Kyle Welch

12/18/19

Project #2

**Summary**

*Business Case*

The scope of this project is to build a database of past purchases for the purpose of performing historical analysis. Historical analysis is the primary factor that determines cost savings for comparable projects. Thus, creating and maintaining a database to understand past project pricing is both necessary and vital to the business. In addition, the new historical pricing database aims to eliminate the loss of past purchase data and to reduce the amount of time purchasers spend looking for comparable purchases. The overview of this project will define the rules, stakeholders, data questions, expectations, conceptual model, and logical model of the potential historical pricing database.

The logical model was implemented into a live working database through SQL Server Management Studio. The data questions that were proposed to be solved by this database were answered via the use FUNCTION, VIEW, and STORED PROCEDURE statements in SQL. Access was chosen for the user interface for this database because all computer’s at our company come standard with Access installed making it convenient for end users to access our database without further program installations.

*Rules*

Purchasers are assigned to a project and tasked with making purchases for projects. Projects can only ever be assigned to one purchaser, but a purchaser can be assigned to multiple projects.

All projects belong to at least one operating company, if not multiple. All operating companies have multiple projects.

A project must have at least one order associated with it, if not multiple. An order must only be associated with one project.

A supplier can be responsible for many orders. An order must only be attached to one supplier.

*Stakeholders*

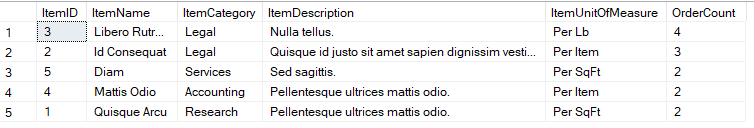
Purchasers – Purchasers needs to be able determine savings based on historical projects that have been awarded by the organization. The reduction in time spent hunting down past historical pricing information will also be a positive for this stakeholder.

Project Managers – Project managers determine proposed budgets based on past comparable projects that have been awarded. Project managers are also tasked with maintaining project’s timelines, with a reduction in the time needed for historical analysis with this proposed database, project managers will be benefitted.

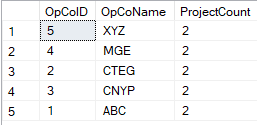
Engineers – Engineers determine what items are needed based on past comparable projects that have been awarded.

*Data Questions*

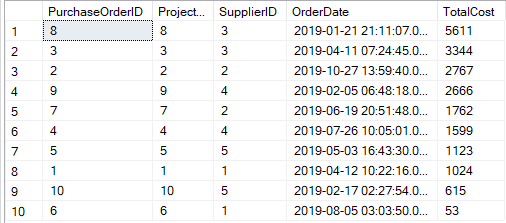
* How often are specific items bought in different orders?



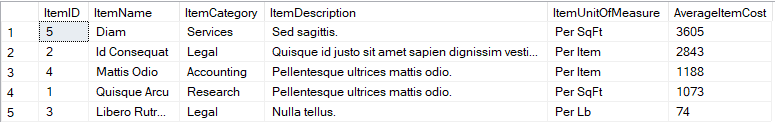
* How many projects have each Operating Company created?



* What are the top 10 orders by highest total cost?



* What is the average cost of items that have been purchased?



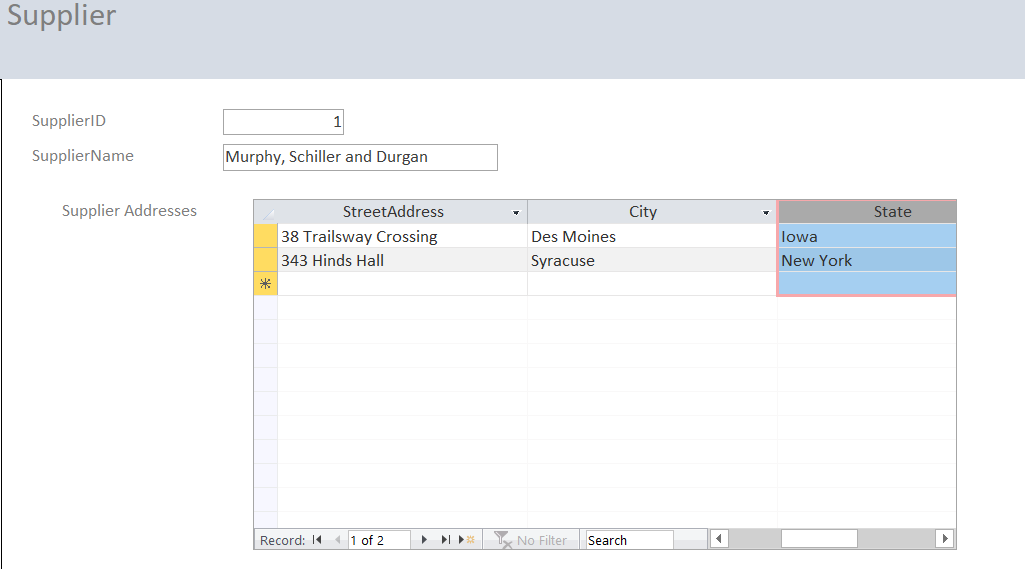
* How will new Purchasers be added to the database



