Bomin K. Powers

bkpowers@syr.edu

IST659\_Lab08

Database Programming

Due date of assignment:6/3/2021

Submitted date: 6/2/2021

-- drop all procedures

DROP PROCEDURE IF EXISTS dbo.rp\_pro\_AddMealModelTypeAmount\_BARF;

go

DROP PROCEDURE IF EXISTS dbo.rp\_pro\_AddIngredientIntoStore;

go

DROP TRIGGER IF EXISTS dbo.trg\_upd\_IngredientStock;

go

DROP TRIGGER IF EXISTS dbo.trg\_upd2\_IngredientStock;

go

DROP PROCEDURE IF EXISTS dbo.rp\_pro\_AddIngredientIntoStock;

go

DROP PROCEDURE IF EXISTS dbo.rp\_pro\_AddNutritionPart;

go

DROP PROCEDURE IF EXISTS dbo.rp\_pro\_AddNutritionSource;

go

DROP PROCEDURE IF EXISTS dbo.rp\_pro\_AddPetHealthinfo;

go

DROP PROCEDURE IF EXISTS dbo.rp\_pro\_AddPet;

go

DROP PROCEDURE IF EXISTS dbo.rp\_pro\_AddStock;

go

DROP PROCEDURE IF EXISTS dbo.rp\_pro\_AddOwner;

go

-- drop all views

DROP VIEW IF EXISTS dbo.rp\_v\_StockReport;

go

DROP VIEW IF EXISTS dbo.rp\_v\_PetPreferenceCount;

go

DROP VIEW IF EXISTS dbo.rp\_v\_CompletePlans;

Go

-- drop all tables in reverse order of their dependencies

DROP TABLE IF EXISTS dbo.rp\_MealModelPartAmount;

go

DROP TABLE IF EXISTS dbo.rp\_CompleteMealPlan;

go

DROP TABLE IF EXISTS dbo.rp\_MealModelTypeAmount;

go

DROP TABLE IF EXISTS dbo.rp\_PetMealModel;

go

DROP TABLE IF EXISTS dbo.rp\_MealModelRatio;

go

DROP TABLE IF EXISTS dbo.rp\_MealModel;

go

DROP TABLE IF EXISTS dbo.rp\_IngredientStore;

go

DROP TABLE IF EXISTS dbo.rp\_Store;

go

DROP TABLE IF EXISTS dbo.rp\_IngredientStockUpdate;

go

DROP TABLE IF EXISTS dbo.rp\_IngredientStock;

go

DROP TABLE IF EXISTS dbo.rp\_Stock;

go

DROP TABLE IF EXISTS dbo.rp\_PetPreference;

go

DROP TABLE IF EXISTS dbo.rp\_IngredientNutritionFacts;

go

DROP TABLE IF EXISTS dbo.rp\_IngredientDetail;

go

DROP TABLE IF EXISTS dbo.rp\_IngredientType;

go

DROP TABLE IF EXISTS dbo.rp\_PetHealthinfo;

go

DROP TABLE IF EXISTS dbo.rp\_Pet\_ActivityRate;

go

DROP TABLE IF EXISTS dbo.rp\_Pet\_WeightCategory;

go

DROP TABLE IF EXISTS dbo.rp\_Pet;

go

DROP TABLE IF EXISTS dbo.rp\_Owner;

go

-- create all tables in order of their dependencies

CREATE TABLE rp\_Owner

(

rp\_OwnerID int NOT NULL IDENTITY

, OwnerFirstName varchar(30) NOT NULL

, OwnerLastName varchar(30) NOT NULL

, EmailAddress varchar(50) NOT NULL

-- place constraints

, CONSTRAINT rp\_Owner\_PK PRIMARY KEY (rp\_OwnerID)

, CONSTRAINT rp\_Owner\_U1 UNIQUE (EmailAddress)

);

CREATE TABLE rp\_Pet

(

rp\_PetID int NOT NULL IDENTITY

, PetName varchar(20) NOT NULL

, PetType varchar(10) NOT NULL

, DateOfBirth datetime NOT NULL

, rp\_OwnerID int NOT NULL

-- place constraints

, CONSTRAINT rp\_Pet\_PK PRIMARY KEY (rp\_PetID)

