Exam Basics DB 22.X.2017 ReportService

/\*PART I DDL \*/

/\* 01DatabaseDesign\_DDL\*/

CREATE TABLE Users

(

Id INT NOT NULL IDENTITY CONSTRAINT PK\_Users PRIMARY KEY,

Username NVARCHAR(30) UNIQUE NOT NULL,

[Password] NVARCHAR(50) NOT NULL,

[Name] NVARCHAR(50),

Gender CHAR(1) CONSTRAINT CH\_Gender CHECK(Gender IN('M', 'F')),

BirthDate DATETIME,

Age INT,

Email NVARCHAR(50) NOT NULL

)

CREATE TABLE Departments

(

Id INT NOT NULL IDENTITY CONSTRAINT PK\_Departments PRIMARY KEY,

[Name] NVARCHAR(50) NOT NULL

)

CREATE TABLE Employees

(

Id INT NOT NULL IDENTITY CONSTRAINT PK\_Employees PRIMARY KEY,

FirstName NVARCHAR(25),

LastName NVARCHAR(25),

Gender CHAR(1) CHECK(Gender IN('M', 'F')),

BirthDate DATETIME,

Age INT,

DepartmentId INT NOT NULL CONSTRAINT FK\_Employees\_Departments FOREIGN KEY REFERENCES Departments(Id)

)

CREATE TABLE Categories

(

Id INT NOT NULL IDENTITY CONSTRAINT PK\_Categories PRIMARY KEY,

[Name] VARCHAR(50) NOT NULL,

DepartmentId INT CONSTRAINT FK\_Categories\_Departments FOREIGN KEY REFERENCES Departments(Id)

)

CREATE TABLE [Status]

(

Id INT NOT NULL IDENTITY CONSTRAINT PK\_Status PRIMARY KEY,

Label VARCHAR(50) NOT NULL

)

CREATE TABLE Reports

(

Id INT NOT NULL IDENTITY CONSTRAINT PK\_Reports PRIMARY KEY,

CategoryId INT NOT NULL CONSTRAINT FK\_Reports\_Categories FOREIGN KEY REFERENCES Categories(Id),

StatusId INT NOT NULL CONSTRAINT FK\_Reports\_Status FOREIGN KEY REFERENCES [Status](Id),

OpenDate DATETIME NOT NULL,

CloseDate DATETIME,

[Description] VARCHAR(200),

UserId INT NOT NULL CONSTRAINT FK\_Reports\_Users FOREIGN KEY REFERENCES Users(Id),

EmployeeId INT CONSTRAINT FK\_Reports\_Employees FOREIGN KEY REFERENCES Employees(Id)

)

/\*PART II DML \*/

/\* 02Insert\_DML\*/

INSERT INTO Employees(FirstName, LastName, Gender, BirthDate, DepartmentId)

VALUES

('Marlo', 'O’Malley', 'M', '9/21/1958', 1),

('Niki', 'Stanaghan', 'F', '11/26/1969', 4),

('Ayrton', 'Senna', 'M', '03/21/1960', 9),

('Ronnie', 'Peterson', 'M', '02/14/1944', 9),

('Giovanna', 'Amati', 'F', '07/20/1959', 5)

INSERT INTO Reports(CategoryId, StatusId, OpenDate, CloseDate, [Description], UserId, EmployeeId)

VALUES

(1, 1, '04/13/2017', NULL, 'Stuck Road on Str.133', 6, 2),

(6, 3, '09/05/2015', '12/06/2015', 'Charity trail running', 3, 5),

(14, 2, '09/07/2015', NULL, 'Falling bricks on Str.58', 5, 2),

(4, 3, '07/03/2017', '07/06/2017', 'Cut off streetlight on Str.11', 1, 1)

/\* 03Update\_DML \*/

UPDATE Reports

SET StatusId = 2 WHERE StatusId = 1 AND CategoryId = 4

--Only one query must be paste in Judge

UPDATE Reports

SET StatusId = (SELECT Id FROM [Status] WHERE Label = 'in progress')

WHERE StatusId = (SELECT Id FROM [Status] WHERE Label = 'waiting') AND

CategoryId = (SELECT Id FROM Categories WHERE [Name] = 'Streetlight')

/\* 04Delete\_DML \*/

DELETE FROM Reports WHERE StatusId = 4

--Only one query must be paste in Judge

DELETE FROM Reports

WHERE StatusId = (SELECT Id FROM [Status] WHERE Label = 'blocked')

