Trabalho - Técnicas avançadas Base de Dados

exec sp\_addlinkedserver

@server = 'RestauranteTABD\_2',

@srvproduct = 'SQLServer Native Client OLEDB Provider',

@provider = 'SQLNCLI11',

@DATASRC = '192.168.157.93'

--Autenticaçao--

exec sp\_addlinkedsrvlogin

@rmtsrvname = 'RestauranteTABD\_2',

@useself = 'false',

@locallogin = 'sa',

@rmtuser = 'sa',

@rmtpassword = '12345'

## Ligação servidor 1

exec sp\_addlinkedserver

@server = 'RestauranteTABD\_1',

@srvproduct = 'SQLServer Native Client OLEDB Provider',

@provider = 'SQLNCLI11',

@DATASRC = '192.168.157.92'

--Autenticaçao

exec sp\_addlinkedsrvlogin

@rmtsrvname = 'RestauranteTABD\_1',

@useself = 'false',

@locallogin = 'sa',

@rmtuser = 'sa',

@rmtpassword = '12345'

CREATE DATABASE RestauranteTABD\_Server1

GO

USE RestauranteTABD\_Server1

GO

CREATE TABLE Utilizador\_AM(

Id INTEGER Not Null,

Pass VARCHAR(20) Not Null,

Username VARCHAR(20) UNIQUE Not Null ,

Email VARCHAR(50) UNIQUE Not Null,

Estado VARCHAR Not Null DEFAULT 0,

PRIMARY KEY(Id, Username),

CHECK(Username < 'N'),

);

CREATE TABLE Administrador(

Id\_A INTEGER Not Null,

criado\_por INTEGER,

PRIMARY KEY(Id\_A),

--Foreign Key(Id\_A) References Utilizador\_AM(Id),

--Foreign Key (criado\_por) References Administrador(Id\_A)

);

CREATE TABLE Cliente(

Id\_C INTEGER Not Null,

Nome VARCHAR(100) Not Null,

Suspenso\_por Integer References Administrador(Id\_A),

PRIMARY KEY(Id\_C),

--Foreign Key(Id\_C) References Utilizador\_AM(Id)

);

Create Table Prato\_AM(

Id\_P INTEGER Not Null,

Nome VARCHAR(50) Not Null,

Tipo VARCHAR(50) Not Null,

PRIMARY KEY(Id\_P, Nome),

CHECK(Nome < 'N'),

);

Create Table Prato\_Dia(

Id\_Pd INTEGER Not Null,

Id\_R INTEGER Not Null,

Preco INTEGER Not Null,

Tipo VARCHAR(20) Not Null,

Descricao VARCHAR(100) Not Null,

Foto VARCHAR(100) Not Null,

Data\_Prato DATE Not Null,

PRIMARY KEY(Id\_Pd, Id\_R),

--Foreign Key(Id\_Pd) References Restaurante(Id\_R),

--Foreign Key(Id\_Pd) References Prato(Id\_P)

);

Create Table Lista\_Fav\_Pratos(

Id\_Cl INTEGER REFERENCES Cliente(Id\_c) Not Null,

Prato\_ID INTEGER, --REFERENCES Prato(Id\_P),

PRIMARY KEY(Id\_cl, Prato\_ID),

Notifica BIT DEFAULT 0,

);

Create Table Lista\_Fav\_Restaurantes(

Id\_c integer references Cliente(Id\_c),

Id\_r INTEGER, --REFERENCES Restaurante(Id\_R),

Primary Key(Id\_r,Id\_c),

Notifica Bit default 0,

);

CREATE DATABASE RestauranteTABD\_Server2

GO

USE RestauranteTABD\_Server2

GO

CREATE TABLE Utilizador\_NZ(

Id INTEGER Not Null,

Pass VARCHAR(20) Not Null,

Username VARCHAR(20) UNIQUE Not Null ,

Email VARCHAR(50) UNIQUE Not Null,

Estado VARCHAR Not Null DEFAULT 0,

PRIMARY KEY(Id, Username),

CHECK(Username >= 'N'),

);

CREATE TABLE Restaurante(

Id\_R INTEGER Not Null,

Nome VARCHAR(100) Not Null,

Telefone INTEGER Not Null,

Foto VARCHAR(100) Not Null,

Localizacao VARCHAR(100) Not Null,

Horario INTEGER Not Null,

Dia\_Desc INTEGER Not Null,

Registado\_por Integer,-- References Administrador(Id\_A),

PRIMARY KEY(Id\_R),

);