, CONSTRAINT rp\_Pet\_FK1 FOREIGN KEY (rp\_OwnerID) REFERENCES rp\_Owner (rp\_OwnerID)

-- limits values for PetType column

, CONSTRAINT rp\_Pet\_chk\_val CHECK (PetType in ('Dog','Cat'))

);

CREATE TABLE rp\_Pet\_WeightCategory

(

rp\_Pet\_WeightCategoryID int NOT NULL IDENTITY

, WeightCategory varchar(20) NOT NULL

-- place constraints

, CONSTRAINT rp\_Pet\_WeightCategory\_PK PRIMARY KEY (rp\_Pet\_WeightCategoryID)

);

CREATE TABLE rp\_Pet\_ActivityRate

(

rp\_Pet\_ActivityRateID int NOT NULL IDENTITY

, ActivityRate varchar(40) NOT NULL

-- place constraints

, CONSTRAINT rp\_Pet\_ActivityRate\_PK PRIMARY KEY (rp\_Pet\_ActivityRateID)

);

CREATE TABLE rp\_PetHealthinfo

(

rp\_PetHealthID int NOT NULL IDENTITY

, PetWeight decimal(10,2) NOT NULL

, rp\_Pet\_WeightCategoryID int NOT NULL

, rp\_Pet\_ActivityRateID int NOT NULL

, Spayed varchar(5) NOT NULL

, rp\_PetID int NOT NULL

, PetRequiredCalorie int NOT NULL

-- place constraints

, CONSTRAINT rp\_PetHealthinfo\_PK PRIMARY KEY (rp\_PetHealthID)

, CONSTRAINT rp\_PetHealthinfo\_FK1 FOREIGN KEY (rp\_PetID) REFERENCES rp\_Pet (rp\_PetID)

, CONSTRAINT rp\_PetHealthinfo\_FK2 FOREIGN KEY (rp\_Pet\_WeightCategoryID) REFERENCES rp\_Pet\_WeightCategory (rp\_Pet\_WeightCategoryID)

, CONSTRAINT rp\_PetHealthinfo\_FK3 FOREIGN KEY (rp\_Pet\_ActivityRateID) REFERENCES rp\_Pet\_ActivityRate (rp\_Pet\_ActivityRateID)

-- limits values for Spayed column

, CONSTRAINT rp\_PetHealthinfo\_chk\_val3 CHECK (Spayed in ('Yes','No'))

);

CREATE TABLE rp\_IngredientType

(

rp\_IngredientTypeID int NOT NULL IDENTITY

, IngredientType varchar(20) NOT NULL

-- place constraints

, CONSTRAINT rp\_IngredientType\_PK PRIMARY KEY (rp\_IngredientTypeID)

, CONSTRAINT rp\_IngredientType\_U1 UNIQUE (IngredientType)

);

CREATE TABLE rp\_IngredientDetail

(

rp\_IngredientDetailID int NOT NULL IDENTITY

, rp\_IngredientTypeID int NOT NULL

, IngredientSource varchar(20) NOT NULL

, IngredientPart varchar(20)

-- place constraints

, CONSTRAINT rp\_IngredientDetail\_PK PRIMARY KEY (rp\_IngredientDetailID)

, CONSTRAINT rp\_IngredientDetail\_FK1 FOREIGN KEY (rp\_IngredientTypeID) REFERENCES rp\_IngredientType (rp\_IngredientTypeID)

);

CREATE TABLE rp\_IngredientNutritionFacts

(

rp\_IngredientNutritionFactsID int NOT NULL IDENTITY

, rp\_IngredientDetailID int NOT NULL

, IngredientCalorie int NOT NULL

, IngredientFat int

, IngredientProtein int

, IngredientCarbs int

, IngredientFiber int

, IngredientSugar int

-- place constraints

, CONSTRAINT rp\_IngredientNutritionFacts\_PK PRIMARY KEY (rp\_IngredientNutritionFactsID)