* How will Purchasers be assigned to a new project



* If a Supplier opens a new location, how can an end user update the Suppliers Address?



*Expectations*

The expectation of this project is to create a centralized database for storing historical pricing data related to awarded projects. This database will prevent the loss and duplication of data that would otherwise have a negative effect on historical data analysis accuracy. The creation of this database will allow purchasers to perform historical analysis quicker and more accurately; as well as, allow project managers and engineers a view of past projects to create their budget estimates and item requirements.

**Conceptual Model**

**A screenshot of a cell phone

Description automatically generated**

*Data Glossary*

|  |  |  |
| --- | --- | --- |
| Entity | Attribute | Definition |
| Purchaser | PurchaserID | Unique Purchaser identifier;  Primary Key, surrogate key |
| Name | Purchaser’s name;  required |
| Email | Purchaser’s email;  Required, unique |
| Phone Number | Purchaser’s Phone number;  Required, unique |
| Project | ProjectID | Unique Project identifier;  Primary Key, surrogate key |
| Project Name | Name of project;  Required, unique |
| Project Description | Description of project;  Required |
| Operating Company | OpCoID | Unique Operating Company identifier;  Primary Key, surrogate key |
| OpCo Name | Name of Operating Company;  Required, unique |
| Supplier | SupplierID | Unique Supplier identifier;  Primary Key, surrogate key |
| Supplier Name | Supplier’s name;  Required |
| Supplier Address | Supplier’s address;  Required, multivalue |
| Order | OrderID | Unique Order identifier;  Primary Key, surrogate key |
| Order Date | Date the order was ordered;  Required |
| Item | Item(s) in a order;  Required, multivalue |
| Item Quantity | Quantity of items ordered;  Required |
| Cost | Cost of items ordered;  Required |

*Relationships*

Purchasers may have zero or many projects, but a project must only have one purchaser.

Projects belong to one or many Operating Companies, an operating company has many projects.

Projects have one or many orders, an order has exactly one project.

Suppliers may have zero or many orders, but an order only has one supplier.

**Logical Normalized Model**

A screenshot of a cell phone

Description automatically generated

*Additional Attributes*

The following list describes which tables were created or gained new attributes that were not in the conceptual model that are now on the logical model.

* Project
  + PurchaserID – Foreign Key
* ProjectOpCo – Join Table
  + ProjectID – Foreign Key
  + OpCOID – Foreign Key
* SupplierAddress – Multivalue Supplier Attribute, made into own table
  + StreetAddress
  + City
  + State
  + ZipCode
  + SupplierID – Foreign Key
* PurchaseOrder
  + ProjectID – Foreign Key
  + Supplier ID – Foreign Key
* OrderItem – Join Table
  + OrderID – Foreign Key
  + ItemID – Foreign Key
* Item - Multivalue Item Attribute, made into own table
  + ItemName
  + ItemCategory
  + ItemDescription
  + ItemUnitofMeasure

*Data Types*

* Primary Keys
  + All primary keys use a surrogate key and are int identity.
* Foreign Keys
  + All foreign keys use int, because all primary key use int identity.
* OrderDate
  + OrderDate on the Order table utilizes datetime because it measures the date orders were placed.
* Cost
  + Cost on the Order table utilizes decimal to not suffer from the loss of rounding cents to a different dollar value, and to maintain the actual amount paid.
* Varchar
  + It was determined to use varchar for all of the remaining attributes because they all fields that use varying lengths of variable characters.

