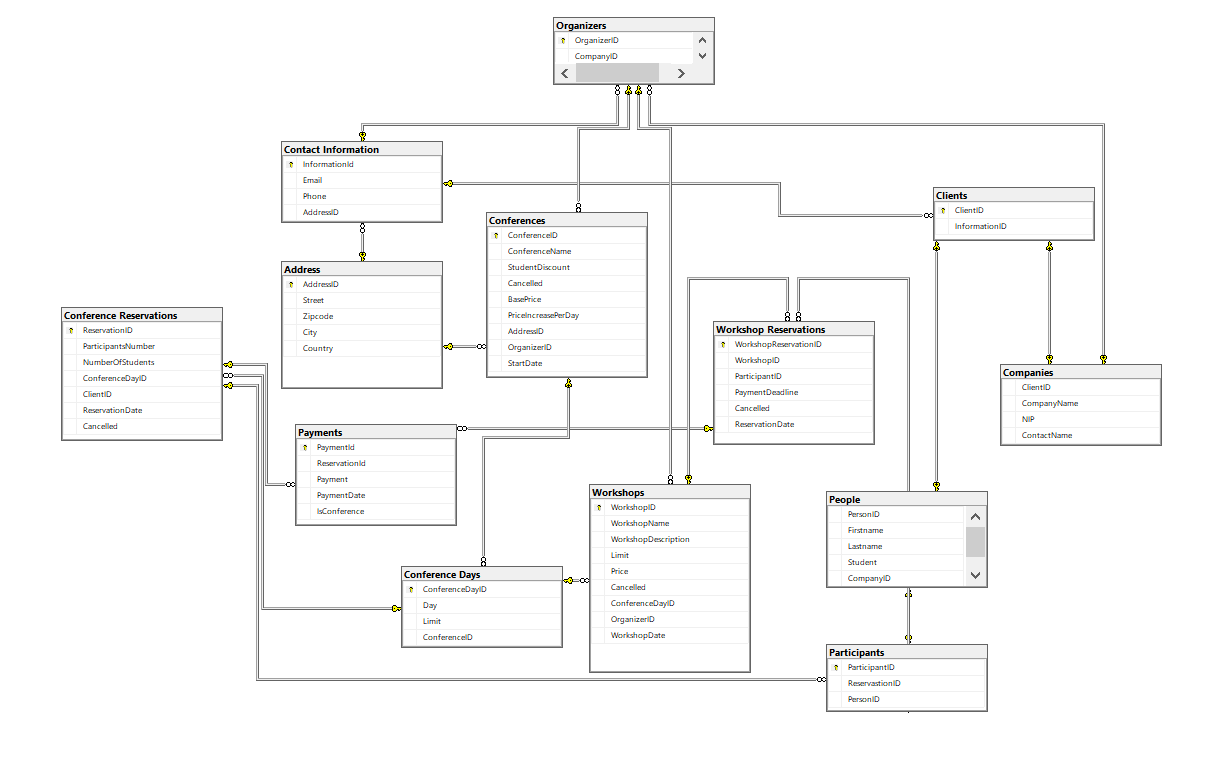
Mikołaj Zatorski

Dominik Jastrząb

1. **Schemat bazy danych**



1. **Opisy tabel i kod tworzący je:**
2. **Workshop Reservations**

Zawiera rezerwacje na warsztaty dokonywane przez uczestników konferencji.

1. **Conference Reservations**

Zawiera rezerwacje miejsc na dany dzień konferencji, jeśli rezerwacja jest zbiorowa to ParticipantsNumber zawiera liczbę uczestników.

1. **Participants**

Tabela wiążąca fizyczne osoby z ich rezerwacjami, każda osoba w tej tabeli musi mieć rezerwacje na dzień konferencji.

1. **Conferences**

Tabela zawierająca szczegóły konferencji.

1. **Conference Days**

Tabela zawierająca poszczególne dni konferencji wraz z limitami osób. Klienci dokonują rezerwacji na dany dzień konferencji .

1. **Workshops**

Tabela zawierająca szczegóły warsztatów.

1. **Organizers**

Tabela ze opisująca szczegóły dotyczące organizatorów konferencji/warsztatów.

1. **Payments**

Tabela zawierająca wpłaty dotyczące danych rezerwacji na konferencję/warsztat.

1. **People**

Tabela zawierająca dane poszczególnych osób fizycznych. Jeśli StudentCardID jest NULL to osoba nie jest studentem, jeśli CompanyID jest NULL to osoba nie jest zatrudniona w żadnej firmie.

