```
/* Задание.
Создание очереди
Цель: Использовать очередь Настроить сервер для работы с очередями Написать скрипты
   для создания и настройки очереди
Создание очереди в БД для фоновой обработки задачи в БД.
Подумайте и реализуйте очередь в рамках своего проекта. */

/* Краткое описание и файлы проекта находятся здесь:
https://github.com/sburovsky/-otus-mssql-2020-02-burovsky/tree/master/Project
в том числе, скрипты по созданию всех таблиц и индексов:
https://github.com/sburovsky/-otus-mssql-2020-02-burovsky/tree/master/Project/
  OtusProjectDDL.sql
скрипты по созданию функций и процедур:
https://github.com/sburovsky/-otus-mssql-2020-02-burovsky/tree/master/Project/
  OtusProjectDML.sql

Схема проекта:
https://dbdesigner.page.link/EyBwQYZnkuVvwxpg6

Краткое описание функционала очереди:
Очередь получает и записывает в таблицу рейтингов рейтинги студента согласно его
  баллам за контрольные работы.
Рейтинги рассчитываются функцией Learning.RatingCount. Для каждого студента по
  соответствующим формулам рассчитываются пять рейтингов по пяти типам:
- средняя оценка,
- средневзвешенная оценка,
- оценка по баварской системе
- медианная оценка
- медианная дискретная оценка
Рассчитанные таким образом рейтинги отправляются в очередь одной строкой.
При получении данных по студенту из очереди данные разворачиваются построчно по
  типам рейтинга и записываются в таблицу рейтингов.
Также для контроля записывается реквизит даты и времени получения рейтинга из
  очереди */

USE master
GO
ALTER DATABASE TutorsWorkspace SET SINGLE_USER WITH ROLLBACK IMMEDIATE

ALTER DATABASE TutorsWorkspace
SET ENABLE_BROKER;

ALTER DATABASE TutorsWorkspace SET TRUSTWORTHY ON;
-- контроль
select DATABASEPROPERTYEX ('TutorsWorkspace','UserAccess');
SELECT is_broker_enabled FROM sys.databases WHERE name = 'TutorsWorkspace';

ALTER AUTHORIZATION
   ON DATABASE::TutorsWorkspace TO [sa];

ALTER DATABASE TutorsWorkspace SET MULTI_USER WITH ROLLBACK IMMEDIATE
GO

--добавим поле, чтобы фиксировать получение данных из очереди
use TutorsWorkspace;
```

```
ALTER TABLE Controls.Ratings
ADD RatingConfirmedForProcessing DATETIME2;

-- типы сообщений
CREATE MESSAGE TYPE
[//TW/Controls/RequestRatingMessage]
VALIDATION=WELL_FORMED_XML;

CREATE MESSAGE TYPE
[//TW/Controls/ReplyRatingMessage]
VALIDATION=WELL_FORMED_XML;
GO

-- контракт
CREATE CONTRACT [//TW/Controls/RatingContract]
      ([//TW/Controls/RequestRatingMessage]
        SENT BY INITIATOR,
       [//TW/Controls/ReplyRatingMessage]
        SENT BY TARGET
      );
GO

--очередь
CREATE QUEUE TargetRatingQueue;

--сервисы
CREATE SERVICE [//TW/Controls/TargetRatingService]
       ON QUEUE TargetRatingQueue
       ([//TW/Controls/RatingContract]);
GO

CREATE QUEUE InitiatorRatingQueue;

CREATE SERVICE [//TW/Controls/InitiatorRatingService]
       ON QUEUE InitiatorRatingQueue
       ([//TW/Controls/RatingContract]);
GO

DROP PROC IF EXISTS Controls.SendStudentRatings;
GO

CREATE PROCEDURE Controls.SendStudentRatings
    @StudentId INT
AS
-- отправка рейтинга в очередь
BEGIN
    SET NOCOUNT ON;

    DECLARE @InitDlgHandle UNIQUEIDENTIFIER;
    DECLARE @RequestMessage NVARCHAR(4000);

    BEGIN TRAN

    SELECT @RequestMessage = (SELECT * from Learning.RatingCount(@StudentId)
                              FOR XML AUTO, root('RequestMessage'));
```

```
    BEGIN DIALOG @InitDlgHandle
    FROM SERVICE
    [//TW/Controls/InitiatorRatingService]
    TO SERVICE
    '//TW/Controls/TargetRatingService'
    ON CONTRACT
    [//TW/Controls/RatingContract]
    WITH ENCRYPTION=OFF;

    --Send the Message
    SEND ON CONVERSATION @InitDlgHandle
    MESSAGE TYPE
    [//TW/Controls/RequestRatingMessage]
    (@RequestMessage);
    COMMIT TRAN
END
GO

DROP PROC IF EXISTS Controls.GetStudentRatings;
GO

CREATE PROCEDURE Controls.GetStudentRatings
AS
-- получение рейтинга из очереди
BEGIN

    DECLARE @TargetDlgHandle UNIQUEIDENTIFIER, --идентификатор диалога
            @Message NVARCHAR(4000),--полученное сообщение
            @MessageType Sysname,--тип полученного сообщения
            @ReplyMessage NVARCHAR(4000),--ответное сообщение
            @xml XML;