*Reflection*

* What assumptions did you have at the start of your project that changed by the end? Think in terms of both your own problem domain as well as your knowledge of the process.
  + At the start of this project with limited knowledge of databases or SQL, I thought the best approach would be to have the most possible columns related to one another on the same table. Understanding database normalization completely changed how I needed to setup the data within their respective tables and how I approached answering my data questions.
* The next time you do this, what will be different?
  + If I were to attempt to do this project over again, I would change how I prioritized the data. My approach in the project was to start at “Purchaser” and work from there to each corresponding table I needed as a Purchaser. Despite being a Purchaser, the data I really care the most about is in the “OrderItem” Table, so I would organize so that OrderItem was my starting point and work outwards.
* Regardless of whether you go through these steps again, how do you think it will inform your approach to data as an information professional?
  + If I were to never create a database again and I only used a spreadsheet program like Excel to manage data, I still think it would have a benefit to how I approach data. The benefit creating a database had on my approach as an information professional was that it makes you evaluate each column of data and ask does this column belong next to the rest of the columns on the table. Going forward this will reduce and hopefully eliminate any of my data duplication.

*Code*

-- IST 659 Project

/\*

Author: Kyle Welch

12/18/19

IST 659 Project

\*/

-- SQL DDL Statements

/\* Drop tables if they exist,

in reverse order of creation

to avoid any errors with foreign keys

\*/

DROP TABLE IF EXISTS dbo.OrderItem

DROP TABLE IF EXISTS dbo.Item

DROP TABLE IF EXISTS dbo.PurchaseOrder

DROP TABLE IF EXISTS dbo.SupplierAddress

DROP TABLE IF EXISTS dbo.Supplier

DROP TABLE IF EXISTS dbo.ProjectOpCO

DROP TABLE IF EXISTS dbo.OperatingCompany

DROP TABLE IF EXISTS dbo.Project

DROP TABLE IF EXISTS dbo.Purchaser

-- Create the tables

CREATE TABLE Purchaser (

PurchaserID int identity,

FirstName varchar(15) not null,

LastName varchar(15) not null,

Email varchar(30) not null,

PhoneNumber varchar(10) not null,

-- Constraints on the Purchaser table

CONSTRAINT PK\_Purchaser PRIMARY KEY (PurchaserID),

CONSTRAINT U1\_Purchaser UNIQUE (Email),

CONSTRAINT U2\_Purchaser UNIQUE (PhoneNumber)

)

CREATE TABLE Project (

ProjectID int identity,

PurchaserID int not null,

ProjectName varchar(30) not null,

ProjectDescription varchar(100) not null,

-- Constraints on the Project Table

CONSTRAINT PK\_Project PRIMARY KEY (ProjectID),

CONSTRAINT FK1\_Project FOREIGN KEY (PurchaserID) REFERENCES Purchaser(PurchaserID)

)

CREATE TABLE OperatingCompany (

OpCoID int identity,

OpCoName varchar(20) not null,

-- Constraints on the OperatingCompany Table

CONSTRAINT PK\_OperatingCompany PRIMARY KEY (OpCoID),

CONSTRAINT U1\_OperatingCompany UNIQUE (OpCoName)

)

CREATE TABLE ProjectOpCo (

ProjectOpCoID int identity,

ProjectID int not null,

OpCoID int not null,

-- Constraints on the ProjectOpCo Table

CONSTRAINT PK\_ProjectOpCo PRIMARY KEY (ProjectOpCoID),

CONSTRAINT FK1\_ProjectOpCo FOREIGN KEY (ProjectID) REFERENCES Project(ProjectID),

CONSTRAINT FK2\_ProjectOpCo FOREIGN KEY (OpCoID) REFERENCES OperatingCompany(OpCoID)

)

CREATE TABLE Supplier (

SupplierID int identity,

SupplierName varchar(30) not null,

-- Constraints on the Supplier Table

CONSTRAINT PK\_Supplier PRIMARY KEY (SupplierID)

)

CREATE TABLE SupplierAddress (

SupplierAddressID int identity,

SupplierID int not null,

StreetAddress varchar(30) not null,

City varchar(30) not null,

State varchar(30) not null,

ZipCode varchar(10) not null,

-- Constraints on the SupplierAddress

CONSTRAINT PK\_SupplierAddress PRIMARY KEY (SupplierAddressID),

CONSTRAINT FK1\_SupplierAddress FOREIGN KEY (SupplierID) REFERENCES Supplier(SupplierID)

)