1. **Clients**

Tabela z klientami, czyli osobami lub firmami, które dokonują rezerwacji na konferencję, opłacają ją, zostawiają swoje dane kontaktowe.

1. **Companies**

Tabela przechowująca dane firm dokonujących rezerwacji na konferencje.

1. **Contact Information**

Dane kontaktowe klientów oraz organizatorów. W niektórych przypadkach nie muszą zawierać adresu (klient może nie życzyć sobie udostępniać takich danych, a komunikacja z firmą organizującą konferencję odbywa się poprzez telefon/e-mail).

1. **Address**

Przechowuje zarówno adresy klientów jak i miejsca odbywania się konferencji.

1. **Widoki**

**Cancelled Conference Reservations**

Spis anulowanych rezerwacji na konferencje.

CREATE VIEW [dbo].[Cancelled Conference Reservations]

AS

SELECT ReservationID, ParticipantsNumber, NumberOfStudents,

ClientID, ReservationDate, Cancelled, ConferenceDayID

FROM dbo.[Conference Reservations]

WHERE (Cancelled = 1)

GO

**Cancelled Conferences**

Spis anulowanych konferencji.

CREATE VIEW [dbo].[Cancelled Conferences]

AS

SELECT ConferenceID, ConferenceName, StudentDiscount, Cancelled,

BasePrice, PriceIncreasePerDay, AddressID, OrganizerID, StartDate

FROM dbo.Conferences

WHERE (Cancelled = 1)

GO

**Cancelled Workshop Reservations**

Spis anulowanych rezerwacji na warsztaty.

CREATE VIEW [dbo].[Cancelled Workshop Reservations]

AS

SELECT WorkshopReservationID, WorkshopID, ParticipantID, PaymentDeadline, Cancelled, ReservationDate

FROM dbo.[Workshop Reservations]

WHERE (Cancelled = 1)

GO

**Cancelled Workshops**

Spis anulowanych warsztatów.

CREATE VIEW [dbo].[Cancelled Workshops]

AS

SELECT WorkshopID, WorkshopName, WorkshopDescription,

Cancelled, Limit, Price, ConferenceDayID, OrganizerID, WorkshopDate

FROM dbo.Workshops

WHERE (Cancelled = 1)

GO

**Conf Reservations Payments Balance**

Spis rezerwacji na dni konferencji wraz ze statusami płatności (czy zapłacono już pełną kwotę, termin zapłaty itp.).

CREATE VIEW [dbo].[Conf Reservations Payments Balance]

AS

SELECT dbo.[Conference Reservations].ReservationID,

dbo.[Conference Reservations].ClientID,

dbo.[Conference Reservations].Cancelled,

dbo.CalculateFinalPrice(dbo.[Conference Reservations].ReservationID) AS FinalPrice,

SUM(dbo.Payments.Payment) AS PaymentsSum,

dbo.CalculateFinalPrice(dbo.[Conference Reservations].ReservationID) - SUM(dbo.Payments.Payment)

AS ToPay,

DATEADD(week, 1, dbo.[Conference Reservations].ReservationDate) AS PaymentDeadline

FROM dbo.[Conference Reservations]

INNER JOIN dbo.Payments

ON dbo.[Conference Reservations].ReservationID = dbo.Payments.ReservationId

WHERE (dbo.Payments.IsConference = 1)

GROUP BY dbo.[Conference Reservations].ReservationID,

dbo.[Conference Reservations].ClientID,

dbo.CalculateFinalPrice(dbo.[Conference Reservations].ReservationID),

DATEADD(week, 1, dbo.[Conference Reservations].ReservationDate),

dbo.[Conference Reservations].Cancelled

GO

**Conferences available spaces**

Spis dni konferencji wraz z liczbą wolnych miejsc na nie.

CREATE VIEW [dbo].[Conferences available spaces]

AS

SELECT dbo.[Conference Days].ConferenceDayID,

dbo.[Conference Days].Limit,

COUNT(dbo.Participants.ParticipantID) AS Reserved,

dbo.[Conference Days].Limit - COUNT(dbo.Participants.ParticipantID) AS Free

FROM dbo.[Conference Days]

INNER JOIN dbo.[Conference Reservations]

ON dbo.[Conference Days].ConferenceDayID = dbo.[Conference Reservations].ConferenceDayID

INNER JOIN dbo.Participants

ON dbo.[Conference Reservations].ReservationID = dbo.Participants.ReservationID

WHERE (dbo.[Conference Reservations].Cancelled = 0)

GROUP BY dbo.[Conference Days].ConferenceDayID, dbo.[Conference Days].Limit

GO

**Incoming Conferences**

Nadchodzące konferencje.

