**Group Project – Skills Demonstration**

Team Name 1: **Joni Dolson** Date: **04.01.24**

Team Name 2: **Levi Noell-Baba**

Team Name 3: **Kassidy Craft**

Team Name 4: **Steven Owens**

# **Group Project Instructions**

1. Type the names of team members that participated and the date in the spaces provided.
2. Use the SQL Server Management Studio and create and use a new database named xxGPDatabase (where the xx are the initials of one of your team members).
3. Complete all exercises in this group project.
4. Upload and submit a completed copy of this group project worksheet before the due date. Each team member that participated is to upload and submit a copy of this document.

**Exercises:**

# **Group Project Exercise 1(Database Design & Creation)**

Design and create a normalized database named xxGPDatabase where the xx are the initials of one of your team members. In this database create tables that a coffee shop can use to manage their Menu, Recipes, Inventory, Employees, and Work Schedules.

The Coffee Shop is open 24 hours a day. Each employee can work a max of 40 hours a week. Each shift is to be staffed with three employees. Each menu item has a recipe that specifies the ingredients and directions. All ingredients except for water are to be tracked in an inventory table.

The database needs to be normalized (3NF / Boyce-Codd Normal Form and no Multi-Valued Dependency). Each table must have a primary key and other appropriate attributes. The attributes on the tables must have appropriate datatypes, lengths, and constraints defined. At least 3 check constraints must be present. In addition, constraints to create appropriate relationships between the tables and enforce referential integrity must be present.

**--- Paste below this line a script file that when executed creates the xxGPDatabase with all the structural components described above. --**

CREATE DATABASE JDGPDatabase;

Go

USE JDGPDatabase;

Go

CREATE TABLE Employees (

EmployeeID INT PRIMARY KEY,

FirstName VARCHAR(100),

LastName VARCHAR(100),

HoursWorked INT CHECK (HoursWorked <= 40)

);

CREATE TABLE Shifts (

ShiftID INT PRIMARY KEY,

ShiftStart DATETIME,

ShiftEnd DATETIME,

Employee1 INT,

Employee2 INT,

Employee3 INT,

FOREIGN KEY (Employee1) REFERENCES Employees(EmployeeID),

FOREIGN KEY (Employee2) REFERENCES Employees(EmployeeID),

FOREIGN KEY (Employee3) REFERENCES Employees(EmployeeID),

CONSTRAINT CHK\_ShiftTime CHECK (ShiftStart < ShiftEnd)

);

CREATE TABLE Inventory (

IngredientID INT PRIMARY KEY,

IngredientName VARCHAR(100),

Quantity INT CHECK (Quantity >= 0)

);

CREATE TABLE Recipes (

RecipeID INT PRIMARY KEY,

IngredientID INT,

QuantityNeeded INT,

Directions TEXT,

FOREIGN KEY (IngredientID) REFERENCES Inventory(IngredientID)

);

CREATE TABLE Menu (

MenuItemID INT PRIMARY KEY,

MenuItemName VARCHAR(100),

RecipeID INT,

FOREIGN KEY (RecipeID) REFERENCES Recipes(RecipeID)

);

**--- Paste below this line an image of a database diagram that shows the relationships between the tables --**

A screenshot of a computer

Description automatically generated

# **Group Project Exercise 2 (Database Design & Creation)**

Code and execute Insert Statements to insert six or seven records into each of the tables in the xxGPDatabase Database. Make sure to use each team members name once when coding attribute values for first and last name attributes. Make up appropriate demonstration values for other attribute values.