Create Table Prato\_NZ(

Id\_P INTEGER Not Null,

Nome VARCHAR(50) Not Null,

Tipo VARCHAR(50) Not Null,

PRIMARY KEY(Id\_P, Nome),

CHECK(Nome >= 'N'),

);

Create Table Suspenso(

Data\_bloqueio DATE DEFAULT GetDate(),

Data\_desbloqueio DATE,

Motivo VARCHAR(50) Not null,

Id\_U Integer, --references Utilizador(Id) Not null,

Id\_Adm Integer, --references Administrador(Id\_A) Not null,

PRIMARY KEY(Id\_U, Id\_Adm)

);

CREATE TABLE Tipo\_Servico(

Id\_Tp INTEGER IDENTITY (1,1) PRIMARY KEY,

Nome VARCHAR(20) Not Null

);

CREATE TABLE Rest\_Tipo(

Id\_R INTEGER Not Null REFERENCES Restaurante(Id\_R) ,

Id\_Tp INTEGER Not Null REFERENCES Tipo\_Servico(Id\_Tp),

PRIMARY KEY (Id\_R, Id\_Tp)

);

CREATE VIEW Utilizador

AS

SELECT \* FROM Utilizador\_AM

UNION ALL

SELECT \* FROM RestauranteTABD\_2.RestauranteTABD\_Server2.dbo.UtilizadoresNZ

GO

CREATE VIEW Restaurante

AS

SELECT \* FROM RestauranteTABD\_2.RestauranteTABD\_Server2.dbo.Restaurante

GO

CREATE VIEW Prato

AS

SELECT \* FROM Prato\_AM

UNION ALL

SELECT \* FROM RestauranteTABD\_2.RestauranteTABD\_Server2.dbo.Prato\_NZ

GO

CREATE VIEW Suspenso

AS

SELECT \* FROM RestauranteTABD\_2.RestauranteTABD\_Server2.dbo.Suspenso

GO

CREATE VIEW Tipo\_Servico

AS

SELECT \* FROM RestauranteTABD\_2.RestauranteTABD\_Server2.dbo.Tipo\_Servico

GO

CREATE VIEW Rest\_Tipo

AS

SELECT \* FROM RestauranteTABD\_2.RestauranteTABD\_Server2.dbo.Rest\_Tipo

GO

CREATE VIEW Utilizador

AS

SELECT \* FROM Utilizador\_NZ

UNION ALL

SELECT \* FROM RestauranteTABD\_1.RestauranteTABD\_Server1.dbo.UtilizadoresAM

GO

CREATE VIEW Administrador

AS

SELECT \* FROM RestauranteTABD\_1.RestauranteTABD\_Server1.dbo.Administrador

GO

CREATE VIEW Cliente

AS

SELECT \* FROM RestauranteTABD\_1.RestauranteTABD\_Server1.dbo.Cliente

GO

CREATE VIEW Prato

AS

SELECT \* FROM Prato\_NZ

UNION ALL

SELECT \* FROM RestauranteTABD\_1.RestauranteTABD\_Server1.dbo.PratoAM

GO

CREATE VIEW Prato\_Dia

AS

SELECT \* FROM RestauranteTABD\_1.RestauranteTABD\_Server1.dbo.Prato\_Dia

GO

CREATE VIEW Lista\_Fav\_Pratos

AS

SELECT \* FROM RestauranteTABD\_1.RestauranteTABD\_Server1.dbo.Lista\_Fav\_Pratos

GO

CREATE VIEW Lista\_Fav\_Restaurantes

AS

SELECT \* FROM RestauranteTABD\_1.RestauranteTABD\_Server1.dbo.Lista\_Fav\_Restaurantes

GO

--Logins servidor 1--

CREATE LOGIN UtizadorAnonimo WITH PASSWORD = 'anonimo'

CREATE LOGIN Restaurante WITH PASSWORD = 'restaurante'

CREATE LOGIN Cliente WITH PASSWORD = 'cliente'

CREATE LOGIN Administrador WITH PASSWORD = 'administrador'