CREATE VIEW [dbo].[Incoming Conferences]

AS

SELECT dbo.Conferences.ConferenceName, dbo.Conferences.BasePrice,

dbo.Conferences.Cancelled, dbo.Conferences.StudentDiscount,

dbo.Conferences.PriceIncreasePerDay, dbo.Conferences.StartDate,

dbo.[Conference Days].Day, dbo.[Conference Days].Limit,

dbo.Companies.CompanyName AS OrganizerName

FROM dbo.Conferences

INNER JOIN dbo.Organizers

ON dbo.Conferences.OrganizerID = dbo.Organizers.OrganizerID

INNER JOIN dbo.[Conference Days]

ON dbo.Conferences.ConferenceID = dbo.[Conference Days].ConferenceID

INNER JOIN dbo.Companies

ON dbo.Organizers.CompanyID = dbo.Companies.ClientID

WHERE (dbo.Conferences.StartDate > DATEADD(week, 2, GETDATE()))

GO

**Missing Client Data**

Spis rezerwacji na konferencję wraz z informacją czy podano dane wszystkich uczestników.

CREATE VIEW [dbo].[Missing Client Data]

AS

SELECT dbo.Conferences.ConferenceName,

dbo.[Conference Reservations].ParticipantsNumber,

COUNT(dbo.Participants.ParticipantID) AS SignedParticipants,

dbo.[Conference Reservations].ParticipantsNumber - COUNT(dbo.Participants.ParticipantID) AS MissingParticipants

FROM dbo.[Conference Reservations]

INNER JOIN dbo.[Conference Days]

ON dbo.[Conference Reservations].ConferenceDayID = dbo.[Conference Days].ConferenceDayID

INNER JOIN dbo.Conferences

ON dbo.[Conference Days].ConferenceID = dbo.Conferences.ConferenceID

INNER JOIN dbo.Clients

ON dbo.[Conference Reservations].ClientID = dbo.Clients.ClientID

INNER JOIN dbo.[Contact Information]

ON dbo.Clients.InformationID = dbo.[Contact Information].InformationId

INNER JOIN dbo.Participants

ON dbo.[Conference Reservations].ReservationID = dbo.Participants.ReservationID

GROUP BY dbo.Conferences.ConferenceName, dbo.[Conference Reservations].ParticipantsNumber

HAVING (COUNT(dbo.Participants.ParticipantID) < dbo.[Conference Reservations].ParticipantsNumber)

GO

**Workshop Reservations Payments Balance**

Spis rezerwacji na warsztaty wraz ze statusami płatności (czy zapłacono już pełną kwotę itp.).

CREATE VIEW [dbo].[Workshop Reservations Payments Balance]

AS

SELECT dbo.[Workshop Reservations].WorkshopReservationID,

dbo.[Workshop Reservations].ParticipantID,

dbo.[Workshop Reservations].Cancelled,

dbo.Workshops.Price, SUM(dbo.Payments.Payment) AS PaymentsSum,

dbo.Workshops.Price - SUM(dbo.Payments.Payment) AS ToPay

FROM dbo.[Workshop Reservations]

INNER JOIN dbo.Workshops

ON dbo.[Workshop Reservations].WorkshopID = dbo.Workshops.WorkshopID

INNER JOIN dbo.Payments

ON dbo.[Workshop Reservations].WorkshopReservationID = dbo.Payments.ReservationId

GROUP BY dbo.[Workshop Reservations].WorkshopReservationID,

dbo.[Workshop Reservations].ParticipantID,

dbo.Workshops.Price,

dbo.[Workshop Reservations].Cancelled

GO

**Workshops available spaces**

Spis warsztatów wraz z liczbą wolnych miejsc na nie.