**--- Paste below this line the insert statements your team coded. –**

Use JDGPDatabase;

INSERT INTO Employees (EmployeeID, FirstName, LastName, HoursWorked)

VALUES (1, 'Joni', 'Dolson', 35),

(2, 'Steven', 'Owens', 30),

(3, 'Kassidy', 'Craft', 33),

(4, 'Levi', 'Noell-Baba', 23),

(5, 'John', 'Nolan', 13),

(6, 'Nathan', 'Phillips', 26),

(7, 'Sarah', 'Price', 31);

INSERT INTO Shifts (ShiftID, ShiftStart, ShiftEnd, Employee1, Employee2, Employee3)

VALUES (1, '2024-04-11 08:00:00', '2024-04-11 14:00:00', 1, 2, 3),

(2, '2024-04-11 14:00:00', '2024-04-11 22:00:00', 4, 5, 6),

(3, '2024-04-12 08:00:00', '2024-04-12 14:00:00', 2, 3, 7),

(4, '2024-04-12 14:00:00', '2024-04-12 22:00:00', 1, 4, 5),

(5, '2024-04-13 08:00:00', '2024-04-13 14:00:00', 3, 5, 6),

(6, '2024-04-13 14:00:00', '2024-04-13 22:00:00', 2, 4, 7),

(7, '2024-04-14 08:00:00', '2024-04-14 14:00:00', 1, 3, 6);

INSERT INTO Inventory (IngredientID, IngredientName, Quantity)

VALUES (1, 'Flour', 100),

(2, 'Sugar', 75),

(3, 'Eggs', 200),

(4, 'Milk', 50),

(5, 'Chocolate Chips', 70),

(6, 'Butter', 60),

(7, 'Cinnamon', 15),

(8, 'Bread', 300),

(9, 'Salt', 50),

(10, 'Chocolate', 225),

(11, 'Vanilla Extract', 25);

INSERT INTO Recipes (RecipeID, IngredientID, QuantityNeeded, Directions)

VALUES (1, 1, 50, 'Beat the butter and sugar, then beat in the eggs and vanilla. Stir in the flour and chocolate chips. Drop dough onto a prepared baking sheet. Bake until the edges are golden brown at 350°F.'),

(2, 2, 15, 'Mix flour, sugar, and vanilla extract in a large bowl. Add eggs and milk. Mix and pour into cake tin. Bake in oven at 350°F until a toothpick inserted into the center of the cake comes out clean.'),

(3, 3, 25, 'Whisk milk, eggs, vanilla, cinnamon, and salt together in a shallow bowl. Lightly butter a griddle or skillet and heat over medium-high heat. Dunk bread in the egg mixture, soaking both sides. Cook in a hot skillet until golden on both sides'),

(4, 4, 10, 'Mix milk with sugar and vanilla extract. Churn in ice cream maker until frozen.'),

(5, 5, 20, 'Melt butter on the stove, then remove from heat and add the remaining wet ingredients. Beat in the chocolate, flour, vanilla extract, eggs, sugar, and salt. Spread in pan and bake at 350°F until the edges are pulling away from the pan.'),

(6, 6, 30, 'Beat sugar, butter, egg, and vanilla extract in a bowl. Gradually blend the flour into the butter mixture. Roll the dough into balls and bake until the edges are golden at 350°F.'),

(7, 7, 5, 'Beat sugar and butter together in a large bowl until light and fluffy. Add Three eggs one at a time, beating well after each addition. Beat in vanilla extract. Stir in flour just until mixed. Spoon batter into the prepared muffin cups until they are 3/4 full. Bake at 350°F until the tops spring back when lightly pressed.');

INSERT INTO Menu (MenuItemID, MenuItemName, RecipeID)

VALUES (1, 'Chocolate chip cookies', 5),

(2, 'Vanilla Cake', 2),

(3, 'French Toast', 3),

(4, 'Vanilla Ice Cream', 4),

(5, 'Chocolate Brownies', 1),

(6, 'Sugar Cookies', 6),

(7, 'Vanilla Cupcakes', 7);

**---** **Paste below this line the completion status messages from executing your team’s insert statements-**

(7 rows affected)

(7 rows affected)

(11 rows affected)

(7 rows affected)

(7 rows affected)

Completion time: 2024-04-11T23:41:45.6563733-04:00

# **Group Project Exercise 3 (Views)**

Create a view named GPViewXXXXXXXXXXXX where the XXXXXXXXXXXX is the last name of one of your team members. This view must use at least two of your tables and the result set needs to be meaningful.