/\*PART III Querying \*/

/\* 05UsersByAge\_Querying\*/

SELECT Username, Age

FROM Users

ORDER BY Age ASC, Username DESC

/\* 06UnassignedReports\_Querying \*/

SELECT [Description], OpenDate

FROM Reports

WHERE EmployeeId IS NULL

ORDER BY OpenDate, [Description]

/\* 07EmployeesAndRecords\_Querying \*/

SELECT e.FirstName, e.LastName, r.[Description], FORMAT(r.OpenDate, 'yyyy-MM-dd') AS OpenDate

FROM Employees AS e

JOIN Reports AS r ON r.EmployeeId = e.Id

ORDER BY e.Id, r.OpenDate, r.Id

/\* 08MostReportedCategory\_Querying \*/

SELECT c.[Name] AS CategoryName, COUNT(r.Id) AS ReportsNumber

FROM Categories AS c

JOIN Reports AS r ON r.CategoryId = c.Id

GROUP BY c.[Name]

ORDER BY COUNT(r.Id) DESC, c.[Name]

/\* 09EmployeesInCategory\_Querying \*/

SELECT c.[Name] AS CategoryName, COUNT(e.Id) AS [Employees Number]

FROM Categories AS c

JOIN Departments AS d ON d.Id = c.DepartmentId

JOIN Employees AS e ON e.DepartmentId = d.Id

GROUP BY c.[Name]

/\* 10UsersPerEmployee\_Querying \*/

SELECT DISTINCT e.FirstName + ' ' + e.LastName AS [Name], COUNT(r.UserId) AS [Users Number]

FROM Employees AS e

LEFT JOIN Reports AS r ON r.EmployeeId = e.Id

GROUP BY e.FirstName + ' ' + e.LastName

ORDER BY [Users Number] DESC, [Name]

--Only one query must be paste in Judge

SELECT e.FirstName + ' ' + e.LastName AS [Name], COUNT(DISTINCT r.UserId) AS [Users Number]

FROM Employees AS e

LEFT JOIN Reports AS r ON r.EmployeeId = e.Id

GROUP BY e.FirstName + ' ' + e.LastName

ORDER BY [Users Number] DESC, [Name]

/\* 11EmergencyPatrol\_Querying \*/

SELECT r.OpenDate, r.[Description], u.Email AS [Reporter Email]

FROM Reports AS r

JOIN Users AS u ON u.Id = r.UserId

JOIN Categories AS c ON c.Id = r.CategoryId

WHERE r.CloseDate IS NULL AND LEN(r.[Description]) > 20 AND r.[Description] LIKE '%str%' AND

c.DepartmentId IN(1, 4, 5)

ORDER BY r.OpenDate, [Reporter Email], r.Id

--Only one query must be paste in Judge

SELECT r.OpenDate, r.[Description], u.Email AS [Reporter Email]

FROM Reports AS r

JOIN Users AS u ON u.Id = r.UserId

JOIN Categories AS c ON c.Id = r.CategoryId

WHERE r.CloseDate IS NULL AND LEN(r.[Description]) > 20 AND r.[Description] LIKE '%str%' AND

c.DepartmentId IN(SELECT Id FROM Departments

WHERE [Name] IN('Infrastructure', 'Emergency', 'Roads Maintenance'))

ORDER BY r.OpenDate, [Reporter Email], r.Id

--Only one query must be paste in Judge

SELECT r.OpenDate, r.[Description], u.Email AS [Reporter Email]

FROM Reports AS r

JOIN Users AS u ON u.Id = r.UserId

JOIN Categories AS c ON c.Id = r.CategoryId

JOIN Departments AS d ON d.Id = c.DepartmentId

WHERE r.CloseDate IS NULL AND LEN(r.[Description]) > 20 AND r.[Description] LIKE '%str%' AND

d.[Name] IN('Infrastructure', 'Emergency', 'Roads Maintenance')

ORDER BY r.OpenDate, [Reporter Email], r.Id

--Only one query must be paste in Judge

SELECT r.OpenDate, r.[Description], u.Email AS [Reporter Email]

FROM Reports AS r

JOIN Users AS u ON u.Id = r.UserId

JOIN Categories AS c ON c.Id = r.CategoryId

JOIN Departments AS d ON (d.Id = c.DepartmentId AND

d.[Name] IN('Infrastructure', 'Emergency', 'Roads Maintenance'))