CREATE VIEW [dbo].[Workshops Available Spaces]

AS

SELECT dbo.Workshops.WorkshopID,

dbo.Workshops.Limit,

COUNT(dbo.Participants.ParticipantID) AS Reserved,

dbo.Workshops.Limit - COUNT(dbo.[Workshop Reservations].ParticipantID) AS Free

FROM dbo.Workshops

INNER JOIN dbo.[Workshop Reservations]

ON dbo.Workshops.WorkshopID = dbo.[Workshop Reservations].WorkshopID

INNER JOIN dbo.Participants

ON dbo.[Workshop Reservations].ParticipantID = dbo.Participants.ParticipantID

WHERE (dbo.[Workshop Reservations].Cancelled = 0)

GROUP BY dbo.Workshops.WorkshopID, dbo.Workshops.Limit

GO

* 1. **Procedury z parametrem, spełniające funkcję widoków**

**ConferenceDayParticipantsList**

Spis uczestników danego dnia konferencji wraz z danymi do identyfikatora.

CREATE PROCEDURE [dbo].[ConferenceDayParticipantsList]

@ConferenceDayID int = 0

AS

BEGIN

SET NOCOUNT ON;

SELECT P.PersonID, P.Firstname, P.Lastname, C.CompanyName

FROM dbo.[Conference Days] AS CD

INNER JOIN dbo.[Conference Reservations] AS CR

ON CD.ConferenceDayID = @ConferenceDayID AND CD.ConferenceDayID = CR.ConferenceDayID

INNER JOIN dbo.Participants AS PART

ON CR.Cancelled = 0 AND CR.ReservationID = PART.ReservationID

INNER JOIN PEOPLE AS P

ON PART.PersonID = P.PersonID

LEFT OUTER JOIN Companies AS C

ON P.CompanyID = C.ClientID

END

**ConfReservationDetails**

Szczegółowe dane pojedynczej rezerwacji.

ALTER PROCEDURE [dbo].[ConfReservationDetails]

@ReservationID int = 0

AS

BEGIN

SET NOCOUNT ON;

SELECT CR.ReservationID, CR.ReservationDate, CR.Cancelled,

CR.ParticipantsNumber, CR.NumberOfStudents, CR\_BALANCE.FinalPrice,

CR\_BALANCE.ToPay, CR\_BALANCE.PaymentDeadline, CR.ClientID, CD.Day,

CD.Limit, C.ConferenceID, C.ConferenceName, C.Cancelled

FROM [Conference Reservations] AS CR

INNER JOIN [Conference Days] AS CD

ON CR.ReservationID = @ReservationID AND CR.ConferenceDayID = CD.ConferenceDayID

INNER JOIN Conferences AS C

ON CD.ConferenceID = C.ConferenceID

INNER JOIN [Conf Reservations Payments Balance] AS CR\_BALANCE

ON CR.ReservationID = CR\_BALANCE.ReservationID

END

**MyConferences**

Spis danych o konferencjach, dniach, na które jest zapisana dana osoba.

ALTER PROCEDURE [dbo].[MyConferences]

@PersonID int = 0

AS

BEGIN

SET NOCOUNT ON;

SELECT C.ConferenceID, C.ConferenceName, C.ConferenceDescription, C.StartDate,

A.Street, A.Zipcode, A.City, COMP.CompanyName, C.Cancelled, CD.ConferenceDayID, CD.Day, CD.Limit

FROM People AS P

INNER JOIN Participants AS PART

ON P.PersonID = @PersonID and P.PersonID = PART.PersonID

INNER JOIN [Conference Reservations] AS CR

ON PART.ReservationID = CR.ReservationID AND CR.Cancelled = 0

INNER JOIN [Conference Days] AS CD

ON CR.ConferenceDayID = CD.ConferenceDayID

INNER JOIN Conferences AS C

ON CD.ConferenceID = C.ConferenceID

INNER JOIN Address as A

ON C.AddressID = A.AddressID

INNER JOIN Organizers AS O

ON C.OrganizerID = O.OrganizerID

INNER JOIN Companies AS COMP

ON O.CompanyID = COMP.ClientID

END

**MyConfReservationsBalance**

Spis rezerwacji na konferencje danego użytkownika wraz z informacją o płatnościach.

ALTER PROCEDURE [dbo].[MyConfReservationsBalance]

@ClientID int = 0

AS

BEGIN

SET NOCOUNT ON;

SELECT R.ReservationID, R.Cancelled, R.FinalPrice, R.PaymentsSum, R.ToPay, R.PaymentDeadline

FROM [Conf Reservations Payments Balance] AS R

WHERE R.ClientID = @ClientID

END

**MyWorkshopReservationBalance**

Spis rezerwacji na warsztaty danego użytkownika wraz z informacją o płatnościach.