**--- Paste below this line your team’s view creation code. –**

DECLARE @sql NVARCHAR(MAX);

SET @sql = N'

CREATE VIEW GPViewCraft AS

SELECT

M.MenuItemID,

M.MenuItemName,

R.RecipeID,

R.Directions,

I.IngredientID,

I.IngredientName,

R.QuantityNeeded,

I.Quantity AS InventoryQuantity

FROM

Menu M

INNER JOIN

Recipes R ON M.RecipeID = R.RecipeID

INNER JOIN

Inventory I ON R.IngredientID = I.IngredientID;

';

EXEC sp\_executesql @sql;

**---** **Paste below this line the completion status messages from executing your team’s view creation code--**

Commands completed successfully.

Completion time: 2024-04-12T02:01:46.8964913-04:00

**---** **Paste below this line the result set returned by running a SELECT \* from the view your team created. --**

1 Chocolate chip cookies 5 Melt butter on the stove, then remove from heat and add the remaining wet ingredients. Beat in the chocolate, flour, vanilla extract, eggs, sugar, and salt. Spread in pan and bake at 350°F until the edges are pulling away from the pan. 5 Chocolate Chips 20 70

2 Vanilla Cake 2 Mix flour, sugar, and vanilla extract in a large bowl. Add eggs and milk. Mix and pour into cake tin. Bake in oven at 350°F until a toothpick inserted into the center of the cake comes out clean. 2 Sugar 15 75

3 French Toast 3 Whisk milk, eggs, vanilla, cinnamon, and salt together in a shallow bowl. Lightly butter a griddle or skillet and heat over medium-high heat. Dunk bread in the egg mixture, soaking both sides. Cook in a hot skillet until golden on both sides 3 Eggs 25 200

4 Vanilla Ice Cream 4 Mix milk with sugar and vanilla extract. Churn in ice cream maker until frozen. 4 Milk 10 50

5 Chocolate Brownies 1 Beat the butter and sugar, then beat in the eggs and vanilla. Stir in the flour and chocolate chips. Drop dough onto a prepared baking sheet. Bake until the edges are golden brown at 350°F. 1 Flour 50 100

6 Sugar Cookies 6 Beat sugar, butter, egg, and vanilla extract in a bowl. Gradually blend the flour into the butter mixture. Roll the dough into balls and bake until the edges are golden at 350°F. 6 Butter 30 60

7 Vanilla Cupcakes 7 Beat sugar and butter together in a large bowl until light and fluffy. Add Three eggs one at a time, beating well after each addition. Beat in vanilla extract. Stir in flour just until mixed. Spoon batter into the prepared muffin cups until they are 3/4 full. Bake at 350°F until the tops spring back when lightly pressed. 7 Cinnamon 5 15

# **Group Project Exercise 4 (Stored Procedures)**

Create a stored procedure named GPProcXXXXXXXXXXXX where the XXXXXXXXXXXX is the last name of one of your team members. This stored procedure must contain logic that involves at least two of your tables and it must produce a meaningful report when executed.

**--- Paste below this line your team’s stored procedure creation code. –**USE JDGPDatabase;

GO

CREATE PROCEDURE GPProcOwens

@StartDate DATE,

@EndDate DATE

AS

BEGIN

SELECT

e.EmployeeID,

e.FirstName,

e.LastName,

s.ShiftID,

s.ShiftStart,

s.ShiftEnd

FROM

Employees e

INNER JOIN

Shifts s ON e.EmployeeID IN (s.Employee1, s.Employee2, s.Employee3)

WHERE

s.ShiftStart >= @StartDate

AND s.ShiftEnd <= @EndDate;

END;

**---** **Paste below this line the completion status messages from executing your team’s stored procedure creation code--**

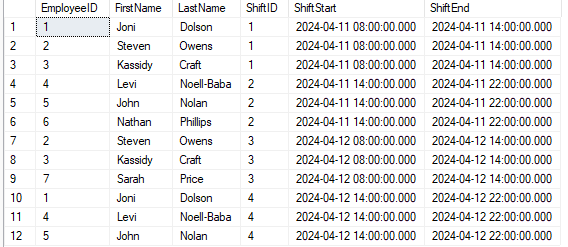
Commands completed successfully.