--Users Servidor 1--

CREATE USER UtilizadoresAnonimos FROM LOGIN UtizadorAnonimo

CREATE USER Restaurantes FROM LOGIN Restaurante

CREATE USER Clientes FROM LOGIN Cliente

CREATE USER Administradores FROM LOGIN Administrador

--Permissões Utilizador Anonimo--

GRANT SELECT, INSERT ON Utilizador TO UtilizadoresAnonimos

GRANT SELECT ON Prato\_Dia TO UtilizadoresAnonimos

GRANT SELECT ON Restaurante TO UtilizadoresAnonimos

--Permissões Clientes--

GRANT SELECT ON Utilizador TO Clientes

GRANT UPDATE ON Utilizador (Pass, Username, Email) TO Clientes

GRANT SELECT ON Cliente TO Clientes

GRANT UPDATE ON Cliente(Nome) TO Clientes

GRANT SELECT ON Prato\_Dia TO Clientes

GRANT SELECT ON Restaurante TO Clientes

GRANT INSERT, SELECT, DElETE ON Lista\_Fav\_Pratos TO Clientes

GRANT UPDATE ON Lista\_Fav\_Pratos(Notifica) TO Clientes

GRANT INSERT, SELECT, DELETE ON Lista\_Fav\_Restaurantes TO Clientes

GRANT UPDATE ON Lista\_Fav\_Restaurantes(Notifica) TO Clientes

--Permissões Administrador--

GRANT SELECT ON Utilizador TO Administradores

GRANT UPDATE ON Utilizador (Estado) TO Administradores

GRANT SELECT ON Cliente TO Administradores

GRANT SELECT ON Restaurante TO Administradores

GRANT INSERT, SELECT ON Administrador TO Administradores

GRANT UPDATE ON Utilizador(Estado) TO Administradores

GRANT INSERT, SELECT ON Suspenso TO Administradores

--Logins Servidor 2--

CREATE LOGIN UtizadorAnonimo WITH PASSWORD = 'anonimo'

CREATE LOGIN Restaurante WITH PASSWORD = 'restaurante'

CREATE LOGIN Cliente WITH PASSWORD = 'cliente'

CREATE LOGIN Administrador WITH PASSWORD = 'administrador'

--Users Servidor 2--

CREATE USER UtilizadoresAnonimos FROM LOGIN UtizadorAnonimo

CREATE USER Restaurantes FROM LOGIN Restaurante

CREATE USER Clientes FROM LOGIN Cliente

CREATE USER Administradores FROM LOGIN Administrador

--Permissões Utilizador Anonimo--

GRANT SELECT, INSERT ON Utilizador TO UtilizadoresAnonimos

GRANT SELECT ON Prato\_Dia TO UtilizadoresAnonimos

GRANT SELECT ON Restaurante TO UtilizadoresAnonimos

--Permissões Restaurantes--

GRANT SELECT ON Utilizador TO Restaurantes

GRANT UPDATE ON Utilizador (Pass, Username, Email) TO Restaurantes

GRANT SELECT ON Restaurante TO Restaurantes

GRANT UPDATE ON Restaurante (Nome, Telefone, Foto, Localizacao, Horario, Dia\_Desc, Tipo\_Servico) TO Restaurantes

GRANT INSERT, SELECT ON Prato TO Restaurantes

GRANT INSERT, SELECT ON Prato\_Dia TO Restaurantes

GRANT UPDATE ON Prato\_Dia (Preco, Tipo, Descricao, Foto, Data\_Prato) TO Restaurantes

GRANT SELECT ON Tipo\_Servico TO Restaurantes

GRANT SELECT ON Rest\_Tipo TO Restaurantes

Procedimentos Servidor 1

--Adicionar Utilizador--

CREATE PROCEDURE CriarUitilizador @Pass VARCHAR(20), @Username VARCHAR(20), @Email VARCHAR(50)

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

DECLARE @max INTEGER

SELECT @max = MAX(Id) FROM Utilizador

IF(@max IS NULL)

SET @max = 0

DECLARE @username\_aux VARCHAR(20)

SELECT @username\_aux

INSERT INTO Utilizador(Id,Pass, Username, Email)

VALUES(@max + 1,@Pass, @Username, @Email)

IF(@@ERROR <> 0) GOTO ErroCriarUtilizador

COMMIT

RETURN @max + 1

ErroCriarUtilizador:

ROLLBACK

RETURN -1

--Criar Cliente--

CREATE PROCEDURE CriarCliente @Pass VARCHAR(20), @Username VARCHAR(20), @Email VARCHAR(50), @Nome VARCHAR(50)

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

DECLARE @id INTEGER

EXECUTE @id = CriarUitilizador @Pass, @Username, @Email

IF(@id = -1) GOTO ErroCriarCliente

INSERT INTO Cliente(Id\_C, Nome)

VALUES(@id, @Nome)

IF(@@ERROR <> 0) GOTO ErroCriarCliente

COMMIT

RETURN 1

ErroCriarCliente:

ROLLBACK

RETURN -1

--Criar Tipo de Restaurante--

CREATE PROCEDURE CriarTipo @Id INTEGER, @Nome VARCHAR(20)

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

INSERT INTO Tipo\_Servico(Id\_Tp, Nome)

VALUES(@@IDENTITY, @Nome)

IF(@@ERROR <> 0) GOTO ErroCriarTipo

COMMIT

RETURN 1

ErroCriarTipo:

ROLLBACK

RETURN -1

--Associar Tipo ao Restaurante--

CREATE PROCEDURE AssociarTipoRest @Id\_Rest INTEGER, @Id\_Tipo INTEGER

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

INSERT INTO Rest\_Tipo(Id\_R, Id\_Tp)

VALUES(@Id\_Rest, @Id\_Tipo)

IF(@@ERROR <> 0) GOTO ErroAssociar

COMMIT

RETURN 1

ErroAssociar:

ROLLBACK

RETURN -1

--Adicionar Restaurante--

CREATE PROCEDURE CriarRestaurante @Id INTEGER, @Nome VARCHAR(100), @Telefone INTEGER, @Foto VARCHAR(100),

@Localizacao VARCHAR(100), @Horario INTEGER, @Dia\_Desc INTEGER, @Tipo\_Servico VARCHAR(50)

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

INSERT INTO Restaurante(Id\_R, Nome, Telefone, Foto, Localizacao, Horario, Dia\_Desc, Tipo\_Servico)

VALUES(@Id, @Nome, @Telefone, @Foto, @Localizacao, @Horario, @Dia\_Desc, @Tipo\_Servico)

IF(@@ERROR <> 0) GOTO ErroCriarRestaurante

COMMIT

RETURN 1

ErroCriarRestaurante:

ROLLBACK

RETURN -1

--Adicionar Admin--

CREATE PROCEDURE CriarAdmin @Pass VARCHAR(20), @Username VARCHAR(20), @Email VARCHAR(50)

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

DECLARE @id INTEGER

EXECUTE @id = CriarUitilizador @Pass, @Username, @Email

IF(@id = -1) GOTO ErroCriarAdministrador

INSERT INTO Administrador(Id\_A)

VALUES(@id)

IF(@@ERROR <> 0) GOTO ErroCriarAdministrador

COMMIT

RETURN 1

ErroCriarAdministrador:

ROLLBACK

RETURN -1

--Criar Prato--

CREATE PROCEDURE CriarPrato @Nome VARCHAR(50), @Tipo VARCHAR(50)

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

DECLARE @max INTEGER

SELECT @max = MAX(Id\_P) FROM Prato

IF(@max IS NULL)

SET @max = 0

INSERT INTO Prato(Id\_P,Nome, Tipo)

VALUES(@max + 1,@Nome, @Tipo)

IF(@@ERROR <> 0) GOTO ErroCriarPrato

COMMIT

RETURN @max + 1

ErroCriarPrato:

ROLLBACK

RETURN -1

--Criar Prato do Dia

CREATE PROCEDURE CriarPratoDia @Id\_Prato INTEGER, @Id\_Rest INTEGER, @Preco INTEGER, @Tipo VARCHAR(20), @Descricao VARCHAR(100),

@Foto VARCHAR(100), @Data\_Prato DATE

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

INSERT INTO Prato\_Dia(Id\_Pd, Id\_R, Preco, Tipo, Descricao ,Foto, Data\_Prato)

VALUES (@Id\_Prato, @Id\_Rest, @Preco, @Tipo, @Descricao, @Foto, @Data\_Prato)

IF(@@ERROR <> 0) GOTO ErroCriarPratoDia

COMMIT

RETURN 1

ErroCriarPratoDia:

ROLLBACK

RETURN -1

--Adicionar Restaurante Favorito à Lista--

CREATE PROCEDURE AdicionarRestauranteFavorito @Id\_Rest INTEGER, @Id\_Cliente INTEGER, @Notifica BIT

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

INSERT INTO Lista\_Fav\_Restaurantes(Id\_r, Id\_c, Notifica)

VALUES (@Id\_Rest, @Id\_Cliente, @Notifica)

IF(@@ERROR <> 0) GOTO ErroCriarRestFav

COMMIT

RETURN 1

ErroCriarRestFav:

ROLLBACK

RETURN -1

--Adicionar Prato Favorito à Lista--

CREATE PROCEDURE AdicionarPratoFavorito @Id\_Prato INTEGER, @Id\_Cliente INTEGER, @Notifica BIT

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

INSERT INTO Lista\_Fav\_Pratos(Prato\_ID, Id\_Cl, Notifica)

VALUES (@Id\_Prato, @Id\_Cliente, @Notifica)

IF(@@ERROR <> 0) GOTO ErroCriarPratoFav

COMMIT

RETURN 1

ErroCriarPratoFav:

ROLLBACK

RETURN -1

--Restaurante Alterar Data--

CREATE PROCEDURE AlterarData @Id\_Prato INTEGER, @NovaData DATE

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

UPDATE Prato\_Dia

SET Data\_Prato = @NovaData

WHERE Id\_Pd = @Id\_Prato

IF(@@ERROR <> 0 OR @@ROWCOUNT = 0) GOTO ErroAlterarData

COMMIT

RETURN 1

ErroAlterarData:

ROLLBACK

RETURN -1

--Restaurante Remover Prato do Dia--

CREATE PROCEDURE RemoverPrato @Id\_Prato INTEGER, @Id\_Rest INTEGER

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

DELETE FROM Prato\_Dia

WHERE Id\_Pd = @Id\_Prato AND Id\_R = @Id\_Rest

IF(@@ERROR <> 0) GOTO ErroRemoverPrato

COMMIT

RETURN 1

ErroRemoverPrato:

ROLLBACK

RETURN -1

--Cliente Alterar Notificaçao Prato Favorito--

CREATE PROCEDURE AlterarNotificaçaoPrato @Id\_Cliente INTEGER ,@Id\_Prato INTEGER, @Notifica BIT

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

UPDATE Lista\_Fav\_Pratos

SET Notifica = @Notifica

WHERE Id\_Cl = @Id\_Cliente AND Prato\_ID = @Id\_Prato

IF(@@ERROR <> 0) GOTO ErroAlterarNotiPrato

COMMIT

RETURN 1

ErroAlterarNotiPrato:

ROLLBACK

RETURN -1

--Cliente Remover Alterar Restaurante Favorito

CREATE PROCEDURE AlterarNotificaçaoRest @Id\_Cliente INTEGER ,@Id\_Rest INTEGER, @Notifica BIT

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

UPDATE Lista\_Fav\_Restaurantes

SET Notifica = @Notifica

WHERE Id\_c = @Id\_Cliente AND Id\_r = @Id\_Rest

IF(@@ERROR <> 0 OR @@ROWCOUNT = 0) GOTO ErroAlterarNotiRest

COMMIT

RETURN 1

ErroAlterarNotiRest:

ROLLBACK

RETURN -1

--Cliente Remover Prato dos Favoritos--

CREATE PROCEDURE RemoverPratoFav @Id\_Cliente INTEGER ,@Id\_Prato INTEGER

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

DELETE FROM Lista\_Fav\_Pratos

WHERE Id\_Cl = @Id\_Cliente AND Prato\_ID = @Id\_Prato

IF(@@ERROR <> 0 OR @@ROWCOUNT <> 0) GOTO ErroRemoverPratoFav

COMMIT

RETURN 1

ErroRemoverPratoFav:

ROLLBACK

RETURN -1

--Cliente Remover Restaurante dos Favoritos--

CREATE PROCEDURE RemoverRestFav @Id\_Cliente INTEGER ,@Id\_Rest INTEGER

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

DELETE FROM Lista\_Fav\_Restaurantes

WHERE Id\_c = @Id\_Cliente AND Id\_r = @Id\_Rest

IF(@@ERROR <> 0) GOTO ErroAlterarNotiRest

COMMIT

RETURN 1

ErroRemoverRestFav:

ROLLBACK

RETURN -1

--Suspender Utilizador--

CREATE PROCEDURE SuspenderUtilizador @Id\_Utilizador INTEGER ,@Id\_Admin INTEGER, @Motivo VARCHAR, @Data\_Desbloqueio DATE

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

INSERT INTO Suspenso(Id\_U, Id\_Adm, Motivo, Data\_bloqueio, Data\_desbloqueio)

VALUES(@Id\_Utilizador, @Id\_Admin, @Motivo, GETDATE(), @Data\_Desbloqueio)

IF(@@ERROR <> 0 OR @@ROWCOUNT = 0) GOTO ErroSuspenderUtilizador

UPDATE Utilizador

SET Estado = -1

WHERE Id = @Id\_Utilizador

IF(@@ERROR <> 0 OR @@ROWCOUNT = 0) GOTO ErroSuspenderUtilizador

COMMIT

RETURN 1

ErroSuspenderUtilizador:

ROLLBACK

RETURN -1

--Desbloquear Utilizador--

CREATE PROCEDURE DesbloquearUtilizador @Id\_Utilizador INTEGER ,@Id\_Admin INTEGER

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

DELETE FROM Suspenso

WHERE Id\_U = @Id\_Utilizador

IF(@@ERROR <> 0 OR @@ROWCOUNT = 0) GOTO ErroSuspenderUtilizador

UPDATE Utilizador

SET Estado = 1

WHERE Id = @Id\_Utilizador

IF(@@ERROR <> 0 OR @@ROWCOUNT = 0) GOTO ErroSuspenderUtilizador

COMMIT

RETURN 1

ErroDesbloquearUtilizador:

ROLLBACK

RETURN -1

--Adicionar Utilizador—procedimentos servidor 2

CREATE PROCEDURE CriarUitilizador @Pass VARCHAR(20), @Username VARCHAR(20), @Email VARCHAR(50)

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

DECLARE @max INTEGER

SELECT @max = MAX(Id) FROM Utilizador

IF(@max IS NULL)

SET @max = 0

DECLARE @username\_aux VARCHAR(20)

SELECT @username\_aux

INSERT INTO Utilizador(Id,Pass, Username, Email)

VALUES(@max + 1,@Pass, @Username, @Email)

IF(@@ERROR <> 0) GOTO ErroCriarUtilizador

COMMIT

RETURN @max + 1

ErroCriarUtilizador:

ROLLBACK

RETURN -1

--Criar Cliente--

CREATE PROCEDURE CriarCliente @Pass VARCHAR(20), @Username VARCHAR(20), @Email VARCHAR(50), @Nome VARCHAR(50)

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

DECLARE @id INTEGER

EXECUTE @id = CriarUitilizador @Pass, @Username, @Email

IF(@id = -1) GOTO ErroCriarCliente

INSERT INTO Cliente(Id\_C, Nome)

VALUES(@id, @Nome)

IF(@@ERROR <> 0) GOTO ErroCriarCliente

COMMIT

RETURN 1

ErroCriarCliente:

ROLLBACK

RETURN -1

--Criar Tipo de Restaurante--

CREATE PROCEDURE CriarTipo @Id INTEGER, @Nome VARCHAR(20)

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

INSERT INTO Tipo\_Servico(Id\_Tp, Nome)

VALUES(@@IDENTITY, @Nome)

IF(@@ERROR <> 0) GOTO ErroCriarTipo

COMMIT

RETURN 1

ErroCriarTipo:

ROLLBACK

RETURN -1

--Associar Tipo ao Restaurante--

CREATE PROCEDURE AssociarTipoRest @Id\_Rest INTEGER, @Id\_Tipo INTEGER

AS

SET TRANSACTION ISOLATION LEVEL READ SERIALIZABLE

BEGIN DISTRIBUTED TRANSACTION

INSERT INTO Rest\_Tipo(Id\_R, Id\_Tp)

VALUES(@Id\_Rest, @Id\_Tipo)

IF(@@ERROR <> 0) GOTO ErroAssociar

COMMIT

RETURN 1

ErroAssociar:

ROLLBACK

RETURN -1

--Adicionar Restaurante--

CREATE PROCEDURE CriarRestaurante @Id INTEGER, @Nome VARCHAR(100), @Telefone INTEGER, @Foto VARCHAR(100),