CREATE TABLE PurchaseOrder (

PurchaseOrderID int identity,

ProjectID int not null,

SupplierID int not null,

OrderDate datetime not null,

-- Constraints on the PurchaseOrder Table

CONSTRAINT PK\_PurchaseOrder PRIMARY KEY (PurchaseOrderID),

CONSTRAINT FK1\_PurchaseOrder FOREIGN KEY (ProjectID) REFERENCES Project(ProjectID),

CONSTRAINT FK2\_PurchaseOrder FOREIGN KEY (SupplierID) REFERENCES Supplier(SupplierID)

)

CREATE TABLE Item (

ItemID int identity,

ItemName varchar(20) not null,

ItemCategory varchar(20) not null,

ItemDescription varchar(100) not null,

ItemUnitOfMeasure varchar(10) not null,

-- Constraints on the Item Table

CONSTRAINT PK\_Item PRIMARY KEY (ItemID)

)

CREATE TABLE OrderItem (

OrderItemID int identity,

PurchaseOrderID int not null,

ItemID int not null,

ItemQuantity int not null,

Cost decimal not null,

-- Constraints on the OrderItem Table

CONSTRAINT PK\_OrderItem PRIMARY KEY (OrderItemID),

CONSTRAINT FK1\_OrderItem FOREIGN KEY (PurchaseOrderID) REFERENCES PurchaseOrder(PurchaseOrderID),

CONSTRAINT FK2\_OrderItem FOREIGN KEY (ItemID) REFERENCES Item(ItemID)

)

-- Insert statements for Purchaser Table

INSERT INTO Purchaser (FirstName, LastName, Email, PhoneNumber) VALUES ('Aurie', 'Tomley', 'atomley0@tripod.com', '5588215627');

INSERT INTO Purchaser (FirstName, LastName, Email, PhoneNumber) VALUES ('Judye', 'Skellington', 'jskellington1@ning.com', '5833567854');

INSERT INTO Purchaser (FirstName, LastName, Email, PhoneNumber) VALUES ('Valentino', 'Swait', 'vswait2@vkontakte.ru', '3952161396');

INSERT INTO Purchaser (FirstName, LastName, Email, PhoneNumber) VALUES ('Dona', 'Wendover', 'dwendover3@domainmarket.com', '4295845475');

INSERT INTO Purchaser (FirstName, LastName, Email, PhoneNumber) VALUES ('Donielle', 'Curl', 'dcurl4@columbia.edu', '7714033391');

INSERT INTO Purchaser (FirstName, LastName, Email, PhoneNumber) VALUES ('Ambrosi', 'Titmus', 'atitmus5@amazon.de', '2626985571');

INSERT INTO Purchaser (FirstName, LastName, Email, PhoneNumber) VALUES ('Isak', 'Skipsea', 'iskipsea6@github.com', '4448355185');

INSERT INTO Purchaser (FirstName, LastName, Email, PhoneNumber) VALUES ('Edik', 'Eburah', 'eeburah7@sogou.com', '1566186760');

INSERT INTO Purchaser (FirstName, LastName, Email, PhoneNumber) VALUES ('Kendra', 'McGrouther', 'kmcgrouther8@123-reg.co.uk', '9864800192');

INSERT INTO Purchaser (FirstName, LastName, Email, PhoneNumber) VALUES ('Todd', 'Tremmil', 'ttremmil9@yandex.ru', '1338783781');

-- Insert staements for Project Table

INSERT INTO Project (PurchaserID, ProjectName, ProjectDescription) VALUES (1, 'benchmark next-generation', 'Aenean lectus.');

INSERT INTO Project (PurchaserID, ProjectName, ProjectDescription) VALUES (2, 'wireless interfaces', 'Donec pharetra, magna vestibulum aliquet ultrices, erat tortor sollicitudin mi.');

INSERT INTO Project (PurchaserID, ProjectName, ProjectDescription) VALUES (3, 'disintermediate systems', 'Sed ante. Vivamus tortor. Duis mattis egestas metus.');

INSERT INTO Project (PurchaserID, ProjectName, ProjectDescription) VALUES (4, 'synergize integrated vortals', 'Vivamus in felis eu sapien cursus vestibulum.');

INSERT INTO Project (PurchaserID, ProjectName, ProjectDescription) VALUES (5, 'monetize vortals', 'Maecenas leo odio, condimentum id, luctus nec, molestie sed, justo.');

INSERT INTO Project (PurchaserID, ProjectName, ProjectDescription) VALUES (6, 'redefine vortals', 'In blandit ultrices enim. Lorem ipsum dolor sit amet.');

