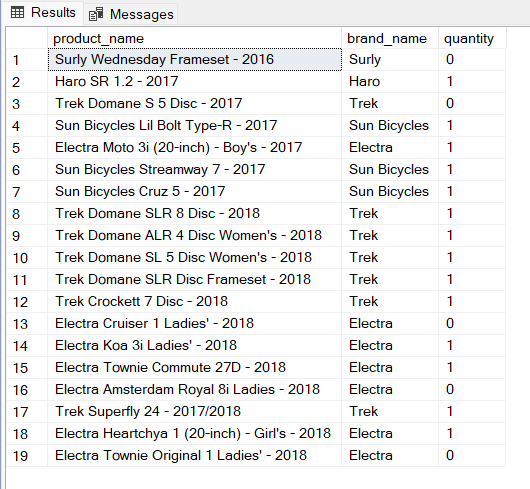
ON SK.store\_id = ST.store\_id

INNER JOIN Production.Brands BR

ON PR.brand\_id = BR.brand\_id

WHERE store\_name = 'Rowlett Bikes'

AND SK.quantity <2;



**Problem 9:** Establish a database view called **LowCostBikesView**that lists the bikes costing $200 or less. The view needs to contain the bike name, brand name, and list price. Then execute a SELECT query to verify the view.

USE BikeStores

GO

CREATE VIEW LowCostBikesView

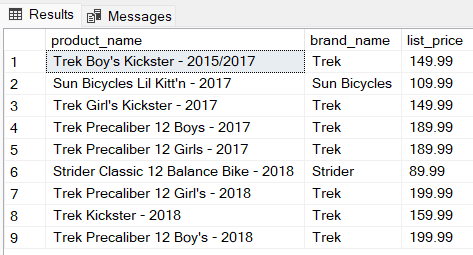
AS

SELECT product\_name, brand\_name, list\_price

FROM Production.Products PR INNER JOIN Production.Brands BR

ON PR.brand\_id = BR.brand\_id

WHERE PR.list\_price <= 200;



**Problem 10:** Establish a database view called **UnshippedRowlettBikesOrdersView**that lists the orders that have not been shipped. The view needs to contain the order ID, customer first name, customer last name, and order date. Unshipped orders are signified by order\_status = 1. Then execute a SELECT query to verify the view.

USE BikeStores

GO

CREATE VIEW UnshippedRowlettBikesOrdersView

AS

SELECT order\_id, first\_name, last\_name, order\_date

FROM Sales.Orders OD INNER JOIN Sales.Customers CR

ON OD.customer\_id = CR.customer\_id

INNER JOIN Sales.Stores ST

ON OD.store\_id = ST.store\_id

WHERE ST.store\_name = 'Rowlett Bikes'

AND OD.order\_status = 1;

