МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”
Кафедра систем штучного інтелекту

Лабораторна робота № 4

з дисципліни
«Організація баз даних та знань»

Виконав:
студент групи ШІ-23
Полуліх Ю.Ю.
Викладач:
Поберейко П. Б.

Львів – 2023 р.

Тема: Написання збережуваних програмних конструкцій

Мета: Навчитися розробляти та виконувати збережені процедури та функції, механізми використання транзакцій, тригерів для забезпечення цілісності значень, обмеження вводу даних, забезпечення кардинальності для таблиць, автоматичної корекції введених даних засобами SQL.

Хід роботи

1. **Створити користувацькі функції для реалізації автоматизованих операцій над даними:**

а. Обчислення статистичних даних;

```
DELIMITER $$

CREATE FUNCTION AvgAge()

RETURNS INT DETERMINISTIC

BEGIN

  SELECT AVG(TIMESTAMPDIFF(YEAR, p.DateOfBirth, CURDATE())) INTO @result

  FROM passengers p;

  RETURN @result;

END $$

DELIMITER ;
```

Функція шукає середній вік користувачів та виводить його. Результат виконання:

б. Вибірка даних за ключем між таблицями.

```
DELIMITER $$
CREATE FUNCTION CountFlightsForAirline(AirlineId INT)
returns INT deterministic
begin
        select count(*) into @result
    from Flights
    where AirlineId = ID;
    return @result;
end $$
delimiter ;
```

Функція повертає кількість рейсів, пов’язаних з цією авіакомпанією.
select CountFlightsForAirline(6);

Результат:

2. Створити збережувані процедури:

a. Параметризована вставка нових значень у таблиці:

```
USE airlinecompany;
DELIMITER $$
CREATE PROCEDURE InsertInPassengers(
    IN name VARCHAR(20),
    IN DateOfBirth DATE,
    IN passport INT,
    IN ContactInformation VARCHAR(50)
```

```
)
BEGIN
    INSERT INTO passengers (Name, DateOfBirth, Passport, ContactInformation)
    VALUES (name, DateOfBirth, passport, ContactInformation);
END $$
DELIMITER ;


CALL InsertInPassengers('Yurii', '1940-03-20', 123456789, 'yurii4@gmail.com');
```

Id	Name	DateOfBirth	Passport	ContactInformation
1	John Doe	1980-05-15	123456789	john@example.com
2	Jane Smith	1995-10-20	987654321	jane@example.com
3	David Lee	1988-07-12	456789123	david@example.com
7	Yurii	1940-03-20	123456689	yurii4@gmail.com

б. Реалізація зв’язку M:M між таблицями;

```
DELIMITER $$
CREATE PROCEDURE AddStaffAirplaneRelation(
    IN _StaffId INT,
    IN _AirplaneId INT
)
BEGIN
    INSERT INTO Staff_Airplane (StaffId, AirplaneId)
    VALUES (_StaffId, _AirplaneId);
END $$
DELIMITER ;
```

с. Динамічний SQL з використанням курсорів.

```
DELIMITER $$
```

```
CREATE PROCEDURE GetFlightDetailsForAirline(IN _AirlineId INT)
BEGIN
    DECLARE _FlightId INT;
    DECLARE _AirplaneId INT;
    DECLARE _PointOfDeparture VARCHAR(20);
    DECLARE _Destination VARCHAR(20);
    DECLARE _DateTimeOfDeparture DATETIME;
    DECLARE _DateTimeOfArrival DATETIME;
    DECLARE _Done INT DEFAULT FALSE;

    DECLARE flight_cursor CURSOR FOR
        SELECT Id, AirplaneId, PointOfDeparture, Destination, DateTimeOfDeparture,
DateTimeOfArrival
        FROM Flights
        WHERE AirlineId = _AirlineId;
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET _Done = TRUE;

    OPEN flight_cursor;

    LOOP_flight_details: LOOP
        FETCH flight_cursor INTO _FlightId, _AirplaneId, _PointOfDeparture,
_Destination, _DateTimeOfDeparture, _DateTimeOfArrival;
        IF _Done THEN
            LEAVE LOOP_flight_details;
```

```
        END IF;


        SET @DynamicSQL = CONCAT('SELECT a.ModelName, a.Manufacturer, ',

                                 '(SELECT COUNT(*) FROM Booking WHERE FlightId = ',
_FlightId, ') AS TotalBookings ',

                                 'FROM Airplane a WHERE a.Id = ', _AirplaneId);


        PREPARE stmt FROM @DynamicSQL;

        EXECUTE stmt;

        DEALLOCATE PREPARE stmt;


        SELECT _FlightId AS FlightID, _PointOfDeparture, _Destination,
_DateTimeOfDeparture, _DateTimeOfArrival;

    END LOOP;


    CLOSE flight_cursor;

END$$


DELIMITER ;
```