INSERT INTO Project (PurchaserID, ProjectName, ProjectDescription) VALUES (7, 'whiteboard front-end e-tailers', 'Donec posuere metus vitae ipsum.');

INSERT INTO Project (PurchaserID, ProjectName, ProjectDescription) VALUES (8, 'reinvent wireless vortals', 'Proin eu mi. Nulla ac enim.');

INSERT INTO Project (PurchaserID, ProjectName, ProjectDescription) VALUES (9, '24/7 synergies', 'Vivamus vel nulla eget eros elementum pellentesque. Quisque porta volutpat erat.');

INSERT INTO Project (PurchaserID, ProjectName, ProjectDescription) VALUES (10, 'utilize visionary metrics', 'Morbi non quam nec dui luctus rutrum. Nulla tellus. In sagittis dui vel nisl.');

-- Insert statements for OperatingCompany Table

INSERT INTO OperatingCompany (OpCoName) VALUES ('ABC');

INSERT INTO OperatingCompany (OpCoName) VALUES ('CTEG');

INSERT INTO OperatingCompany (OpCoName) VALUES ('CNYP');

INSERT INTO OperatingCompany (OpCoName) VALUES ('MGE');

INSERT INTO OperatingCompany (OpCoName) VALUES ('XYZ');

-- Insert statements for ProjectOpCo Table

INSERT INTO ProjectOpCo (ProjectID, OpCoID) VALUES (1, 1);

INSERT INTO ProjectOpCo (ProjectID, OpCoID) VALUES (2, 2);

INSERT INTO ProjectOpCo (ProjectID, OpCoID) VALUES (3, 3);

INSERT INTO ProjectOpCo (ProjectID, OpCoID) VALUES (4, 4);

INSERT INTO ProjectOpCo (ProjectID, OpCoID) VALUES (5, 5);

INSERT INTO ProjectOpCo (ProjectID, OpCoID) VALUES (6, 1);

INSERT INTO ProjectOpCo (ProjectID, OpCoID) VALUES (7, 2);

INSERT INTO ProjectOpCo (ProjectID, OpCoID) VALUES (8, 3);

INSERT INTO ProjectOpCo (ProjectID, OpCoID) VALUES (9, 4);

INSERT INTO ProjectOpCo (ProjectID, OpCoID) VALUES (10, 5);

-- Insert statements for Supplier Table

INSERT INTO Supplier (SupplierName) VALUES ('Murphy, Schiller and Durgan');

INSERT INTO Supplier (SupplierName) VALUES ('Littel, DuBuque and Mosciski');

INSERT INTO Supplier (SupplierName) VALUES ('Steuber Inc');

INSERT INTO Supplier (SupplierName) VALUES ('Fay, Larson and Reichert');

INSERT INTO Supplier (SupplierName) VALUES ('Parker-Blick');

-- Insert statements for SupplierAddress Table

INSERT INTO SupplierAddress (SupplierID, StreetAddress, City, State, ZipCode) VALUES (1, '38 Trailsway Crossing', 'Des Moines', 'Iowa', '50369');

INSERT INTO SupplierAddress (SupplierID, StreetAddress, City, State, ZipCode) VALUES (2, '33973 Buhler Circle', 'Oklahoma City', 'Oklahoma', '73152');

INSERT INTO SupplierAddress (SupplierID, StreetAddress, City, State, ZipCode) VALUES (3, '3 Mccormick Junction', 'Chattanooga', 'Tennessee', '37410');

INSERT INTO SupplierAddress (SupplierID, StreetAddress, City, State, ZipCode) VALUES (4, '488 Del Sol Avenue', 'New York City', 'New York', '10004');

INSERT INTO SupplierAddress (SupplierID, StreetAddress, City, State, ZipCode) VALUES (5, '9851 Riverside Pass', 'Peoria', 'Arizona', '85383');

-- Insert statements for PurchaseOrder Table

INSERT INTO PurchaseOrder (ProjectID, SupplierID, OrderDate) VALUES (1, 1, '2019-04-12 10:22:16');

INSERT INTO PurchaseOrder (ProjectID, SupplierID, OrderDate) VALUES (2, 2, '2019-10-27 13:59:40');

INSERT INTO PurchaseOrder (ProjectID, SupplierID, OrderDate) VALUES (3, 3, '2019-04-11 07:24:45');