WHERE r.CloseDate IS NULL AND LEN(r.[Description]) > 20 AND r.[Description] LIKE '%str%'

ORDER BY r.OpenDate, [Reporter Email], r.Id

/\* 12BirthdayReport\_Querying \*/

SELECT DISTINCT c.[Name] AS [Category Name]

FROM Categories AS c

JOIN Reports AS r ON r.CategoryId = c.Id

JOIN Users AS u ON (u.Id = r.UserId AND DAY(u.BirthDate) = DAY(r.OpenDate)

AND MONTH(u.BirthDate) = MONTH(r.OpenDate))

ORDER BY c.[Name]

--Only one query must be paste in Judge

SELECT DISTINCT c.[Name] AS [Category Name]

FROM Categories AS c

JOIN Reports AS r ON r.CategoryId = c.Id

JOIN Users AS u ON u.Id = r.UserId

WHERE DAY(u.BirthDate) = DAY(r.OpenDate) AND MONTH(u.BirthDate) = MONTH(r.OpenDate)

ORDER BY c.[Name]

/\* 13NumbersCoincidence\_Querying \*/

SELECT DISTINCT u.Username

FROM Users AS u

JOIN Reports AS r ON r.UserId = u.Id

WHERE (LEFT(u.Username, 1) LIKE '[0-9]' AND TRY\_CAST(LEFT(u.Username, 1) AS INT) = r.CategoryId) OR

(RIGHT(u.Username, 1) LIKE '[0-9]' AND TRY\_CAST(RIGHT(u.Username, 1) AS INT) = r.CategoryId)

ORDER BY u.Username

/\* 14OpenIClosedStatistics\_Querying \*/

SELECT e.FirstName + ' ' + e.LastName AS [Name],

CONCAT(ISNULL(rC.ReportSum, 0), '/', ISNULL(rO.ReportSum, 0))

AS [Closed Open Reports]

FROM Employees AS e

JOIN (SELECT EmployeeId, COUNT(\*) AS ReportSum

FROM Reports WHERE YEAR(OpenDate) = 2016

GROUP BY EmployeeId) AS rO ON rO.EmployeeId = e.Id

LEFT JOIN (SELECT EmployeeId, COUNT(\*) AS ReportSum

FROM Reports WHERE YEAR(CloseDate) = 2016

GROUP BY EmployeeId) AS rC ON rC.EmployeeId = e.Id

ORDER BY [Name], e.Id

--Only one query must be paste in Judge

SELECT e.Firstname + ' ' + e.Lastname AS [Name],

ISNULL(CONVERT(varchar, CC.ReportSum), '0') + '/' +

ISNULL(CONVERT(varchar, OC.ReportSum), '0') AS [Closed Open Reports]

FROM Employees AS e

JOIN (SELECT EmployeeId, COUNT(\*) AS ReportSum

FROM Reports R

WHERE YEAR(OpenDate) = 2016

GROUP BY EmployeeId) AS OC ON OC.Employeeid = E.Id

LEFT JOIN (SELECT EmployeeId, COUNT(\*) AS ReportSum

FROM Reports R

WHERE YEAR(CloseDate) = 2016

GROUP BY EmployeeId) AS CC ON CC.Employeeid = E.Id

ORDER BY [Name], e.Id

/\* 15AverageClosingTime\_Querying \*/

SELECT d.[Name] AS [Department Name],

ISNULL(CONVERT(VARCHAR, AVG(DATEDIFF(DAY, r.OpenDate, r.CloseDate))), 'no info') AS [Aerage Duration]

FROM Departments AS d

JOIN Categories AS c ON c.DepartmentId = d.Id

LEFT JOIN Reports AS r ON r.CategoryId = c.Id

GROUP BY d.[Name]

ORDER BY d.[Name]

/\* 16FavoiteCategories\_Querying \*/

SELECT [Department Name], [Category Name], [Percentage]

FROM (SELECT d.[Name] AS [Department Name], c.[Name] AS [Category Name],

CAST(ROUND(COUNT(\*) OVER(PARTITION BY c.Id) \* 100.0 / COUNT(\*) OVER(PARTITION BY d.Id), 0) AS INT)

AS [Percentage]

FROM Departments AS d

JOIN Categories AS c ON c.DepartmentId = d.Id

JOIN Reports AS r ON r.CategoryId = c.Id) AS fc

GROUP BY [Department Name], [Category Name], [Percentage]

--ORDER BY [Department Name], [Category Name], [Percentage]

/\*PART IV Programmability \*/

/\* 17EmployeeSLoad\_Programmability \*/

--The query below causes a compile time error in Jugge, but but in database work correctly!!!

