#20 Разработка API для базы данных

Зачем делать эту лабораторную работу?

1. Чтобы научиться комплексно разрабатывать функционал системы на стороне сервера баз данных.
2. Чтобы научиться определять, какие объекты базы данных необходимы для полноценного обеспечения защиты и целостности данных сервера.
3. Чтобы научиться формировать необходимые наборы данных, необходимые для внешней разработки системы.

Что нужно делать?

1. Создать базу данных.

В качестве примера будет рассматривать вариант разработки серверной части модуля системы онлайн-курсов с сертификацией. Модуль должен обеспечивать следующие функции:

1) Внедрение этапов обучения, каждый из которых имеет определенный срок прохождения.
2) Выдача электронных сертификатов о прохождении курса осуществляется в ограниченном количестве для первых успешно сдавших экзамен студентов.
3) Для клиентов должны обеспечиваться регистрация на курс и отслеживание их активности и прогресса.
4) За активное участие в курсах начисляются баллы, которые могут повышать шансы на получение дополнительных бонусов при сертификации.
5) Для клиентов происходят рассылки уведомлений о начале обучения и напоминания о сроках сдачи заданий.
6) Клиенты информируются о том, какой оставшийся объём работы нужно выполнить, чтобы засчиталась активность для возможности получения бонусных возможностей.
7) Информация клиентам осуществляется отправкой сообщений на электронную почту.
8) Клиентам, не подписавшимся на рассылки о процессе обучения, отправляется только информации о курсах.
9) Сообщения представляют с собой оформленные html-страницы с логотипом, заголовком, контентом, графическим изображением активностей и сертификатов.
10) Рассылка сертификатов происходит после сдачи итоговых экзаменов.
11) По итогам курсов происходит выбор лучших обучающихся для награждения дополнительными призами исходя из накопленных бонусов и оценок.
12) Информации об итогах курса отправляется всем клиентам, не проходившим ни одного курса, а персонализированное поздравление тем, кто успешно завершил обучение.

Для такого модуля может использоваться база данных, имеющая следующую структуру:

```
CREATE TABLE Курсы (
    Курс_ID INT PRIMARY KEY,
    Название NVARCHAR(255),
    Описание NVARCHAR(MAX),
    Срок_начала DATE,
    Срок_окончания DATE
);
```

```
CREATE TABLE Этапы_обучения (
    Этап_ID INT PRIMARY KEY,
    Курс_ID INT,
    Название_этапа NVARCHAR(255),
    Срок_начала DATE,
    Срок_окончания DATE,
    FOREIGN KEY (Курс_ID) REFERENCES Курсы(Курс_ID)
);

CREATE TABLE Студенты (
    Студент_ID INT PRIMARY KEY,
    Имя NVARCHAR(255),
    Фамилия NVARCHAR(255),
    Электронная_почта NVARCHAR(255),
    Пароль NVARCHAR(255)
);

CREATE TABLE Прогресс_студента (
    Прогресс_ID INT PRIMARY KEY,
    Студент_ID INT,
    Этап_ID INT,
    Баллы INT,
    Завершен BIT,
    FOREIGN KEY (Студент_ID) REFERENCES Студенты(Студент_ID),
    FOREIGN KEY (Этап_ID) REFERENCES Этапы_обучения(Этап_ID)
);

CREATE TABLE Сертификаты (
    Сертификат_ID INT PRIMARY KEY,
    Студент_ID INT,
    Курс_ID INT,
    Дата_выдачи DATE,
    FOREIGN KEY (Студент_ID) REFERENCES Студенты(Студент_ID),
    FOREIGN KEY (Курс_ID) REFERENCES Курсы(Курс_ID)
);

CREATE TABLE Рассылка (
    Рассылка_ID INT PRIMARY KEY,
    Студент_ID INT,
    Тип NVARCHAR(255),
    Содержание NVARCHAR(MAX),
    Дата_отправки DATE,
    FOREIGN KEY (Студент_ID) REFERENCES Студенты(Студент_ID)
);

CREATE TABLE Награды (
    Награда_ID INT PRIMARY KEY,
    Студент_ID INT,
    Описание NVARCHAR(MAX),
    Дата DATE,
    FOREIGN KEY (Студент_ID) REFERENCES Студенты(Студент_ID)
)
```

Здесь таблицы имеют следующее назначение:

Курсы - необходима для хранения информации о каждом курсе, включая его сроки проведения;

Этапы_обучения - отражает структуру курса, позволяет задать сроки для каждого этапа обучения;

Студенты - содержит информацию о пользователях системы, которые регистрируются на курсы;

Прогресс_студента - отслеживает прогресс каждого студента по этапам обучения и начисленные баллы за активное участие;

Сертификаты - ведение учета выданных сертификатов студентам после успешного прохождения курсов;

Рассылка - фиксирует информацию отправленных студентам уведомлений и рекламных сообщений;

Награды - предназначена для отслеживания дополнительных призов и наград для лучших студентов на основании накопленных баллов и оценок.