Completion time: 2024-04-13T11:09:28.0262693-04:00

**---** **Paste below the execute statement your team used to test the stored procedure --**

EXEC GPProcOwens '2024-04-11', '2024-04-13';

**---** **Paste below this line the run results from the statement used to test the stored procedure--**



# **Group Project Exercise 5 (Triggers)**

Code a trigger named TR\_GP\_DML\_XXXXXXXXXXXXXX where the XXXXXXXXXXXXXX is the last name of one of your team members. This trigger must respond to inserts and/or updates to one of your tables and the trigger must perform a meaningful purpose.

**--- Paste below this line your team’s trigger creation code. –**

USE JDGPDatabase;

GO

CREATE TRIGGER TR\_GP\_DML\_NoellBaba

ON Employees

INSTEAD OF INSERT

AS

BEGIN

IF EXISTS(

SELECT 1

FROM inserted i

JOIN Employees e ON LOWER(e.FirstName) = LOWER(i.FirstName) AND LOWER(e.LastName) = LOWER(i.LastName)

)

BEGIN

PRINT 'Error: An employee with the same name already exists.';

ROLLBACK TRANSACTION;

END;

ELSE

BEGIN

INSERT INTO Employees (EmployeeID, FirstName, LastName, HoursWorked)

SELECT EmployeeID, FirstName, LastName, HoursWorked FROM inserted;

END;

End;

**---** **Paste below this line the completion status messages from executing your team’s trigger creation code--**

Commands completed successfully.

Completion time: 2024-04-15T14:02:13.7306824-04:00

**---** **Paste below this line the insert and/or update statements your team used to test the trigger --**

USE JDGPDatabase;

INSERT INTO Employees (EmployeeID, FirstName, LastName, HoursWorked)

VALUES (8, 'Matt', 'johnson', 40);

USE JDGPDatabase;

INSERT INTO Employees (EmployeeID, FirstName, LastName, HoursWorked)

VALUES (9, 'Matt', 'Smith', 40);

USE JDGPDatabase;

INSERT INTO Employees (EmployeeID, FirstName, LastName, HoursWorked)

VALUES (10, 'John', 'Smith', 40);

USE JDGPDatabase;

INSERT INTO Employees (EmployeeID, FirstName, LastName, HoursWorked)

VALUES (11, 'John', 'Smith', 40);

**---** **Paste below this line the run results from executing the statements used to test the trigger (include comments explaining what each test demonstrated) --**

(1 row affected)

(1 row affected)

Completion time: 2024-04-15T14:04:59.3698340-04:00

This first test shows that a newly inserted employee will be added to the Employees table as long as their first name and last name are different from an existing employee.

(1 row affected)

(1 row affected)

Completion time: 2024-04-15T14:06:33.3527785-04:00

This second test shows that a newly inserted employee will be added to the Employees table even if their first name is the same as an existing employee as long as a different last name is inserted.

(1 row affected)

(1 row affected)

Completion time: 2024-04-15T14:08:07.3659109-04:00

This third test shows that a newly inserted employee will be added to the Employees table even if their last name is the same as an existing employee as long as a different first name is inserted.

Error: An employee with the same name already exists.

Msg 3609, Level 16, State 1, Line 2

The transaction ended in the trigger. The batch has been aborted.

Completion time: 2024-04-15T14:09:28.1680561-04:00

This fourth test shows that a newly inserted employee will not be added to the Employees table if their first and last name is the same as an existing employees first and last name. The message properly displays the error message.