, CONSTRAINT rp\_IngredientNutritionFacts\_FK1 FOREIGN KEY (rp\_IngredientDetailID) REFERENCES rp\_IngredientDetail (rp\_IngredientDetailID)

);

CREATE TABLE rp\_PetPreference

(

rp\_PetPreferenceID int NOT NULL IDENTITY

, rp\_PetID int NOT NULL

, rp\_IngredientDetailID int NOT NULL

-- place constraints

, CONSTRAINT rp\_PetPreference\_PK PRIMARY KEY (rp\_PetPreferenceID)

, CONSTRAINT rp\_PetPreference\_FK1 FOREIGN KEY (rp\_PetID) REFERENCES rp\_Pet (rp\_PetID)

, CONSTRAINT rp\_PetPreference\_FK2 FOREIGN KEY (rp\_IngredientDetailID) REFERENCES rp\_IngredientDetail (rp\_IngredientDetailID)

);

CREATE TABLE rp\_Stock

(

rp\_StockID int NOT NULL IDENTITY

, StorageName varchar(30) NOT NULL

, rp\_OwnerID int NOT NULL

-- place constraints

, CONSTRAINT rp\_Stock\_PK PRIMARY KEY (rp\_StockID)

, CONSTRAINT rp\_Stock\_FK1 FOREIGN KEY (rp\_OwnerID) REFERENCES rp\_Owner (rp\_OwnerID)

);

CREATE TABLE rp\_IngredientStock

(

rp\_IngredientStockID int NOT NULL IDENTITY

, rp\_StockID int NOT NULL

, rp\_IngredientDetailID int NOT NULL

, IngredientAmount int NOT NULL

, StoredDate datetime NOT NULL default GETDATE()

, Finished varchar(3)

-- place constraints

, CONSTRAINT rp\_IngredientStock\_PK PRIMARY KEY (rp\_IngredientStockID)

, CONSTRAINT rp\_IngredientStock\_FK1 FOREIGN KEY (rp\_StockID) REFERENCES rp\_Stock (rp\_StockID)

, CONSTRAINT rp\_IngredientStock\_FK2 FOREIGN KEY (rp\_IngredientDetailID) REFERENCES rp\_IngredientDetail (rp\_IngredientDetailID)

);

CREATE TABLE rp\_IngredientStockUpdate

(

rp\_IngredientStockUpdateID int NOT NULL IDENTITY

, rp\_IngredientStockID int NOT NULL

, UpdatedDate datetime default GETDATE() NOT NULL

-- place constraints

, CONSTRAINT rp\_IngredientStockUpdate\_PK PRIMARY KEY (rp\_IngredientStockUpdateID)

, CONSTRAINT rp\_IngredientStockUpdate\_FK1 FOREIGN KEY (rp\_IngredientStockID) REFERENCES rp\_IngredientStock (rp\_IngredientStockID)

);

CREATE TABLE rp\_Store

(

rp\_StoreID int NOT NULL IDENTITY

, StoreName varchar(30) NOT NULL

-- place constraints

, CONSTRAINT rp\_Store\_PK PRIMARY KEY (rp\_StoreID)

);

CREATE TABLE rp\_IngredientStore

(

rp\_IngredientStoreID int NOT NULL IDENTITY

, rp\_StoreID int NOT NULL

, rp\_IngredientDetailID int NOT NULL

, IngredientPricePer100g decimal(10,2) NOT NULL

-- place constraints

, CONSTRAINT rp\_IngredientStore\_PK PRIMARY KEY (rp\_IngredientStoreID)

, CONSTRAINT rp\_IngredientStore\_FK1 FOREIGN KEY (rp\_StoreID) REFERENCES rp\_Store (rp\_StoreID)

, CONSTRAINT rp\_IngredientStore\_FK2 FOREIGN KEY (rp\_IngredientDetailID) REFERENCES rp\_IngredientDetail (rp\_IngredientDetailID)

, CONSTRAINT rp\_IngredientStore\_U1 UNIQUE (rp\_IngredientDetailID)

);

CREATE TABLE rp\_MealModel

(

rp\_MealModelID int NOT NULL IDENTITY

, MealModelName varchar(10) NOT NULL

, MealModelDescription varchar(50)