2. Разработать объекты базы данных, реализующие функционал модуля.

В качестве примера второй пункт: «Выдача электронных сертификатов о прохождении курса осуществляется в ограниченном количестве для первых успешно сдавших экзамен студентов», может быть реализован хранимой процедурой.

Пример кода хранимой процедуры следующий:

```
CREATE PROCEDURE Выдать_Сертификаты
    @Курс_ID INT,
    @Лимит_сертификатов INT
AS
BEGIN
    -- Выборка успешно сдавших экзамен студентов
    WITH Студенты_сдали_экзамен AS (
        SELECT Прогресс_студента.Студент_ID, MIN(Этапы_обучения.Срок_окончания) as
Дата_Сдачи
        FROM Прогресс_студента
        JOIN Этапы_обучения ON Прогресс_студента.Этап_ID = Этапы_обучения.Этап_ID
        WHERE Этапы_обучения.Название_этапа like '%экзамен%'
          AND Прогресс_Студента.Завершен = 1
          AND Этапы_обучения.Курс_ID = @Курс_ID
        GROUP BY Прогресс_студента.Студент_ID
    )
    INSERT INTO Сертификаты (Студент_ID, Курс_ID, Дата_выдачи)
        -- Выборка первых N сдавших по дате сдачи экзамена
        SELECT TOP (@Лимит_сертификатов) Студент_ID, @Курс_ID, GETDATE()
        FROM Студенты_сдали_экзамен
        ORDER BY Дата_Сдачи;
END
```

В качестве примера четвёртый пункт «За активное участие в курсах начисляются баллы, которые могут повышать шансы на получение дополнительных бонусов при сертификации», может быть реализован триггером.

Пример кода триггера следующий:

```
CREATE TRIGGER Обновление_активности
ON Прогресс_студента
AFTER UPDATE
AS
BEGIN
    -- Проверка, что столбец Завершен был обновлен с 0 на 1
    IF (UPDATE(Завершен))
    BEGIN
        -- Объявление переменных
        DECLARE @StudentID INT, @StageID INT, @PointsToAdd INT

        SET @PointsToAdd = 10; -- количество баллов за завершенный этап (например, 10)

        -- Цикл для всех обновлённых строк
        DECLARE updatedCursor CURSOR FOR
        SELECT Студент_ID, Этап_ID
```

```
        FROM INSERTED
        WHERE Завершен = 1
              OPEN updatedCursor
        FETCH NEXT FROM updatedCursor INTO @StudentID, @StageID

        WHILE @@FETCH_STATUS = 0
        BEGIN
            -- Начисление баллов
            UPDATE Прогресс_студента
            SET Баллы = Баллы + @PointsToAdd
            WHERE Студент_ID = @StudentID AND Этап_ID = @StageID

            FETCH NEXT FROM updatedCursor INTO @StudentID, @StageID
        END;

        -- Закрытие курсора и освобождение памяти
        CLOSE updatedCursor
        DEALLOCATE updatedCursor
    END
END
```

3. Сформировать сообщения для рассылок.