# **Group Project Exercise 6 (Security)**

Write a script that creates a user-defined database role in the XXGPDatabase database (reminder replace the XX with the appropriate initials). Name this new role ProductManagementXX where the XX are your initials. Give UPDATE permission to the new role for the tables that contain menu, recipe, and inventory data, and SELECT permission for all other user tables. Make sure to include a comment at the top of the script with your name and the date you coded the script. Format this comment like the following sample:

/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* ProductManagementXX Role Creation Script \*/

/\* Coded by: Your Name \*/

/\* Coded on: MM/DD/YY \*/

/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**---** **Paste below this line the Script your team wrote for this exercise --**

/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* ProductManagementKC Role Creation and Permission Script \*/

/\* Coded by: Kassidy Craft \*/

/\* Coded on: 04/14/24 \*/

/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

USE JDGPDatabase;

DECLARE @roleName NVARCHAR(128) = N'ProductManagementKC';

DECLARE @sql NVARCHAR(MAX);

DECLARE @exists INT;

SELECT @exists = COUNT(\*)

FROM sys.database\_principals

WHERE type\_desc = 'DATABASE\_ROLE' AND name = @roleName;

IF @exists = 0

BEGIN

SET @sql = N'CREATE ROLE ' + QUOTENAME(@roleName);

EXEC sp\_executesql @sql;

END

SET @sql = N'GRANT UPDATE ON Menu TO ' + QUOTENAME(@roleName) + ';' +

N'GRANT UPDATE ON Recipes TO ' + QUOTENAME(@roleName) + ';' +

N'GRANT UPDATE ON Inventory TO ' + QUOTENAME(@roleName) + ';';

EXEC sp\_executesql @sql;

SET @sql = N'GRANT SELECT ON Employees TO ' + QUOTENAME(@roleName) + ';' +

N'GRANT SELECT ON Shifts TO ' + QUOTENAME(@roleName) + ';';

EXEC sp\_executesql @sql;

**---** **Paste below this line the run results from executing the script your team wrote--**

Completion time: 2024-04-14T19:16:59.9571742-04:00

Commands completed successfully.

# **Group Project Exercise 7 (Security)**

Use the XXGPDatabase database and write a script that:

1. Creates a login ID named “GPDLOG01XX” where the XX are your initials
2. Assigns this login an initial password of “GPDr33333”
3. Sets the default database for this login to the XXGPDatabase database
4. Creates a user named “GPDLOG01XX” where the XX are your initials
5. Assigns the “GPDLOG01XX” user to the ProductManagementXX role you created in exercise 6
6. Includes a comment at the top of the script with your name and the date you coded the script. Format this comment like the following sample:

/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* GPDLOG01XX Login and User Creation Script \*/

/\* Coded by: Your Name \*/

/\* Coded on: MM/DD/YY \*/

/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**--- Paste below this line the Script your team wrote for this exercise --**

USE JDGPDatabase;

Go

/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* GPDLOG01SO Login and User Creation Script \*/

/\* Coded by: Steven Owens \*/

/\* Coded on: 04/13/24 \*/

/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

DECLARE @loginName NVARCHAR(100);

DECLARE @initials NVARCHAR(2);

SET @initials = 'SO';

SET @loginName = 'GPDLOG01' + @initials;

DECLARE @sqlCreateLogin NVARCHAR(MAX);

SET @sqlCreateLogin = '

CREATE LOGIN ' + QUOTENAME(@loginName) + ' WITH PASSWORD = ''GPDr33333'', DEFAULT\_DATABASE = JDGPDatabase;

';

EXEC sp\_executesql @sqlCreateLogin;

DECLARE @sqlCreateUser NVARCHAR(MAX);

SET @sqlCreateUser = '

CREATE USER ' + QUOTENAME(@loginName) + ' FOR LOGIN ' + QUOTENAME(@loginName) + ';

';

EXEC sp\_executesql @sqlCreateUser;

DECLARE @sqlAssignRole NVARCHAR(MAX);

SET @sqlAssignRole = '

ALTER ROLE ProductManagementKC ADD MEMBER ' + QUOTENAME(@loginName) + ';

';

EXEC sp\_executesql @sqlAssignRole;

**---** **Paste below this line the run results from executing the script your team wrote--**

Completion time: 2024-04-14T20:25:28.1607066-04:00

# **Group Project Exercise 8 (Security)**

Write a script that uses a table, dynamic SQL, and a cursor to create login IDs in the XXGPDatabase database based on the contents of the table.