-- place constraints

, CONSTRAINT rp\_MealModel\_PK PRIMARY KEY (rp\_MealModelID)

);

CREATE TABLE rp\_MealModelRatio

(

rp\_MealModelRatioID int NOT NULL IDENTITY

, rp\_MealModelID int NOT NULL

, rp\_IngredientTypeID int NOT NULL

, MealModelTypeRatio decimal(10,2) NOT NULL

-- place constraints

, CONSTRAINT rp\_MealModelRatio\_PK PRIMARY KEY (rp\_MealModelRatioID)

, CONSTRAINT rp\_MealModelRatio\_FK1 FOREIGN KEY (rp\_MealModelID) REFERENCES rp\_MealModel (rp\_MealModelID)

, CONSTRAINT rp\_MealModelRatio\_FK2 FOREIGN KEY (rp\_IngredientTypeID) REFERENCES rp\_IngredientType (rp\_IngredientTypeID)

);

CREATE TABLE rp\_PetMealModel

(

rp\_PetMealModelID int NOT NULL IDENTITY

, rp\_PetID int NOT NULL

, rp\_MealModelID int NOT NULL

-- place constraints

, CONSTRAINT rp\_PetMealModel\_PK PRIMARY KEY (rp\_PetMealModelID)

, CONSTRAINT rp\_PetMealModel\_FK1 FOREIGN KEY (rp\_PetID) REFERENCES rp\_Pet (rp\_PetID)

, CONSTRAINT rp\_PetMealModel\_FK2 FOREIGN KEY (rp\_MealModelID) REFERENCES rp\_MealModel (rp\_MealModelID)

);

CREATE TABLE rp\_MealModelTypeAmount

(

rp\_MealModelTypeAmountID int NOT NULL IDENTITY

, rp\_PetMealModelID int NOT NULL

, rp\_IngredientTypeID int NOT NULL

, MealModelTypeAmount int NOT NULL

-- place constraints

, CONSTRAINT rp\_MealModelTypeAmount\_PK PRIMARY KEY (rp\_MealModelTypeAmountID)

, CONSTRAINT rp\_MealModelTypeAmount\_FK1 FOREIGN KEY (rp\_PetMealModelID) REFERENCES rp\_PetMealModel (rp\_PetMealModelID)

, CONSTRAINT rp\_MealModelTypeAmount\_FK2 FOREIGN KEY (rp\_IngredientTypeID) REFERENCES rp\_IngredientType (rp\_IngredientTypeID)

);

CREATE TABLE rp\_CompleteMealPlan

(

rp\_CompleteMealPlanID int NOT NULL IDENTITY

, rp\_PetID int NOT NULL

, SuggestedDate datetime NOT NULL default GETDATE()

-- place constraints

, CONSTRAINT rp\_CompleteMealPlan\_PK PRIMARY KEY (rp\_CompleteMealPlanID)

, CONSTRAINT rp\_CompleteMealPlan\_FK1 FOREIGN KEY (rp\_PetID) REFERENCES rp\_Pet (rp\_PetID)

);

GO

CREATE TABLE rp\_MealModelPartAmount

(

rp\_MealModelPartAmountID int NOT NULL IDENTITY

, rp\_MealModelTypeAmountID int NOT NULL

, rp\_IngredientDetailID int NOT NULL

, MealModelPartAmount int NOT NULL

, rp\_CompleteMealPlanID int NOT NULL

-- place constraints

, CONSTRAINT rp\_MealModelPartAmount\_PK PRIMARY KEY (rp\_MealModelPartAmountID)

, CONSTRAINT rp\_MealModelPartAmount\_FK1 FOREIGN KEY (rp\_MealModelTypeAmountID) REFERENCES rp\_MealModelTypeAmount (rp\_MealModelTypeAmountID)

, CONSTRAINT rp\_MealModelPartAmount\_FK2 FOREIGN KEY (rp\_IngredientDetailID) REFERENCES rp\_IngredientDetail (rp\_IngredientDetailID)