ALTER PROCEDURE [dbo].[MyWorkshopReservationsBalance]

@PersonID int = 0

AS

BEGIN

SET NOCOUNT ON;

SELECT WR.WorkshopReservationID, WR.Cancelled, WR.Price, WR.PaymentsSum, WR.ToPay

FROM People AS P

INNER JOIN Participants AS PART

ON P.PersonID = @PersonID AND P.PersonID = PART.PersonID

INNER JOIN [Workshop Reservations Payments Balance] AS WR

ON PART.ParticipantID = WR.ParticipantID

END

**My Workshops**

Spis danych o warsztatach, na które jest zapisana dana osoba.

ALTER PROCEDURE [dbo].[MyWorkshops]

@PersonID int = 0

AS

BEGIN

SET NOCOUNT ON;

SELECT W.WorkshopID, W.WorkshopName, W.WorkshopDescription, W.WorkshopDate,

COMP.CompanyName as 'Organizer', W.Cancelled, W.Limit, W.Price

FROM People AS P

INNER JOIN Participants AS PART

ON P.PersonID = @PersonID and P.PersonID = PART.PersonID

INNER JOIN [Workshop Reservations] AS WR

ON PART.ParticipantID = WR.ParticipantID AND WR.Cancelled = 0

INNER JOIN Workshops AS W

ON WR.WorkshopID= W.WorkshopID

INNER JOIN Organizers AS O

ON W.OrganizerID = O.OrganizerID

INNER JOIN Companies AS COMP

ON O.CompanyID = COMP.ClientID

END

**WorkshopParticipantsList**

Spis uczestników danego warsztatu wraz z danymi do identyfikatora.

ALTER PROCEDURE [dbo].[WorkshopParticipantsList]

@WorkshopID int = 0

AS

BEGIN

SET NOCOUNT ON;

SELECT P.PersonID, P.Firstname, P.Lastname, C.CompanyName

FROM dbo.Workshops AS W

INNER JOIN dbo.[Workshop Reservations] WR

ON W.WorkshopID = @WorkshopID AND W.WorkshopID = WR.WorkshopID

INNER JOIN dbo.Participants AS PART

ON WR.Cancelled = 0 and WR.ParticipantID = PART.ParticipantID

INNER JOIN PEOPLE AS P

ON PART.PersonID = P.PersonID

LEFT OUTER JOIN Companies AS C

ON P.CompanyID = C.ClientID

END

**WorkshopsReservationDetails**

Szczegółowe informacje na temat danej rezerwacji na warsztat.

ALTER PROCEDURE [dbo].[WorkshopReservationDetails]

@ReservationID int = 0

AS

BEGIN

SET NOCOUNT ON;

SELECT WR.WorkshopReservationID, WR.ReservationDate, WR.Cancelled, W.Price,

WR.PaymentDeadline, WR.ParticipantID, W.WorkshopID, W.WorkshopName,

W.Cancelled, W.Limit

FROM [Workshop Reservations] AS WR

INNER JOIN Workshops AS W

ON WR.WorkshopReservationID = @ReservationID AND WR.WorkshopID = W.WorkshopID

END

**TopCustomers**

Dla danego organizatora zwraca listę klientów najczęściej korzystających z jego usług.

ALTER PROCEDURE [dbo].[TopCustomers]

@OrganizerID int = 0

AS

BEGIN

SET NOCOUNT ON;

SELECT CL.ClientID, COUNT(CR.ReservationID) as 'ReservationsNumber'

FROM Organizers AS O

INNER JOIN Conferences AS C

ON O.OrganizerID = @OrganizerID AND O.OrganizerID = C.OrganizerID

INNER JOIN [Conference Days] AS CD

ON C.ConferenceID = CD.ConferenceID

INNER JOIN [Conference Reservations] AS CR

ON CR.Cancelled = 0 AND CD.ConferenceDayID = CR.ConferenceDayID

INNER JOIN Clients AS CL

ON CR.ClientID = CL.ClientID

GROUP BY CL.ClientID

ORDER BY [ReservationsNumber] DESC

END

**ClientDetails**

Szczegółowe dane danego klienta.

