Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ № 5

«процедуры, функции, триггеры в PostgreSQL»
по дисциплине «Проектирование и реализация баз данных»

Обучающийся Федоров Даниил Михайлович
Факультет прикладной информатики
Группа K3240
Направление подготовки 09.03.03 Прикладная информатика
Образовательная программа Мобильные и сетевые технологии 2023
Преподаватель Говорова Марина Михайловна

Санкт-Петербург
2024/2025

СОДЕРЖАНИЕ

Стр.

1 Цель работы

Цель работы: овладеть практическими создания и использования процедур, функций и триггеров в базе данных PostgreSQL.

2 Практическое задание

1. Создать 3 процедуры для индивидуальной БД согласно варианту (часть 4 ЛР 2). Допустимо использование IN/OUT параметров. Допустимо создать авторские процедуры. (3 балла)
2. Создать триггеры для индивидуальной БД согласно варианту:
   Вариант 2.1. 3 триггера - 3 балла (min). Допустимо использовать триггеры логирования из практического занятия по функциям и триггерам.
   Вариант 2.2. 7 оригинальных триггеров - 7 баллов (max).

4 Выполнение

4.1 Процедуры и функции

- Для поиска билетов в заданный пункт назначения

```
CREATE OR REPLACE FUNCTION find_tickets_dest(dest_isao VARCHAR)
RETURNS TABLE(
        ticket_id INT
) AS $$
BEGIN
        RETURN QUERY
        SELECT ticket.ticket_id
        FROM airport_scheme.ticket
        JOIN airport_scheme.flight ON flight.flight_id = ticket.flight_id
        JOIN airport_scheme.route ON route.route_id = flight.route_id
        JOIN airport_scheme.airport ON route.arrival_airport_id =
airport.airport_id
        WHERE airport.code_isao = dest_isao;
END;
$$ LANGUAGE plpgsql;
```

```
airport_db=# CREATE OR REPLACE FUNCTION find_tickets_dest(dest_isao VARCHAR)
airport_db-# RETURNS TABLE(
airport_db(# ticket_id INT
airport_db(# ) AS $$
airport_db$# BEGIN
airport_db$# RETURN QUERY
airport_db$# SELECT ticket.ticket_id
airport_db$# FROM airport_scheme.ticket
airport_db$# JOIN airport_scheme.flight ON flight.flight_id = ticket.flight_id
airport_db$# JOIN airport_scheme.route ON route.route_id = flight.route_id
airport_db$# JOIN airport_scheme.airport ON route.arrival_airport_id = airport.airport_id
airport_db$# WHERE airport.code_isao = dest_isao;
airport_db$# END;
airport_db$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
airport_db=#
```

SELECT * FROM find_tickets_dest('UUEE');

```
airport_db=# SELECT * FROM find_tickets_dest('UUEE');
 ticket_id
-----------
        40
(1 row)
```

- Создания новой кассы продажи билетов.

```
CREATE OR REPLACE PROCEDURE create_cashdesc(cashdesk_id INT,
cashdesk_number INT, is_online BOOLEAN)
AS $$
BEGIN
```

```
    INSERT INTO airport_scheme.cash_desk(cashdesk_id, cashdesk_number,
is_online)
    VALUES (cashdesk_id, cashdesk_number, is_online);
END;
$$ LANGUAGE plpgsql;
```

```
airport_db=# CREATE OR REPLACE PROCEDURE create_cashdesc(cashdesk_id INT, cashdesk_number INT, is_online BOOLEAN)
airport_db-# AS $$
airport_db$# BEGIN
airport_db$# INSERT INTO airport_scheme.cash_desk(cashdesk_id, cashdesk_number, is_online)
airport_db$# VALUES (cashdesk_id, cashdesk_number, is_online);
airport_db$# END;
airport_db$# $$ LANGUAGE plpgsql;
CREATE PROCEDURE
```

```
CALL create_cashdesc(41, 141, TRUE);
```

```
airport_db=# CALL create_cashdesc(41, 141, TRUE);
CALL
```

