

## Секция 2

# ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ И МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ В БИЗНЕСЕ

**A. I. Andreichykova, A. S. Trushkina**

*School of Business of BSU, Minsk, Belarus*

*Scientific supervisor – E.G. Grinevich*

## DEVELOPMENT OF AN INFORMATION SYSTEM FOR MANAGEMENT OF UNIVERSITY EVENTS

*This scientific work introduces the design and implementation of an information system specifically built to optimize events management in a university context. It outlines the development process, highlighting the use of advanced technologies and tools to create a resilient system that can store, process, and analyze extensive event data. Ultimately, this information system serves as a valuable asset to enhance the efficiency of university event management, simplifying procedures and improving coordination and execution of various events and activities at the Belarusian State University, Nezavisimosti Av., 4.*

**Keywords:** database design, database management system, MS SQL Server DBMS, Transact-SQL, functions creation, procedures creation

Events take an important place in the university environment, covering a wide range of events from academic conferences to cultural exhibitions and social gatherings. In order to effectively manage such events, it is necessary to use a modern technological solution that will optimize administrative processes. The creation of a specialised information system is becoming a key element for solving a multitude of tasks related to the organisation of events at universities.

Today, registration for events often takes place via Telegram messenger, which is not very convenient and creates many problems. The main difficulty is that it is impossible to accurately control the number of available seats. Students submit requests in different formats: some write personal messages; others leave requests in a common group. This chaotic approach complicates the work of organizers who need to keep track of the number of participants and allocate resources.

The proposed information system is intended to be a single tool for managing all events. It will serve as a centralised platform that aggregates event information and makes it available to students, faculty and staff, providing convenience regardless of their department or location.

After analyzing the activities of the institution, the seven main subjects were identified: Registration, Feedback, Student, Event, Location, Category, and Organizer.

Using the online tool draw.io, the logical schema of the database was created (Fig. 1).

Because SQL is reliable and has little chance of data loss, it was utilized to design and maintain the database [1].

The event management system includes several key queries that streamline student participation and assist organizers in efficiently managing events. One query retrieves a student's complete registration history, showing details like event names, dates, and statuses, allowing users to track their involvement. Another query lists all students registered for a specific event, aiding organizers in managing participant lists and attendance.

A query for upcoming events allows students to check their future registrations, ensuring they stay informed about upcoming activities. Additionally, a feedback query calculates the average

event rating, offering organizers valuable insights for improving future events [3]. Finally, a query updates registration statuses for past events, automatically marking them as "Cancelled" or "Completed," helping to maintain accurate records and improve administrative clarity. These queries work together to enhance both user experience and administrative efficiency.

The query for retrieving a student's complete registration history is shown in Fig. 2, and the result is shown in Fig. 3.