, CONSTRAINT rp\_MealModelPartAmount\_FK3 FOREIGN KEY (rp\_CompleteMealPlanID) REFERENCES rp\_CompleteMealPlan (rp\_CompleteMealPlanID)

);

GO

-- create procedures

-- Add Owner procedures

CREATE OR ALTER PROCEDURE rp\_pro\_AddOwner (@firstName varchar(30), @lastName varchar(30), @email varchar(50))

AS

-- clarify there is same Email Address value

IF EXISTS

(

SELECT \* FROM rp\_Owner WHERE EmailAddress = @email

)

BEGIN

PRINT 'Unique value(=EmailAddress) existed'

RETURN (SELECT rp\_OwnerID FROM rp\_Owner WHERE EmailAddress = @email);

END

ELSE

BEGIN

INSERT INTO rp\_Owner (OwnerFirstName, OwnerLastName, EmailAddress)

VALUES (@firstName, @lastName, @email)

PRINT 'Inserted Owner'

RETURN SCOPE\_IDENTITY()

END

GO

-- Add Stock procedures

CREATE OR ALTER PROCEDURE rp\_pro\_AddStock (@email varchar(50), @storageName varchar(30))

AS

BEGIN

-- We have a owner email address but we need the owner ID

-- Declare a variable to hold the ID

DECLARE @ownerID int

SELECT @ownerID = rp\_OwnerID FROM rp\_Owner

WHERE EmailAddress = @email

-- Add the row

INSERT INTO rp\_Stock(rp\_OwnerID, StorageName)

VALUES (@ownerID, @storageName)

RETURN SCOPE\_IDENTITY()

END

GO

-- Add Pet procedures

CREATE OR ALTER PROCEDURE rp\_pro\_AddPet (@email varchar(50), @petName varchar(20), @petType varchar(10), @DOB datetime)

AS

BEGIN

-- We have a owner email address but we need the owner ID

-- Declare a variable to hold the ID

DECLARE @ownerID int

SELECT @ownerID = rp\_OwnerID FROM rp\_Owner

WHERE EmailAddress = @email

-- Add the row

INSERT INTO rp\_Pet (rp\_OwnerID, PetName, PetType, DateOfBirth)

VALUES (@ownerID, @petName, @petType, @DOB)

RETURN SCOPE\_IDENTITY()

END

GO

-- Add Pet health info procedures

CREATE OR ALTER PROCEDURE rp\_pro\_AddPetHealthinfo (@petName varchar(20), @weight int, @weightCT int, @activity int, @spayed varchar(5), @cal int)

AS

BEGIN

-- We have a pet name but we need the pet ID

-- Declare a variable to hold the ID

DECLARE @petID int

SELECT @petID = rp\_PetID FROM rp\_Pet

WHERE PetName = @petName

-- Add the row

INSERT INTO rp\_PetHealthinfo (rp\_PetID, PetWeight, rp\_Pet\_WeightCategoryID, rp\_Pet\_ActivityRateID, Spayed, PetRequiredCalorie)

VALUES (@petID, @weight, @weightCT, @activity, @spayed, @cal)

RETURN SCOPE\_IDENTITY()

END

GO

-- Add ingredient nutrition (type,source) procedures

CREATE OR ALTER PROCEDURE rp\_pro\_AddNutritionSource (@type varchar(20), @source varchar(20), @calorie int, @protein int, @fat int, @carbs int, @fiber int, @sugar int)

AS

BEGIN

DECLARE @typeID int

SELECT @typeID = rp\_IngredientTypeID FROM rp\_IngredientType

WHERE IngredientType = @type

DECLARE @detailID int

SELECT @detailID = rp\_IngredientDetailID FROM rp\_IngredientDetail

WHERE rp\_IngredientTypeID = @typeID AND IngredientSource = @source

-- Add the row

INSERT INTO rp\_IngredientNutritionFacts(rp\_IngredientDetailID, IngredientCalorie, IngredientProtein, IngredientFat, IngredientCarbs, IngredientFiber, IngredientSugar)

VALUES (@detailID, @calorie, @protein, @fat, @carbs, @fiber, @sugar)

RETURN SCOPE\_IDENTITY()