ALTER PROCEDURE [dbo].[ClientDetails]

@ClientID int = 0

AS

BEGIN

SET NOCOUNT ON;

SELECT C.ClientID, P.Firstname, P.Lastname, P.Student,

COMP.CompanyName, COMP.ContactName, COMP.NIP, I.Email,

I.Phone, A.Country, A.City, A.Street, A.Zipcode

FROM Clients AS C

INNER JOIN [Contact Information] AS I

ON C.ClientID = @ClientID AND C.InformationID = I.InformationId

LEFT OUTER JOIN Address AS A

ON I.AddressID = A.AddressID

LEFT OUTER JOIN Companies AS COMP

ON C.ClientID = COMP.ClientID

LEFT OUTER JOIN People AS P

ON C.ClientID = P.PersonID

END

1. **Funkcje ze skalarną wartością zwracaną**

**CalculateFinalPrice**

Dla danej rezerwacji na dzień konferencji oblicza łączną należność.

ALTER FUNCTION [dbo].[CalculateFinalPrice]

(

@ReservationID int

)

RETURNS money

AS

BEGIN

DECLARE @BasePrice money;

DECLARE @StudentDiscount float;

DECLARE @Students int;

DECLARE @Participants int;

DECLARE @PriceIncreasePerDay money;

DECLARE @ConfDate date;

DECLARE @ReservationDate date;

SELECT @BasePrice = C.BasePrice,

@StudentDiscount = C.StudentDiscount,

@PriceIncreasePerDay = C.PriceIncreasePerDay,

@ConfDate = C.AddedOn,

@Students = CR.NumberOfStudents,

@Participants = CR.ParticipantsNumber,

@ReservationDate =CR.ReservationDate

FROM [Conference Reservations] AS CR

INNER JOIN [Conference Days] AS CD

ON CR.ReservationID = @ReservationID

AND CR.ConferenceDayID = CD.ConferenceDayID

INNER JOIN Conferences AS C

ON CD.ConferenceID = C.ConferenceID

DECLARE @NormalPrice money = @BasePrice

+ DATEDIFF(day, @ConfDate, @ReservationDate)\*@PriceIncreasePerDay

DECLARE @StudentPrice money = @NormalPrice \* (1 - @StudentDiscount)

RETURN @NormalPrice\*(@Participants - @Students)

+ @StudentPrice \* @Students

END

**ShouldBeCancelled**

Dla danej rezerwacji, sprawdza czy przekroczono termin płatności i czy należy anulować rezerwację.

ALTER FUNCTION [dbo].[ShouldBeCancelled]

(

@ReservationID int

)

RETURNS bit

AS

BEGIN

DECLARE @Deadline date;

DECLARE @ToPay money;

SELECT @Deadline = CR.PaymentDeadline, @ToPay = CR.ToPay

FROM [Conf Reservations Payments Balance] AS CR

WHERE CR.ReservationID = @ReservationID

DECLARE @Result bit;

SET @Result =IIF((@ToPay > 0 and getdate() > @Deadline), 1, 0)

RETURN @Result

END

**CanRegisterForWorkshop**

Dla danego uczestnika konferencji sprawdza czy może ona zarezerwować miejsce na dany warsztat.

ALTER FUNCTION [dbo].[CanRegisterForWorkshop]

(

@ParticipantID int,

@WorkshopID int

)

RETURNS bit

AS

BEGIN

DECLARE @Result bit

DECLARE @ConferenceDayID int;

DECLARE @WorkshopStartDate date;

DECLARE @WorkshopEndDate date;

SELECT @ConferenceDayID = W.ConferenceDayID,

@WorkshopStartDate = W.StartDate,

@WorkshopEndDate = W.EndDate

FROM Workshops AS W

WHERE W.WorkshopID = @WorkshopID

IF @ParticipantID not in (

SELECT P.ParticipantID

FROM [Conference Days] AS CD

INNER JOIN [Conference Reservations] AS CR

ON CD.ConferenceDayID = @ConferenceDayID AND CD.ConferenceDayID = CR.ConferenceDayID

INNER JOIN Participants AS P

ON CR.ReservationID = P.ReservationID

)

BEGIN

RETURN 0;

END

DECLARE @count INT;

SELECT @count = COUNT(\*)

FROM Workshops W

INNER JOIN [Workshop Reservations] AS WR

ON W.WorkshopID = WR.WorkshopID

INNER JOIN Participants AS P

ON P.ParticipantID = @ParticipantID AND WR.ParticipantID = P.ParticipantID

WHERE W.StartDate <= @WorkshopEndDate AND W.EndDate >= @WorkshopStartDate

RETURN iif(@count > 0, 0, 1);

END

1. **Procedury**
2. **Abc**
3. **Trigery**
4. **Abc**
5. **Indeksy**
6. **Abc**
7. **Warunki integralności**
8. **Abc**
9. **Role**