@Localizacao VARCHAR(100), @Horario INTEGER, @Dia\_Desc INTEGER, @Tipo\_Servico VARCHAR(50)

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

INSERT INTO Restaurante(Id\_R, Nome, Telefone, Foto, Localizacao, Horario, Dia\_Desc, Tipo\_Servico)

VALUES(@Id, @Nome, @Telefone, @Foto, @Localizacao, @Horario, @Dia\_Desc, @Tipo\_Servico)

IF(@@ERROR <> 0) GOTO ErroCriarRestaurante

COMMIT

RETURN 1

ErroCriarRestaurante:

ROLLBACK

RETURN -1

--Adicionar Admin—

CREATE PROCEDURE CriarAdmin @Pass VARCHAR(20), @Username VARCHAR(20), @Email VARCHAR(50)

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

DECLARE @id INTEGER

EXECUTE @id = CriarUitilizador @Pass, @Username, @Email

IF(@id = -1) GOTO ErroCriarAdministrador

INSERT INTO Administrador(Id\_A)

VALUES(@id)

IF(@@ERROR <> 0) GOTO ErroCriarAdministrador

COMMIT

RETURN 1

ErroCriarAdministrador:

ROLLBACK

RETURN -1

--Criar Prato--

CREATE PROCEDURE CriarPrato @Nome VARCHAR(50), @Tipo VARCHAR(50)

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

DECLARE @max INTEGER

SELECT @max = MAX(Id\_P) FROM Prato

IF(@max IS NULL)

SET @max = 0

INSERT INTO Prato(Id\_P,Nome, Tipo)

VALUES(@max + 1,@Nome, @Tipo)

IF(@@ERROR <> 0) GOTO ErroCriarPrato

COMMIT

RETURN @max + 1

ErroCriarPrato:

ROLLBACK

RETURN -1

--Criar Prato do Dia

CREATE PROCEDURE CriarPratoDia @Id\_Prato INTEGER, @Id\_Rest INTEGER, @Preco INTEGER, @Tipo VARCHAR(20), @Descricao VARCHAR(100),

@Foto VARCHAR(100), @Data\_Prato DATE

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

INSERT INTO Prato\_Dia(Id\_Pd, Id\_R, Preco, Tipo, Descricao ,Foto, Data\_Prato)

VALUES (@Id\_Prato, @Id\_Rest, @Preco, @Tipo, @Descricao, @Foto, @Data\_Prato)

IF(@@ERROR <> 0) GOTO ErroCriarPratoDia

COMMIT

RETURN 1

ErroCriarPratoDia:

ROLLBACK

RETURN -1

--Adicionar Restaurante Favorito à Lista--

CREATE PROCEDURE AdicionarRestauranteFavorito @Id\_Rest INTEGER, @Id\_Cliente INTEGER, @Notifica BIT

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

INSERT INTO Lista\_Fav\_Restaurantes(Id\_r, Id\_c, Notifica)

VALUES (@Id\_Rest, @Id\_Cliente, @Notifica)

IF(@@ERROR <> 0) GOTO ErroCriarRestFav

COMMIT

RETURN 1

ErroCriarRestFav:

ROLLBACK

RETURN -1

--Adicionar Prato Favorito à Lista--

CREATE PROCEDURE AdicionarPratoFavorito @Id\_Prato INTEGER, @Id\_Cliente INTEGER, @Notifica BIT

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

INSERT INTO Lista\_Fav\_Pratos(Prato\_ID, Id\_Cl, Notifica)

VALUES (@Id\_Prato, @Id\_Cliente, @Notifica)

IF(@@ERROR <> 0) GOTO ErroCriarPratoFav

COMMIT

RETURN 1

ErroCriarPratoFav:

ROLLBACK

RETURN -1

--Restaurante Alterar Data--

CREATE PROCEDURE AlterarData @Id\_Prato INTEGER, @NovaData DATE

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

UPDATE Prato\_Dia

SET Data\_Prato = @NovaData

WHERE Id\_Pd = @Id\_Prato

IF(@@ERROR <> 0 OR @@ROWCOUNT = 0) GOTO ErroAlterarData

COMMIT

RETURN 1

ErroAlterarData:

ROLLBACK

RETURN -1

--Restaurante Remover Prato do Dia--

CREATE PROCEDURE RemoverPrato @Id\_Prato INTEGER, @Id\_Rest INTEGER

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

DELETE FROM Prato\_Dia

WHERE Id\_Pd = @Id\_Prato AND Id\_R = @Id\_Rest

IF(@@ERROR <> 0) GOTO ErroRemoverPrato

COMMIT

RETURN 1

ErroRemoverPrato:

ROLLBACK

RETURN -1

--Cliente Alterar Notificaçao Prato Favorito--

CREATE PROCEDURE AlterarNotificaçaoPrato @Id\_Cliente INTEGER ,@Id\_Prato INTEGER, @Notifica BIT

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

UPDATE Lista\_Fav\_Pratos

SET Notifica = @Notifica

WHERE Id\_Cl = @Id\_Cliente AND Prato\_ID = @Id\_Prato

IF(@@ERROR <> 0) GOTO ErroAlterarNotiPrato

COMMIT

RETURN 1

ErroAlterarNotiPrato:

ROLLBACK

RETURN -1

--Cliente Remover Alterar Restaurante Favorito

CREATE PROCEDURE AlterarNotificaçaoRest @Id\_Cliente INTEGER ,@Id\_Rest INTEGER, @Notifica BIT

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

UPDATE Lista\_Fav\_Restaurantes

SET Notifica = @Notifica

WHERE Id\_c = @Id\_Cliente AND Id\_r = @Id\_Rest

IF(@@ERROR <> 0 OR @@ROWCOUNT = 0) GOTO ErroAlterarNotiRest

COMMIT

RETURN 1

ErroAlterarNotiRest:

ROLLBACK

RETURN -1

--Cliente Remover Prato dos Favoritos--

CREATE PROCEDURE RemoverPratoFav @Id\_Cliente INTEGER ,@Id\_Prato INTEGER

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

DELETE FROM Lista\_Fav\_Pratos

WHERE Id\_Cl = @Id\_Cliente AND Prato\_ID = @Id\_Prato

IF(@@ERROR <> 0 OR @@ROWCOUNT <> 0) GOTO ErroRemoverPratoFav

COMMIT

RETURN 1

ErroRemoverPratoFav:

ROLLBACK

RETURN -1

--Cliente Remover Restaurante dos Favoritos--

CREATE PROCEDURE RemoverRestFav @Id\_Cliente INTEGER ,@Id\_Rest INTEGER

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

DELETE FROM Lista\_Fav\_Restaurantes

WHERE Id\_c = @Id\_Cliente AND Id\_r = @Id\_Rest

IF(@@ERROR <> 0) GOTO ErroAlterarNotiRest

COMMIT

RETURN 1

ErroRemoverRestFav:

ROLLBACK

RETURN -1

--Suspender Utilizador--

CREATE PROCEDURE SuspenderUtilizador @Id\_Utilizador INTEGER ,@Id\_Admin INTEGER, @Motivo VARCHAR, @Data\_Desbloqueio DATE

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

INSERT INTO Suspenso(Id\_U, Id\_Adm, Motivo, Data\_bloqueio, Data\_desbloqueio)

VALUES(@Id\_Utilizador, @Id\_Admin, @Motivo, GETDATE(), @Data\_Desbloqueio)

IF(@@ERROR <> 0 OR @@ROWCOUNT = 0) GOTO ErroSuspenderUtilizador

UPDATE Utilizador

SET Estado = -1

WHERE Id = @Id\_Utilizador

IF(@@ERROR <> 0 OR @@ROWCOUNT = 0) GOTO ErroSuspenderUtilizador

COMMIT

RETURN 1

ErroSuspenderUtilizador:

ROLLBACK

RETURN -1

--Desbloquear Utilizador--

CREATE PROCEDURE DesbloquearUtilizador @Id\_Utilizador INTEGER ,@Id\_Admin INTEGER

AS

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN DISTRIBUTED TRANSACTION

DELETE FROM Suspenso

WHERE Id\_U = @Id\_Utilizador

IF(@@ERROR <> 0 OR @@ROWCOUNT = 0) GOTO ErroSuspenderUtilizador

UPDATE Utilizador

SET Estado = 1

WHERE Id = @Id\_Utilizador

IF(@@ERROR <> 0 OR @@ROWCOUNT = 0) GOTO ErroSuspenderUtilizador

COMMIT

RETURN 1

ErroDesbloquearUtilizador:

ROLLBACK

RETURN -1