END

GO

-- Add ingredient nutrition (type,source,part) procedures

CREATE OR ALTER PROCEDURE rp\_pro\_AddNutritionPart (@type varchar(20), @source varchar(20), @part varchar(20), @calorie int, @protein int, @fat int, @carbs int, @fiber int, @sugar int)

AS

BEGIN

DECLARE @typeID int

SELECT @typeID = rp\_IngredientTypeID FROM rp\_IngredientType

WHERE IngredientType = @type

DECLARE @detailID int

SELECT @detailID = rp\_IngredientDetailID FROM rp\_IngredientDetail

WHERE rp\_IngredientTypeID = @typeID AND IngredientSource = @source AND IngredientPart = @part

-- Add the row

INSERT INTO rp\_IngredientNutritionFacts(rp\_IngredientDetailID, IngredientCalorie, IngredientProtein, IngredientFat, IngredientCarbs, IngredientFiber, IngredientSugar)

VALUES (@detailID, @calorie, @protein, @fat, @carbs, @fiber, @sugar)

RETURN SCOPE\_IDENTITY()

END

GO

-- Add ingredient into stock procedures

CREATE OR ALTER PROCEDURE rp\_pro\_AddIngredientIntoStock (@email varchar(50), @stName varchar(30), @type varchar(20), @source varchar(20), @part varchar(20) NULL, @amount int)

AS

BEGIN

DECLARE @stID int

SELECT @stID = rp\_StockID FROM rp\_Stock

WHERE rp\_OwnerID = (SELECT rp\_OwnerID FROM rp\_Owner WHERE EmailAddress = @email) AND StorageName = @stName

DECLARE @typeID int

SELECT @typeID = rp\_IngredientTypeID FROM rp\_IngredientType

WHERE IngredientType = @type

DECLARE @detailID int

SELECT @detailID = rp\_IngredientDetailID FROM rp\_IngredientDetail

WHERE rp\_IngredientTypeID = @typeID AND IngredientSource = @source OR IngredientPart = @part

-- Add the row

INSERT INTO rp\_IngredientStock(rp\_StockID, rp\_IngredientDetailID, IngredientAmount)

VALUES (@stID, @detailID, @amount)

RETURN SCOPE\_IDENTITY()

END

GO

-- trigger that is fired whenever update occurs against rp\_IngredientStock table

CREATE TRIGGER trg\_upd\_IngredientStock

ON rp\_IngredientStock

FOR UPDATE

AS

BEGIN

INSERT INTO rp\_IngredientStockUpdate (

rp\_IngredientStockID,

UpdatedDate

)

SELECT

rp\_IngredientStockID,

GETDATE()

FROM inserted

END

GO

-- trigger that is fired whene specific update occurs against rp\_IngredientStock table

CREATE TRIGGER trg\_upd2\_IngredientStock

ON rp\_IngredientStock

AFTER UPDATE

AS

BEGIN

SET NOCOUNT ON;

-- specific update = ingredient amount updated to 0g

UPDATE rp\_IngredientStock SET Finished = 'Y'

FROM rp\_IngredientStock t

INNER JOIN inserted i on t.rp\_IngredientStockID = i.rp\_IngredientStockID

AND i.IngredientAmount = 0

END

GO

-- Add ingredient price info into store procedures

CREATE OR ALTER PROCEDURE rp\_pro\_AddIngredientIntoStore (@storeN varchar(30), @type varchar(20), @source varchar(20), @part varchar(20) NULL, @price int)

AS

BEGIN

DECLARE @storeID int

SELECT @storeID = rp\_StoreID FROM rp\_Store

WHERE StoreName = @storeN

DECLARE @typeID int

SELECT @typeID = rp\_IngredientTypeID FROM rp\_IngredientType

WHERE IngredientType = @type

DECLARE @detailID int

SELECT @detailID = rp\_IngredientDetailID FROM rp\_IngredientDetail

WHERE rp\_IngredientTypeID = @typeID AND IngredientSource = @source OR IngredientPart = @part

-- Add the row

INSERT INTO rp\_IngredientStore(rp\_StoreID, rp\_IngredientDetailID, IngredientPricePer100g)