INSERT INTO PurchaseOrder (ProjectID, SupplierID, OrderDate) VALUES (4, 4, '2019-07-26 10:05:01');

INSERT INTO PurchaseOrder (ProjectID, SupplierID, OrderDate) VALUES (5, 5, '2019-05-03 16:43:30');

INSERT INTO PurchaseOrder (ProjectID, SupplierID, OrderDate) VALUES (6, 1, '2019-08-05 03:03:50');

INSERT INTO PurchaseOrder (ProjectID, SupplierID, OrderDate) VALUES (7, 2, '2019-06-19 20:51:48');

INSERT INTO PurchaseOrder (ProjectID, SupplierID, OrderDate) VALUES (8, 3, '2019-01-21 21:11:07');

INSERT INTO PurchaseOrder (ProjectID, SupplierID, OrderDate) VALUES (9, 4, '2019-02-05 06:48:18');

INSERT INTO PurchaseOrder (ProjectID, SupplierID, OrderDate) VALUES (10, 5, '2019-02-17 02:27:54');

-- Insert statements for Item Table

INSERT INTO Item (ItemName, ItemCategory, ItemDescription, ItemUnitOfMeasure) VALUES ('Quisque Arcu', 'Research', 'Pellentesque ultrices mattis odio.', 'Per SqFt');

INSERT INTO Item (ItemName, ItemCategory, ItemDescription, ItemUnitOfMeasure) VALUES ('Id Consequat', 'Legal', 'Quisque id justo sit amet sapien dignissim vestibulum. Nulla dapibus dolor vel est.', 'Per Item');

INSERT INTO Item (ItemName, ItemCategory, ItemDescription, ItemUnitOfMeasure) VALUES ('Libero Rutrum', 'Legal', 'Nulla tellus.', 'Per Lb');

INSERT INTO Item (ItemName, ItemCategory, ItemDescription, ItemUnitOfMeasure) VALUES ('Mattis Odio', 'Accounting', 'Pellentesque ultrices mattis odio.', 'Per Item');

INSERT INTO Item (ItemName, ItemCategory, ItemDescription, ItemUnitOfMeasure) VALUES ('Diam', 'Services', 'Sed sagittis.', 'Per SqFt');

-- Insert statements for OrderItem Table

INSERT INTO OrderItem (PurchaseOrderID, ItemID, ItemQuantity, Cost) VALUES (1, 1, 256, 1024.28);

INSERT INTO OrderItem (PurchaseOrderID, ItemID, ItemQuantity, Cost) VALUES (2, 2, 708, 2625.31);

INSERT INTO OrderItem (PurchaseOrderID, ItemID, ItemQuantity, Cost) VALUES (2, 3, 265, 142.1);

INSERT INTO OrderItem (PurchaseOrderID, ItemID, ItemQuantity, Cost) VALUES (3, 2, 465, 3296.17);

INSERT INTO OrderItem (PurchaseOrderID, ItemID, ItemQuantity, Cost) VALUES (3, 3, 842, 48.17);

INSERT INTO OrderItem (PurchaseOrderID, ItemID, ItemQuantity, Cost) VALUES (4, 5, 165, 1599.12);

INSERT INTO OrderItem (PurchaseOrderID, ItemID, ItemQuantity, Cost) VALUES (5, 1, 811, 1122.55);

INSERT INTO OrderItem (PurchaseOrderID, ItemID, ItemQuantity, Cost) VALUES (6, 3, 488, 53.05);

INSERT INTO OrderItem (PurchaseOrderID, ItemID, ItemQuantity, Cost) VALUES (7, 4, 247, 1762.49);

INSERT INTO OrderItem (PurchaseOrderID, ItemID, ItemQuantity, Cost) VALUES (8, 5, 367, 5610.68);

INSERT INTO OrderItem (PurchaseOrderID, ItemID, ItemQuantity, Cost) VALUES (9, 2, 148, 2610.28);

INSERT INTO OrderItem (PurchaseOrderID, ItemID, ItemQuantity, Cost) VALUES (9, 3, 412, 56.34);

INSERT INTO OrderItem (PurchaseOrderID, ItemID, ItemQuantity, Cost) VALUES (10, 4, 967, 614.85);

-- Answers to Data questions

-- How many orders include a specific item?