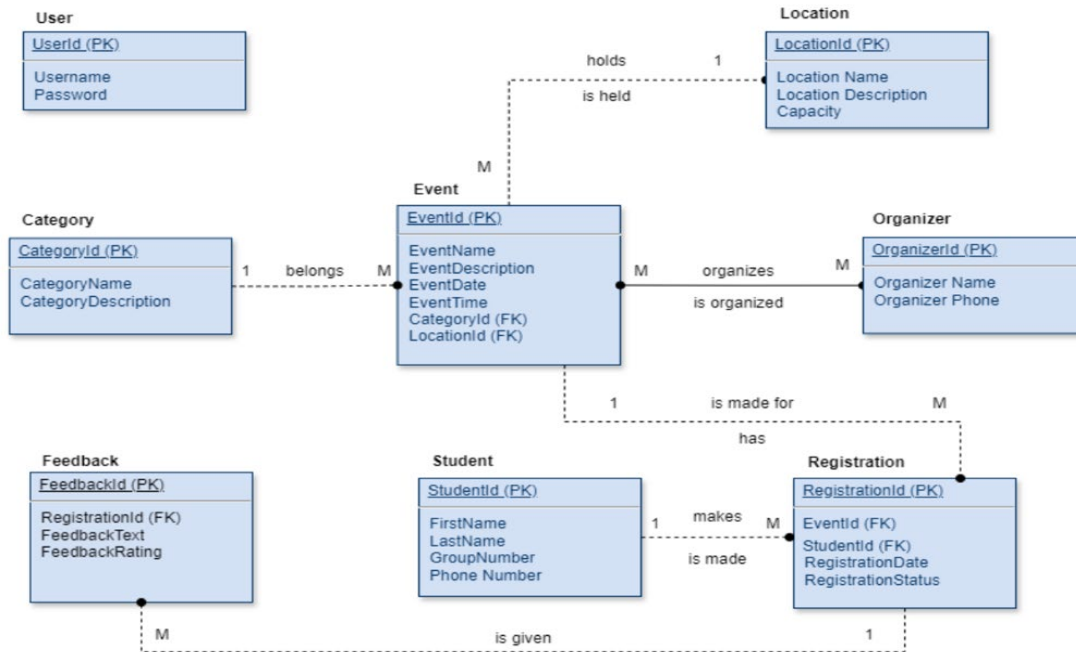


Fig. 1. Logical schema in draw.io

```

CREATE FUNCTION GetStudentsForEvent(@EventId INT)
RETURNS @StudentTable TABLE(
    UserId INT,
    FirstName VARCHAR (50),
    LastName VARCHAR(50),
    RegistrationDate DATE
)
AS
BEGIN
    INSERT INTO @StudentTable (UserId, FirstName, LastName, RegistrationDate)
    SELECT s. UserId, s.FirstName, s.LastName, r.RegistrationDate
    FROM Student s
    JOIN Registration r ON s. UserId = r.StudentId
    WHERE r.EventId = @EventId;
    RETURN;
END;
SELECT * FROM dbo.GetStudentsForEvent(10);
  
```

Fig. 2. Resulting of creating query

	UserId	FirstName	LastName	RegistrationDate
1	100	John	Doe	2024-04-18
2	104	Anita	Andreichykova	2024-05-07

Fig. 3. Resulting of implementing query

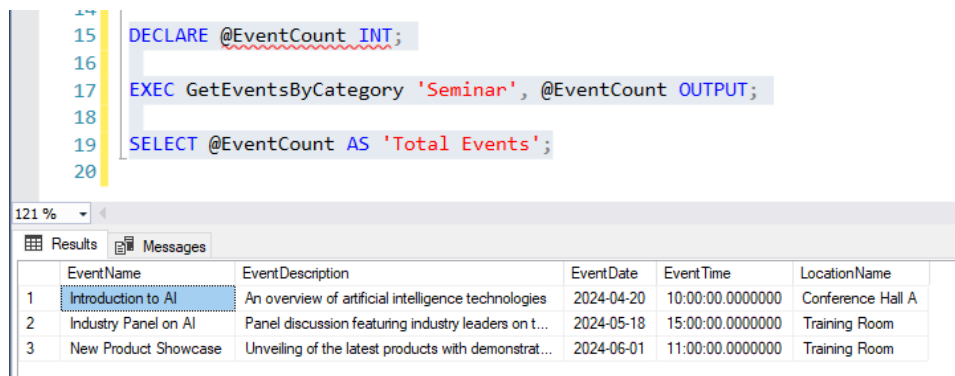
On the administration's side, the significant aspect of the project is its use of automation for repetitive tasks. For instance, specific queries ensure that expired registrations are automatically marked, and procedures are in place to register students for events and manage overbooked situations. Feedback collection is another crucial feature, with the system designed to automatically insert pending feedback requests for events that have passed. This makes it possible to streamline operations, reduce manual input, and enhance data processing.

Advanced database functionalities like triggers and stored procedures play a pivotal role [2]. They automate event-related tasks such as calculating average feedback ratings, managing student information, and handling registration status updates. Additionally, user-defined functions allow administrators to retrieve critical data easily, such as the number of events in a particular category or details about students who haven't registered for any event.

One of the examples of developed advanced functionalities involves a procedure to Retrieve Events by Category: this procedure takes a category name as input and returns information about all events in that category. The code is shown in Fig. 4, and the result is shown in Fig. 5.

```
CREATE PROCEDURE GetEventsByCategory @CategoryName VARCHAR(100), @EventCount INT OUTPUT
BEGIN
SELECT
    e.EventName,
    e.EventDescription,
    e.EventDate,
    e.EventTime,
    l.LocationName
FROM Event e
INNER JOIN Location l
    ON e.LocationId = l.LocationId
    INNER JOIN Category c ON e.CategoryId = c.CategoryId WHERE
    c.CategoryName = @CategoryName;
SET @EventCount = @@ROWCOUNT; END;
DECLARE @EventCount INT;
EXEC GetEventsByCategory 'Seminar', @EventCount OUTPUT; SELECT @EventCount AS 'Total Events';
```

Fig. 4. Procedure to Retrieve Events by Category



```
15 DECLARE @EventCount INT;
16
17 EXEC GetEventsByCategory 'Seminar', @EventCount OUTPUT;
18
19 SELECT @EventCount AS 'Total Events';
20
```

	EventName	EventDescription	EventDate	EventTime	LocationName
1	Introduction to AI	An overview of artificial intelligence technologies	2024-04-20	10:00:00.0000000	Conference Hall A
2	Industry Panel on AI	Panel discussion featuring industry leaders on t...	2024-05-18	15:00:00.0000000	Training Room
3	New Product Showcase	Unveiling of the latest products with demonstrat...	2024-06-01	11:00:00.0000000	Training Room

Fig 5. Resulting of implementing the GetEventsByCategory procedure

The event management system also employs triggers to automate key processes, improving the efficiency of administrative tasks. As displayed in Fig. 6, a trigger using a cursor is used to notify administrators whenever a student's details are updated [2]. This trigger automatically prints a notification for each update, ensuring that administrators are kept informed about changes without manual intervention, as shown in Fig. 7. Such automation ensures real-time monitoring and enhances the overall data management process in the system.