1. Get started by first creating the table. Below is a create statement that you can modify and execute to create this table. Make sure to change the XX in the table name to your initials:

CREATE TABLE NewLoginsXX (LoginName varchar(128));

1. After the table is created. Insert four rows in it. Below is an insert statement that you can modify and execute to put the rows in your NewLoginsXX table. In addition to modifying the table name, make sure to change the XX at the start of each login name to your initials before running this insert.

INSERT NewLoginsXX VALUES ('XXFFlake'), ('XXGGlute'), ('XXMMrit'), ('XXPPert');

1. Start coding the script with a comment block. Format this comment like the following sample (make sure to revise the sample to reflect your name and the date you coded it):

/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Login Generator Script \*/

/\* This Script uses dynamic SQL and a cursor to process the \*/

/\* NEWLoginsXX table one row at a time. For each row on this \*/

/\* table the script will perform four actions. \*/

/\* \*/

/\* Coded by: Your Name \*/

/\* Coded on: MM/DD/YY \*/

/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

1. Code the rest of your script using dynamic SQL and a cursor to perform four actions for each row in this table:

(1) create a login with a temporary password that's based on the first four letters of the login name followed by “33333”

(2) set the default database to the XXGPDatabase database

(3) create a user for the login with the same name as the login

(4) assign the user to the ProductManagementXX role you created in exercise 6.

**--- Paste below this line the Script your team wrote for this exercise --A.)**

CREATE TABLE NewLoginsKC (

LoginName VARCHAR(128)

);

**B.)**

INSERT INTO NewLoginsKC VALUES

('KCFFlake'),

('KCGGlute'),

('KCMMrit'),

('KCPPert');

**C – D .)**

/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Login Generator Script \*/

/\* This Script uses dynamic SQL and a cursor to process the \*/

/\* NewLoginsKC table one row at a time. For each row in this \*/

/\* table, the script will perform four actions. \*/

/\* \*/

/\* Coded by: Kassidy Craft \*/

/\* Coded on: 04/14/24 \*/

/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

USE JDGPDatabase;

GO

DECLARE @LoginName NVARCHAR(128);

DECLARE @Sql NVARCHAR(MAX);

DECLARE @DefaultDB NVARCHAR(128) = N'JDGPDatabase';

DECLARE @Password NVARCHAR(128);

DECLARE @RoleName NVARCHAR(128) = N'ProductManagementKC';

DECLARE login\_cursor CURSOR FOR

SELECT LoginName FROM NewLoginsKC;

OPEN login\_cursor;

FETCH NEXT FROM login\_cursor INTO @LoginName;

WHILE @@FETCH\_STATUS = 0

BEGIN

SET @Password = LEFT(@LoginName, 4) + '33333';

SET @Sql = 'CREATE LOGIN ' + QUOTENAME(@LoginName) + ' WITH PASSWORD = ''' + @Password + ''', DEFAULT\_DATABASE = ' + @DefaultDB;

EXEC sp\_executesql @Sql;

SET @Sql = 'CREATE USER ' + QUOTENAME(@LoginName) + ' FOR LOGIN ' + QUOTENAME(@LoginName);

EXEC sp\_executesql @Sql;

SET @Sql = 'ALTER ROLE ' + QUOTENAME(@RoleName) + ' ADD MEMBER ' + QUOTENAME(@LoginName);

EXEC sp\_executesql @Sql;

FETCH NEXT FROM login\_cursor INTO @LoginName;

END

CLOSE login\_cursor;

DEALLOCATE login\_cursor;