GO

CREATE FUNCTION dbo.OrderItemCount(@ItemID int)

RETURNS int AS

BEGIN

DECLARE @returnValue int

SELECT @returnValue = COUNT(ItemID) FROM OrderItem

WHERE OrderItem.ItemID = @ItemID

RETURN @returnValue

END

GO

CREATE VIEW FrequentlyOrderedItems AS

SELECT TOP 5

\*

, dbo.OrderItemCount(ItemID) as OrderCount

FROM Item

ORDER BY OrderCount DESC

GO

SELECT \* FROM FrequentlyOrderedItems

GO

-- How many projects belong to each OpCo?

GO

CREATE FUNCTION dbo.ProjectOpCoCount(@OpCoID int)

RETURNS int AS

BEGIN

DECLARE @returnValue int

SELECT @returnValue = COUNT(OpCoID) FROM ProjectOpCo

WHERE ProjectOpCo.OpCoID = @OpCoID

RETURN @returnValue

END

GO

CREATE VIEW ProjectsPerOpCo AS

SELECT TOP 5

\*

, dbo.ProjectOpCoCount(OpCoID) as ProjectCount

FROM OperatingCompany

ORDER BY ProjectCount DESC

GO

SELECT \* FROM ProjectsPerOpCo

GO

-- What are the top 10 Purchase Orders by total cost?

GO

CREATE FUNCTION dbo.PurchaseOrderCost(@purchaseorderID int)

RETURNS int AS

BEGIN

DECLARE @returnValue int

SELECT @returnValue = SUM(Cost) From OrderItem

WHERE OrderItem.PurchaseOrderID = @purchaseorderID

RETURN @returnValue

END

GO

CREATE VIEW OrderTotalCost AS

SELECT TOP 10

\*

,dbo.PurchaseOrderCost(PurchaseOrderID) AS TotalCost

FROM PurchaseOrder

ORDER BY TotalCost DESC

GO

SELECT \* FROM OrderTotalCost

GO

-- What is the average cost for each item that has been purchased?

GO

CREATE FUNCTION dbo.AVGItemCost(@ItemID int)

RETURNS int AS

BEGIN

DECLARE @returnValue int

SELECT @returnValue = AVG(Cost) FROM OrderItem

WHERE OrderItem.ItemID = @itemID

RETURN @returnValue

END

GO

CREATE VIEW AverageCost AS

SELECT TOP 5

\*

, dbo.AVGItemCost(ItemID) AS AverageItemCost

FROM Item

ORDER BY AverageItemCost DESC

GO

SELECT \* FROM AverageCost

GO

-- How will new Purchasers be added to the database

GO

CREATE PROCEDURE AddPurchaser(@FirstName varchar(15), @LastName varchar(15), @Email varchar(30), @PhoneNumber varchar(10))

AS

BEGIN

INSERT INTO Purchaser (FirstName, LastName, Email, PhoneNumber)

VALUES (@FirstName, @LastName, @Email, @PhoneNumber)

RETURN @@identity

END

GO

DECLARE @newPurchaserID int

EXEC @newPurchaserID = AddPurchaser 'Miles', 'Bridges', 'MBridges@MSU.edu', '9451308221'

SELECT \* FROM Purchaser WHERE PurchaserID = @newPurchaserID

GO

-- How will projects be assigned to a Purchaser?

CREATE PROCEDURE ReassignProject(@ProjectID int)

As

BEGIN

UPDATE Project SET PurchaserID ='6'

WHERE ProjectID = @ProjectID

END

GO

DECLARE @newProject int

INSERT INTO Project

(PurchaserID, ProjectName, ProjectDescription)

VALUES(

(SELECT PurchaserID FROM Purchaser WHERE Email = 'atitmus5@amazon.de')

, '2020 Project'

, 'New Project starting in 2020'

)

SET @newProject = @@identity

EXEC ReassignProject @newProject

SELECT \* FROM Project WHERE ProjectID = 11

GO

-- DROPs for each Function, View, and StoredProcedure

DROP FUNCTION dbo.OrderItemCount

GO

DROP VIEW FrequentlyOrderedItems

GO

DROP FUNCTION dbo.ProjectOpCoCount

GO

DROP VIEW ProjectPerOpCo

GO

DROP FUNCTION dbo.PurchaseOrderCost

GO

DROP VIEW OrderTotalCost

GO

Drop FUNCTION dbo.AVGItemCost

GO

DROP VIEW AverageCost

GO

DROP PROCEDURE AddPurchaser

GO

DROP PROCEDURE ReassignProject

GO