    BEGIN TRAN;

    --Receive message from Initiator
    RECEIVE TOP(1)
        @TargetDlgHandle = Conversation_Handle,
        @Message = Message_Body,
        @MessageType = Message_Type_Name
    FROM dbo.TargetRatingQueue;

    SELECT @Message; --выводим в консоль полученный месседж

    SET @xml = CAST(@Message AS XML);

    -- получаем данные по рейтингу, разворачиваем по строкам и записываем в таблицу
       рейтингов
    WITH cteRatings AS (Select StudentID, RatingTypeID, Rating
    FROM (
     Select
        Nod.Loc.value('@StudentID', 'int') AS StudentID,
        Nod.Loc.value('@GPA', 'decimal(10,2)') AS GPA,
        Nod.Loc.value('@GPWA', 'decimal(10,2)') AS GPWA,
        Nod.Loc.value('@GPA_Germany', 'decimal(10,2)') AS 'GPA Germany',
        Nod.Loc.value('@Median_Mark', 'decimal(10,2)') AS 'Median Mark',
        Nod.Loc.value('@Median_Discrete_Mark', 'decimal(10,2)') AS 'Median Discrete
```

```
            Mark'
      FROM @xml.nodes('/RequestMessage/Learning.RatingCount') as Nod(Loc)) AS
        Ratings
      UNPIVOT (Rating FOR RatingName IN (GPA, GPWA, "GPA Germany", "Median Mark",
        "Median Discrete Mark" )) AS unpt
       INNER JOIN Controls.RatingTypes as rt
        ON rt.RatingName = unpt.RatingName)

    MERGE [Controls].[Ratings] AS target
        USING (Select StudentID, RatingTypeID, Rating FROM cteRatings)
            AS source (StudentID, RatingTypeID, Rating)
            ON
         (target.StudentID = source.StudentID AND target.RatingTypeID =
           source.RatingTypeID)
        WHEN MATCHED
            THEN UPDATE SET rating = source.rating,
                        RatingConfirmedForProcessing = GetDate()
        WHEN NOT MATCHED
            THEN INSERT (RatingID, StudentID, RatingTypeID, Rating,
              RatingConfirmedForProcessing)
                VALUES (default, source.StudentID, source.RatingTypeID,
                  source.Rating, GetDate());


    IF @MessageType=N'//TW/Controls/RequestRatingMessage'
    BEGIN
        SET @ReplyMessage =N'<ReplyMessage> Message received</ReplyMessage>';

        SEND ON CONVERSATION @TargetDlgHandle
        MESSAGE TYPE
        [//TW/Controls/ReplyRatingMessage]
        (@ReplyMessage);
        END CONVERSATION @TargetDlgHandle;
    END

    COMMIT TRAN;
END
GO

DROP PROCEDURE IF EXISTS Controls.ConfirmStudentRatings
GO

CREATE PROCEDURE Controls.ConfirmStudentRatings
AS
-- получение подтверждения
BEGIN
    DECLARE @InitiatorReplyDlgHandle UNIQUEIDENTIFIER,
            @ReplyReceivedMessage NVARCHAR(1000)

    BEGIN TRAN;
        RECEIVE TOP(1)
            @InitiatorReplyDlgHandle=Conversation_Handle
            ,@ReplyReceivedMessage=Message_Body
        FROM dbo.InitiatorRatingQueue;

        END CONVERSATION @InitiatorReplyDlgHandle;
```

```
        SELECT @ReplyReceivedMessage AS ReceivedRepliedMessage; --в консоль

    COMMIT TRAN;
END

-- активируем очереди

ALTER QUEUE [dbo].[InitiatorRatingQueue] WITH STATUS = ON , RETENTION = OFF ,
  POISON_MESSAGE_HANDLING (STATUS = OFF)
    , ACTIVATION (   STATUS = ON ,
        PROCEDURE_NAME = Controls.ConfirmStudentRatings, MAX_QUEUE_READERS = 100,
          EXECUTE AS OWNER) ;

GO
ALTER QUEUE [dbo].[TargetRatingQueue] WITH STATUS = ON , RETENTION = OFF ,
  POISON_MESSAGE_HANDLING (STATUS = OFF)
    , ACTIVATION (  STATUS = ON ,
        PROCEDURE_NAME = Controls.GetStudentRatings, MAX_QUEUE_READERS = 100,
          EXECUTE AS OWNER) ;

GO

-- рассчитаем и отправим в очередь рейтинг для студента
EXEC Controls.SendStudentRatings @StudentID = 9;

-- убедимся, что рейтинг рассчитан и дата получения рейтинга из очереди установлена
Select
    RatingTypeID,
    Rating,
    RatingConfirmedForProcessing
from Controls.Ratings AS r
Where r.StudentID = 9

-- проверим статус диалогов (не должно остаться открытых)
SELECT conversation_handle, is_initiator, s.name as 'local service',
far_service, sc.name 'contract', ce.state_desc
FROM sys.conversation_endpoints ce
LEFT JOIN sys.services s
ON ce.service_id = s.service_id
LEFT JOIN sys.service_contracts sc
ON ce.service_contract_id = sc.service_contract_id
ORDER BY conversation_handle;
```