**---** **Paste below this line the run results from executing the script your team wrote—**

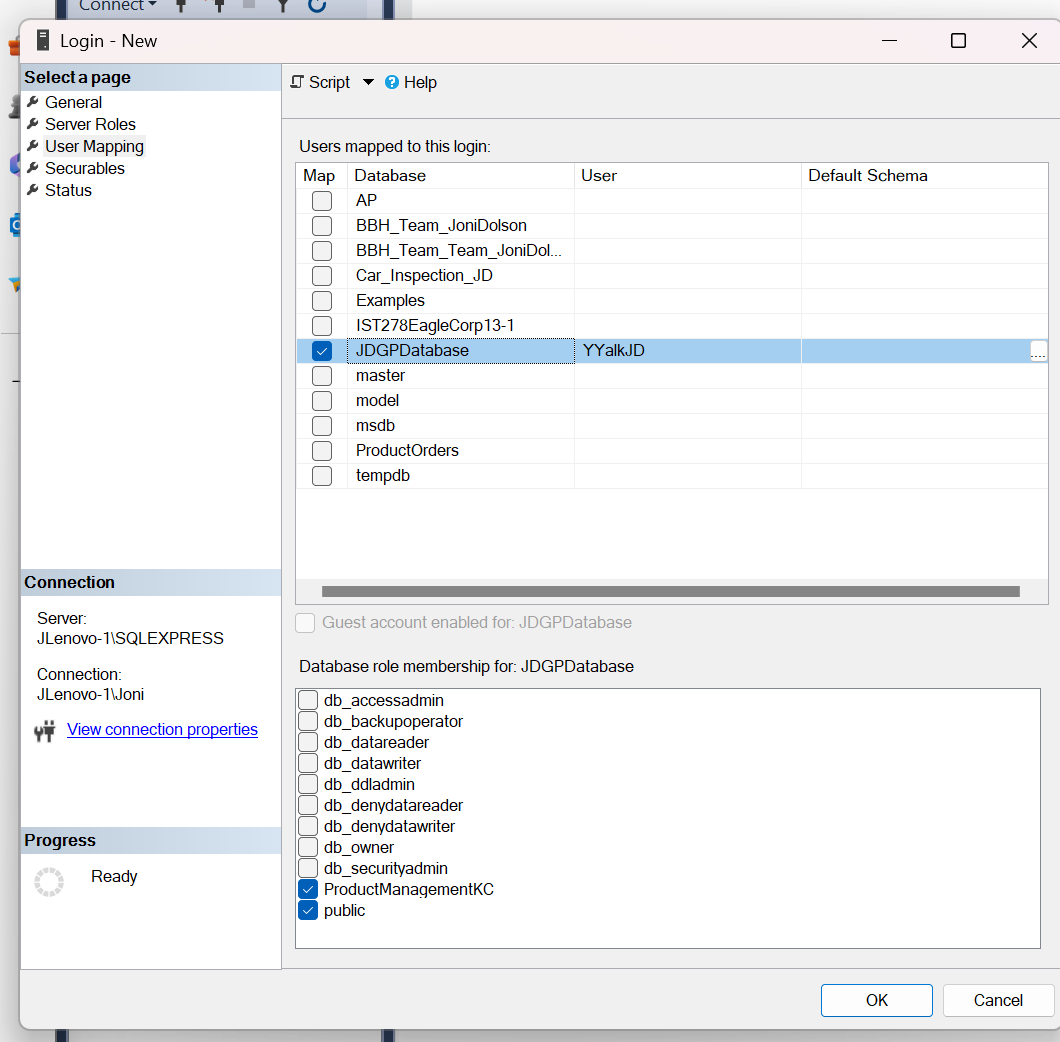
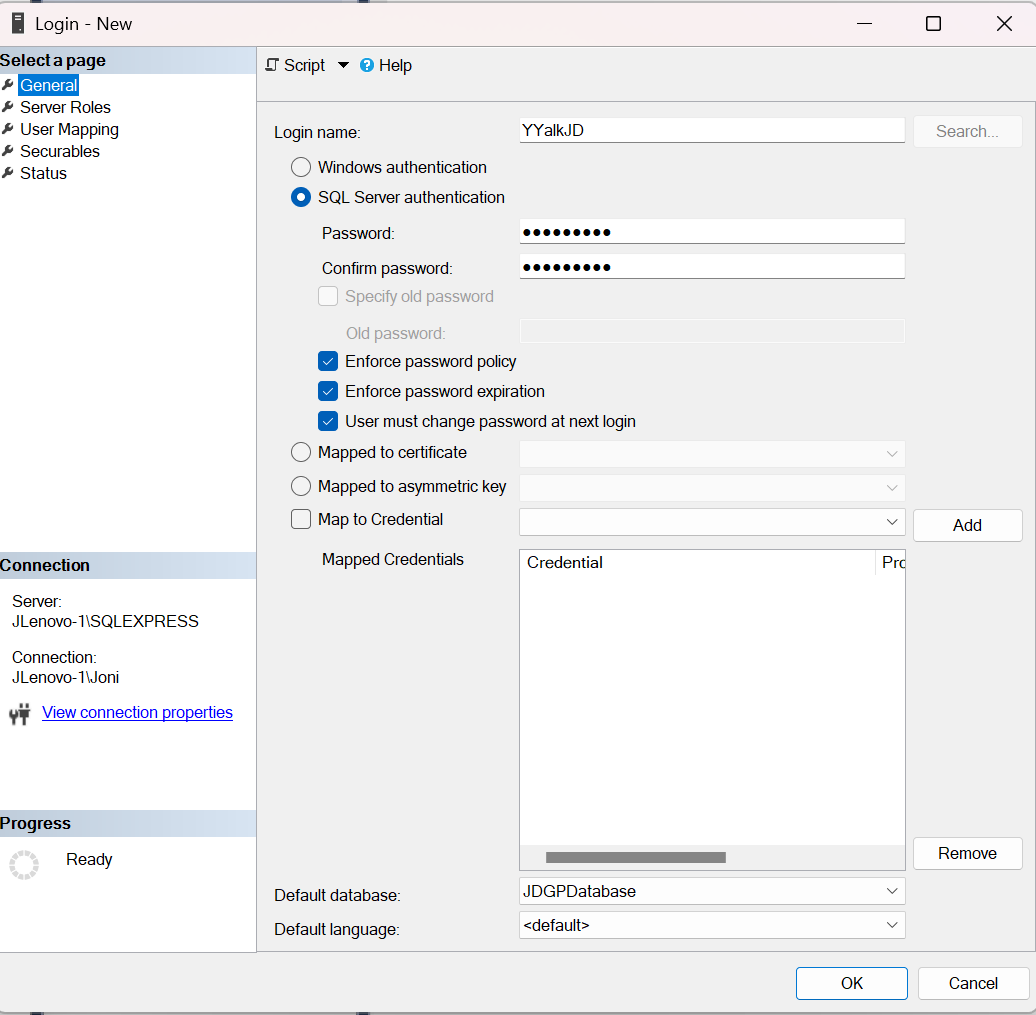
Commands completed successfully.

# **Group Project Exercise 9 (Security)**

Using the Management Studio, create a login ID named “YYalkXX” where the XX are your initials. Assign this login ID a password “YYal33333,” and set the default database to the XXGPDatabase database. Then, grant the login ID access to the XXGPDatabase database, create a user for the login ID named “YYalkXX”, and assign the user to the PaymentEntry role you created in exercise 6.

Note: If you get an error that says, “The MUST\_CHANGE option is not supported”, you can deselect the “Enforce password policy” option for the login ID.

**---** **Paste below this line a screen shot that shows the new login creation window with the YYalkXX information showing—**



**---** **Paste below a screen shot that shows the login properties window with the YYalkXX’s mapping to the XXGPDatabase database and membership in the ProductManagementXX role your team created in exercise 6.--**