Для формирования сообщений могут быть реализованы хранимые процедуры, формирующий HTML-структуру электронных писем. Пример процедуры, формирующей сертификат для дальнейшей рассылки, следующий:

```
CREATE PROCEDURE Формирование_сертификата
    @Рассылка_ID INT,
    @Студент_ID INT,
    @Название_Курса NVARCHAR(255),
    @ФИО_Студента NVARCHAR(255),
    @Дата_Завершения NVARCHAR(10),
    @Логотип_URL NVARCHAR(MAX),
    @Изображение_Сертификата_URL NVARCHAR(MAX)
AS
BEGIN
    -- Определение HTML-структуры
    DECLARE @Содержание NVARCHAR(MAX);
    SET @Содержание = N'
    <html>
    <head>
        <style type="text/css">
            body { font-family: Arial, sans-serif; }
            .container { width: 600px; margin: 0 auto; padding: 20px; border: 1px solid
#ccc; }
            .logo { width: 100px; }
            .header { font-size: 24px; margin-top: 20px; }
            .certificate-img { margin-top: 20px; }
        </style>
    </head>
    <body>
        <div class="container">
            <img src="' + @Логотип_URL + '" alt="Logo" class="logo">
            <div class="header">Сертификат о завершении курса</div>
            <p>Поздравляем, ' + @ФИО_Студента + ', с успешным завершением курса:</p>
            <h2>' + @Название_Курса + '</h2>
            <div class="certificate-img">
                <img src="' + @Изображение_Сертификата_URL + '" alt="Certificate">
            </div>
            <p>Дата завершения: ' + @Дата_Завершения + '</p>
        </div>
    </body>
    </html>';
```

```
    -- Вставка HTML-сообщения в таблицу Рассылка
    INSERT INTO Рассылка (Рассылка_ID, Студент_ID, Тип, Содержание, Дата_отправки)
    VALUES (@Рассылка_ID, @Студент_ID, 'Сертификат', @Содержание, GETDATE());
END
```

Как узнать, что все выполнено?

Проверьте пункты в этом чек-листе:

☐ Создана база данных со всеми таблицами

☐ В базе данных установлены базовые ограничения целостности

☐ Разработаны объекты базы данных, формирующие низкоуровневое API согласно функционалу модуля

☐ При разработке соблюдены все пользовательские ограничения целостности по каждому пункту задания

☐ Разработаны процедуры, формирующие сообщения для рассылки

☐ Все объекты протестированы и работают корректно

Варианты заданий:

1 вариант

Разработать серверную часть модуля розыгрыша призов для клиентов. Модуль должен обеспечивать следующие функции:

1. Акция розыгрыша проходит в несколько этапов, ограниченных по времени;
2. На каждом этапе акции в качестве призов разыгрываются товары из ассортимента в заранее определенном количестве;
3. Для покупателей должна быть возможность регистрации в акции и отправки уникальных кодов;
4. Каждый код относится к покупке определенной стоимости и пропорционально повышает вероятность выигрыша более ценного приза;
5. В ходе каждого этапа акции осуществляется информирование клиентов о начале акции и за несколько дней до её завершения;
6. В случае регистрации кодов на сумму, приближающейся к сумме, повышающей вероятность выигрыша более ценного приза, также осуществляется информирование;
7. Информация клиентам отправляется в виде сообщений на электронную почту;
8. Клиентам, не подписавшимся на рекламные акции, отправляются только сообщения о розыгрыше;
9. Сообщения представляют с собой оформленные html-страницы с логотипом, заголовком, контентом, фотографиями призов;
10. Розыгрыш проходит через несколько дней после завершения каждого этапа акции;
11. Розыгрыш призов происходит случайным образом с учётом вероятностей получения более ценных призов клиентами;
12. Информация об итогах розыгрыша отправляется всем клиентам без персональной информации и выигравшим клиентам с указанием приза.

2 вариант:

Разработать серверную часть модуля программы лояльности для клиентов авиакомпании. Модуль должен обеспечивать следующие функции:

1. Мили накапливаются клиентами в течении определенных периодов и списываются автоматически, либо самими пользователями;
2. Для начисления миль существует несколько видов карт разных уровней;
3. Для клиентов должна обеспечиваться возможность регистрации в системе и занесения данных о совершенных полётах;
4. За каждый перелёт начисляется определенное количество миль, пропорциональное дальности перелёта;
5. Каждые два месяца для клиентов проводятся акции, в которых при совершении полётов, количество миль удваивается;
6. Клиенты информируются о каждом начислении миль, а также о проводимых акциях;
7. В случае накопления количества миль, приближающееся к необходимому количеству для повышения уровня карты, также осуществляется информирование;
8. Информация клиентам отправляется в виде сообщений на электронную почту;
9. Клиентам, не подписавшимся на рекламные акции, отправляются только сообщения о начислении миль;
10. Сообщения представляют с собой оформленные html-страницы с логотипом, заголовком, контентом, графическим изображением карт и накопленных миль;
11. За месяц до конца отчётного периода клиенты информируются о необходимости потратить мили;
12. Через два дня после окончания каждой акции клиенты информируются о накопленных милях за время акции.