VALUES (@storeID, @detailID, @price)

RETURN SCOPE\_IDENTITY()

END

GO

-- Add meal model type amount by calcultaing within the data we already have

-- for BARF meal model

CREATE OR ALTER PROCEDURE rp\_pro\_AddMealModelTypeAmount\_BARF (@petName varchar(20))

AS

BEGIN

DECLARE @mealmodelID int

SELECT @mealmodelID = rp\_PetMealModelID FROM rp\_PetMealModel

WHERE rp\_PetID = (SELECT rp\_PetID FROM rp\_Pet WHERE PetName = @petName)

-- declare type1:7 from BARF model

DECLARE @type1ID int

SELECT @type1ID = rp\_IngredientTypeID FROM rp\_IngredientType

WHERE IngredientType ='Muscle meat'

DECLARE @type2ID int

SELECT @type2ID = rp\_IngredientTypeID FROM rp\_IngredientType

WHERE IngredientType ='Edible bone'

DECLARE @type3ID int

SELECT @type3ID = rp\_IngredientTypeID FROM rp\_IngredientType

WHERE IngredientType ='Organ'

DECLARE @type4ID int

SELECT @type4ID = rp\_IngredientTypeID FROM rp\_IngredientType

WHERE IngredientType ='Liver'

DECLARE @type5ID int

SELECT @type5ID = rp\_IngredientTypeID FROM rp\_IngredientType

WHERE IngredientType ='Vegetable'

DECLARE @type6ID int

SELECT @type6ID = rp\_IngredientTypeID FROM rp\_IngredientType

WHERE IngredientType ='Fruits'

DECLARE @type7ID int

SELECT @type7ID = rp\_IngredientTypeID FROM rp\_IngredientType

WHERE IngredientType ='Seeds/Nuts'

-- declare required calorie for pet

DECLARE @cal int

SELECT @cal = PetRequiredCalorie FROM rp\_PetHealthinfo

WHERE rp\_PetID = (SELECT rp\_PetID FROM rp\_Pet WHERE PetName = @petName)

-- declare ingredient amount of each type

DECLARE @t1amount int

SELECT @t1amount = @cal \* (SELECT MealModelTypeRatio FROM rp\_MealModelRatio WHERE rp\_IngredientTypeID = @type1ID)

DECLARE @t2amount int

SELECT @t2amount = @cal \* (SELECT MealModelTypeRatio FROM rp\_MealModelRatio WHERE rp\_IngredientTypeID = @type2ID)

DECLARE @t3amount int

SELECT @t3amount = @cal \* (SELECT MealModelTypeRatio FROM rp\_MealModelRatio WHERE rp\_IngredientTypeID = @type3ID)

DECLARE @t4amount int

SELECT @t4amount = @cal \* (SELECT MealModelTypeRatio FROM rp\_MealModelRatio WHERE rp\_IngredientTypeID = @type4ID)

DECLARE @t5amount int

SELECT @t5amount = @cal \* (SELECT MealModelTypeRatio FROM rp\_MealModelRatio WHERE rp\_IngredientTypeID = @type5ID)

DECLARE @t6amount int

SELECT @t6amount = @cal \* (SELECT MealModelTypeRatio FROM rp\_MealModelRatio WHERE rp\_IngredientTypeID = @type6ID)

DECLARE @t7amount int

SELECT @t7amount = @cal \* (SELECT MealModelTypeRatio FROM rp\_MealModelRatio WHERE rp\_IngredientTypeID = @type7ID)

-- Add the row

INSERT INTO rp\_MealModelTypeAmount (rp\_PetMealModelID, rp\_IngredientTypeID, MealModelTypeAmount)

VALUES (@mealmodelID, @type1ID, @t1amount)

, (@mealmodelID, @type2ID, @t2amount)

, (@mealmodelID, @type3ID, @t3amount)

, (@mealmodelID, @type4ID, @t4amount)

, (@mealmodelID, @type5ID, @t5amount)

, (@mealmodelID, @type6ID, @t6amount)

, (@mealmodelID, @type7ID, @t7amount)

RETURN SCOPE\_IDENTITY()

END

GO