CREATE FUNCTION udf\_GetReportsCount(@employeeId INT, @statusId INT)

RETURNS INT AS

BEGIN

DECLARE @reportsCount INT

SET @reportsCount = (SELECT COUNT(r.StatusId)

FROM Employees AS e

JOIN Reports AS r ON r.EmployeeId = e.Id

WHERE e.Id = @employeeId AND r.StatusId = @statusId

GROUP BY r.EmployeeId)

RETURN ISNULL(@reportsCount, 0)

END

--The query above causes a compile time error in Jugge, but but in database work correctly!!!

--In Judge must be paste only one query

CREATE FUNCTION udf\_GetReportsCount(@employeeId INT, @statusId INT)

RETURNS INT AS

BEGIN

DECLARE @reportsCount INT

SET @reportsCount = (SELECT COUNT(\*)

FROM Reports

WHERE EmployeeId = @employeeId AND StatusId = @statusId)

RETURN @reportsCount

END

--In Judge must be paste without this below

GO

SELECT Id, FirstName, Lastname, dbo.udf\_GetReportsCount(Id, 2) AS ReportsCount

FROM Employees

ORDER BY Id

/\* 18AssignEmployee\_Programmability \*/

CREATE PROCEDURE usp\_AssignEmployeeToReport(@employeeId INT, @reportId INT) AS

BEGIN

BEGIN TRANSACTION

DECLARE @reportCategoryId INT = (SELECT CategoryId FROM Reports WHERE Id = @reportId)

DECLARE @employeeDepartmentId INT = (SELECT DepartmentId FROM Employees WHERE Id = @employeeId)

DECLARE @reportCategoryDepartmentId INT = (SELECT DepartmentId FROM Categories

WHERE Id = @reportCategoryId)

UPDATE Reports

SET EmployeeId = @employeeId WHERE Id = @reportId

IF (@employeeDepartmentId <> @reportCategoryDepartmentId)

BEGIN

ROLLBACK

RAISERROR('Employee doesn''t belong to the appropriate department!', 16, 1 )

RETURN

END

COMMIT

END

--In Judge must be paste without this below

EXEC usp\_AssignEmployeeToReport 17, 2

SELECT EmployeeId FROM Reports WHERE Id = 2

/\* 19CloseReports\_Programmability \*/

CREATE TRIGGER tr\_Reports\_CloseReports ON Reports

AFTER UPDATE AS

BEGIN

UPDATE Reports

SET StatusId = (SELECT Id FROM [Status] WHERE Label = 'completed')

WHERE Id IN (SELECT Id FROM inserted

WHERE Id IN (SELECT Id FROM deleted WHERE CloseDate IS NULL)

AND CloseDate IS NOT NULL)

END

--In Judge must be paste without this below

UPDATE Reports

SET CloseDate = GETDATE()

WHERE Id = 5

/\*PART V Bonus \*/

/\* 20CategoriesRevision\_Bonus \*/

SELECT c.[Name] AS [Category Name], COUNT(r.Id) AS [Reports Number],

CASE

WHEN InProgressCount > WaitingCount THEN 'in progress'

WHEN InProgressCount < WaitingCount THEN 'waiting'

ELSE 'equal'

END AS [Main Status]

FROM Reports AS r

JOIN Categories AS c ON c.Id = r.CategoryId

JOIN [Status] AS s ON s.Id = r.StatusId

JOIN (SELECT r.CategoryId, SUM(CASE WHEN s.Label = 'in progress' THEN 1 ELSE 0 END) AS InProgressCount,

SUM(CASE WHEN s.Label = 'waiting' THEN 1 ELSE 0 END) AS WaitingCount

FROM Reports AS r

JOIN [Status] AS s ON s.Id = r.StatusId

WHERE s.Label IN('waiting', 'in progress')

GROUP BY r.CategoryId) AS sc ON sc.CategoryId = c.Id

WHERE s.Label IN('waiting', 'in progress')

GROUP BY c.[Name],

CASE

WHEN InProgressCount > WaitingCount THEN 'in progress'

WHEN InProgressCount < WaitingCount THEN 'waiting'

ELSE 'equal'

END

ORDER BY [Category Name], [Reports Number], [Main Status]