To bring project to life, C# was for building the backbone of the information system. With its versatility and efficiency, C# is the perfect fit for creating robust and scalable applications. The C# form to check all the registrations of a student is presented in Fig. 8.

A dedicated backup plan is implemented to safeguard the database, ensuring that all event-related data is secure and can be restored in the event of failure or data loss [4]. By regularly creating these backups, the system provides a safety net, allowing for quick recovery and minimizing downtime in case of unexpected issues. This focus on data protection ensures that the system remains reliable and resilient, even in the face of potential disruptions.

```
CREATE TRIGGER NotifyAdminOnStudentUpdate
ON Student
AFTER UPDATE
AS
BEGIN
DECLARE @UserId INT;
DECLARE @FirstName VARCHAR(50);
DECLARE @LastName VARCHAR(50);
DECLARE StudentCursor CURSOR FOR
SELECT UserId, FirstName, LastName
FROM INSERTED; -- Records that were updated
OPEN StudentCursor;
FETCH NEXT FROM StudentCursor INTO @UserId, @FirstName, @LastName;
WHILE @@FETCH_STATUS = 0
BEGIN
PRINT 'Student updated: ' + @FirstName + ' ' + @LastName;
FETCH NEXT FROM StudentCursor INTO @UserId, @FirstName, @LastName;
END;
CLOSE StudentCursor;
DEALLOCATE StudentCursor;
END;
UPDATE Student
SET PhoneNumber = '80257382948'
WHERE UserId = 108;
```

Fig. 6. Trigger to Notify Admins of Student Changes

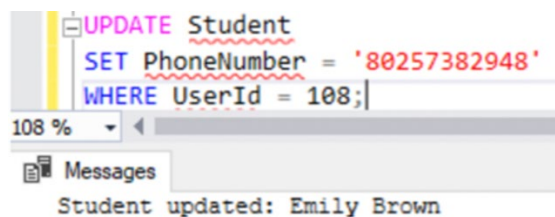


Fig. 7. Resulting of implementing the NotifyAdminOnStudentUpdate trigger

Registration of Student   Register for Event   Feedback   Profile						
1	Для 9					
StudentId	FirstName	LastName	GroupNumber	PhoneNumber	Email	DateOfBirth
100	John	Doe	101	80293648374	John.Doe@bsu.by	15.01.2004
102	Alona	Trushkina	153	80442847464	Alona.Trushkina@bsu.by	10.12.2003
104	Anita	Andreichykova	153	80296298489	Anita.Andreichykova@sbmt...	03.10.2003
106	Bob	Williams	154	80294657572	Bob.Williams@bsu.by	01.11.2003
108	Emily	Brown	253	80257382948	Emily.Brown@bsu.by	02.10.2003
110	Michael	Davis	244	80297464692	Michael.Davis@bsu.by	10.12.2003
112	Sarah	Martinez	151	80296483737	Sarah.Martinez@bsu.by	17.09.2003
RegistrationId	EventId	StudentId	RegistrationDate	RegistrationStatus		
1	10	100	18.04.2024	Confirmed		

Fig. 8. Form to check all the registration of one student

The university event management system uses MS SQL Server and C# to streamline processes, lighten the administrative load, and optimize resource use [1]. The implementation of this system not only automates routine tasks, but also improves students' interaction with the university, which contributes to their increased involvement in academic and extracurricular activities.

The social significance of the project lies in the fact that the system contributes to strengthening the student community, improving communication and optimising educational and cultural processes at the university. Opportunities for further development include the integration of mobile applications, the addition of a real-time notification system, and the use of artificial intelligence technologies for personalised event recommendations.

### References

1. *Ben-Gan, I.* T-SQL Fundamentals / I. Ben-Gan. – Pearson Education, Inc., 2023 – 139 p.
2. *Coronel, C.* Database Systems Design, Implementation & Management / C. Coronel, S. Morris. – Cengage Learning, 2023. – 223 p.
3. *Dewson, R.* Beginning SQL Server for Developers (The Expert's Voice in SQL Server) / Dewson, R. – Apress, 2014. – 56 p.
4. *Митин, А. И.* Работа с базами данных Microsoft SQL Server : сценарии практических занятий / А. И. Митин. – Москва, 2020. – 127 p.