Процедура буде використовувати AirlineId для вибірки даних з таблиці Flights, після чого для кожного рейсу буде знаходити відповідні літаки та інформацію про бронювання.

```
call GetFlightDetailsForAirline(3);
```

FlightID	_PointOfDepartu...	_Destination	_DateTimeOfDepart...	_DateTimeOfArrival
8	Galaxy	Shoreline	2023-11-12 07:45:00	2023-11-12 10:50:00

3. Розробка та застосування транзакцій

a. Застосувати механізми використання транзакцій:

START TRANSACTION;

SELECT NumberOfSeats FROM Airplane

WHERE Id = (SELECT AirplaneId FROM Flights WHERE Id = 1)

FOR UPDATE;

INSERT INTO Booking (AirlineId, FlightId, PassengerId, Date, PlaceOfPassenger, Status)

VALUES (1, 2, 2, CURDATE(), 3, 'Confirmed');

UPDATE Airplane SET NumberOfSeats = NumberOfSeats - 1

WHERE Id = (SELECT AirplaneId FROM Flights WHERE Id = 1);

COMMIT;

У цій транзакції перевіряється доступність місць у літаку і закриваться для редагування, додається бронювання з відповідними даними та кількість доступниї місці у літаку зменшується на 1. Робимо commit після перевірки даних.

b. Розроблення SQL запитів в рамках однієї транзакції

START TRANSACTION;

INSERT INTO Staff (Name, Position, DateOfEmployment, ContactPhone, FlightId, AirlineId)
VALUES ('John Doe', 'Pilot', '2023-01-15', '555-1234', 1, 1);

UPDATE Staff
SET ContactPhone = ,555-4321'
WHERE Id = LAST_INSERT_ID();

COMMIT;

Створюємо в рамках однієї транзакції запит, що додає одного співробітника та запит, що оновлює його номер телефону. Після перевірки даних робимо commit.

4. Розробка та застосування тригерів:

a. забезпечення цілісності значень:

```
DELIMITER //
CREATE TRIGGER UpdateSeatsAfterBooking AFTER INSERT ON Booking
FOR EACH ROW
BEGIN
   UPDATE Airplane
   SET NumberOfSeats = NumberOfSeats - 1
   WHERE Id = (SELECT AirplaneId FROM Flights WHERE Id = NEW.FlightId);
END;
//
DELIMITER ;
```

Тригер призначений для автоматичного оновлення кількості місць в літаку в таблиці Airplane після кожного нового бронювання, що додається в таблицю Booking.

b. обмеження вводу даних:

```
DELIMITER //
CREATE TRIGGER BeforeInsertFlight BEFORE INSERT ON Flights
FOR EACH ROW
BEGIN
   IF NEW.DateTimeOfDeparture < NOW() THEN
      SIGNAL SQLSTATE '45000'
         SET MESSAGE_TEXT = 'The departure time cannot be in the past.';
   END IF;
END;
//
DELIMITER ;
```

Тригер для таблиці Flights, щоб перевірити, що дата та час вильоту не з минулого.

c. забезпечення кардинальності для таблиць

```
DELIMITER //

CREATE TRIGGER EnsureOneAirplanePerFlight BEFORE INSERT ON Flights
FOR EACH ROW
BEGIN
   DECLARE airplane_count INT;
   SELECT COUNT(*) INTO airplane_count FROM Flights WHERE AirplaneId =
NEW.AirplaneId;
   IF airplane_count > 0 THEN
      SIGNAL SQLSTATE '45000'
```

```
        SET MESSAGE_TEXT = 'An airplane can only be assigned to one flight at a
time.';
    END IF;
END;

//
DELIMITER ;
```
Тригер, який перевірятиме чи до одного рейсу Flight може бути прив'язано не більше одного літака Airplane перед вставкою нового запису.

d. автоматична корекція введених даних.
```
DELIMITER //

CREATE TRIGGER DefaultIATACode BEFORE INSERT ON Airlines
FOR EACH ROW
BEGIN
    IF CHAR_LENGTH(NEW.IATA_Code) <> 3 THEN

        SET NEW.IATA_Code = 'DEF';
    END IF;
END;

//
DELIMITER ;
```

Встановлює стандартний IATA код, якщо введений код не є трьохбуквенним.

Висновок: на цій лабораторній роботі я навчився розробляти та використовувати збережені процедури і функції, ознайомився із механізмом транзакцій та було розглянуто тригери, їх призначення, створення та використання у СУБД.