- Определить расход топлива по всем маршрутам за истекший месяц.

```
CREATE OR REPLACE FUNCTION fuel_consumption_last_month()
RETURNS TABLE(
    route_id INT,
    fuel_consumption NUMERIC
)
AS $$
BEGIN
    RETURN QUERY
    SELECT
        route.route_id,
        SUM(aircraft.fuel_consumption * (
      EXTRACT(EPOCH FROM (
        (CAST(flight.arrival_date AS timestamp) + flight.arrival_time) -
        (CAST(flight.departure_date AS timestamp) + flight.departure_time)
      )) / 3600
    )) AS fuel_consumption
    FROM airport_scheme.flight
    JOIN airport_scheme.route ON route.route_id = flight.route_id
    JOIN airport_scheme.aircraft ON aircraft.aircraft_id = flight.aircraft_id
    WHERE flight.departure_date BETWEEN (CURRENT_DATE -
INTERVAL '1 month') and CURRENT_DATE
    GROUP BY route.route_id;
END;
$$ LANGUAGE plpgsql;
```

```
airport_db=# CREATE OR REPLACE FUNCTION fuel_consumption_last_month()
airport_db-# RETURNS TABLE(
airport_db(# route_id INT,
airport_db(# fuel_consumption NUMERIC
airport_db(# )
airport_db-# AS $$
airport_db$# BEGIN
airport_db$# RETURN QUERY
airport_db$# SELECT
airport_db$# route.route_id,
airport_db$# SUM(aircraft.fuel_consumption * (
airport_db$#             EXTRACT(EPOCH FROM (
airport_db$#                 (CAST(flight.arrival_date AS timestamp) + flight.arrival_time) -
airport_db$#                 (CAST(flight.departure_date AS timestamp) + flight.departure_time)
airport_db$#             )) / 3600
airport_db$#         )) AS fuel_consumption
airport_db$# FROM airport_scheme.flight
airport_db$# JOIN airport_scheme.route ON route.route_id = flight.route_id
airport_db$# JOIN airport_scheme.aircraft ON aircraft.aircraft_id = flight.aircraft_id
airport_db$# WHERE flight.departure_date BETWEEN (CURRENT_DATE - INTERVAL '1 month') and CURRENT_DATE
airport_db$# GROUP BY route.route_id;
airport_db$# END;
airport_db$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
```

SELECT * FROM fuel_consumption_last_month();

```
airport_db=# SELECT * FROM fuel_consumption_last_month();
 route_id |    fuel_consumption
----------+------------------------
       29 | 10.600000000000000000
       34 |  9.200000000000000000
       32 | 10.200000000000000000
       10 |  9.200000000000000000
        9 | 10.000000000000000000
        7 | 10.600000000000000000
       35 | 10.000000000000000000
       15 | 10.200000000000000000
        6 |  8.400000000000000000
       26 |  9.400000000000000000
       12 | 14.200000000000000000
       24 |  9.200000000000000000
       19 | 13.200000000000000000
       36 |  9.600000000000000000
       25 | 10.000000000000000000
       31 | 10.800000000000000000
       30 |  9.800000000000000000
       21 | 10.200000000000000000
       14 |  9.600000000000000000
       17 | 10.400000000000000000
       28 | 14.000000000000000000
       22 |  9.000000000000000000
       20 | 14.400000000000000000
       33 | 10.400000000000000000
       13 |  9.800000000000000000
       18 |  8.800000000000000000
       16 |  9.800000000000000000
       27 | 12.600000000000000000
       23 | 10.400000000000000000
       11 | 13.000000000000000000
        8 |  9.400000000000000000
(31 rows)
```

4.2 Триггеры

- Триггер для автоматического обновления даты последнего изменения записи в таблице ticket

```
airport_db=# CREATE OR REPLACE FUNCTION update_last_updated()
airport_db-# RETURNS TRIGGER AS $$
airport_db$# BEGIN
airport_db$# NEW.last_updated = CURRENT_TIMESTAMP;
airport_db$# RETURN NEW;
airport_db$# END;
airport_db$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
airport_db=# CREATE TRIGGER trigger_update_last_updated
airport_db-# BEFORE UPDATE ON airport_scheme.ticket
airport_db-# FOR EACH ROW
airport_db-# EXECUTE FUNCTION update_last_updated();
CREATE TRIGGER
```

```
CREATE OR REPLACE FUNCTION airport_scheme.update_last_updated()
RETURNS TRIGGER AS $$
BEGIN
	NEW.last_updated = CURRENT_TIMESTAMP;
	RETURN NEW;
END;
$$ LANGUAGE plpgsql;


CREATE TRIGGER airport_scheme.trigger_update_last_updated
BEFORE UPDATE ON airport_scheme.ticket
FOR EACH ROW
EXECUTE FUNCTION airport_scheme.update_last_updated();
```

- Триггер для проверки наличия места на рейсе при добавлении нового билета

```
airport_db=# CREATE OR REPLACE FUNCTION airport_scheme.check_seat_availability()
airport_db-# RETURNS TRIGGER AS $$
airport_db$# BEGIN
airport_db$# IF (SELECT COUNT(*) FROM airport_scheme.ticket WHERE flight_id = NEW.flight_id) >=
airport_db$# (SELECT seat_capacity FROM airport_scheme.aircraft WHERE aircraft_id =
airport_db$# (SELECT aircraft_id FROM airport_scheme.flight WHERE flight_id = NEW.flight_id)) THEN
airport_db$# RAISE EXCEPTION 'Нет мест на этом рейсе';
airport_db$# END IF;
airport_db$# RETURN NEW;
airport_db$# END;
airport_db$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
airport_db=# CREATE TRIGGER trigger_check_seat_availability
airport_db-# BEFORE INSERT ON airport_scheme.ticket
airport_db-# FOR EACH ROW
airport_db-# EXECUTE FUNCTION airport_scheme.check_seat_availability();
CREATE TRIGGER
airport_db=#
```

```
CREATE OR REPLACE FUNCTION
airport_scheme.check_seat_availability()
```

```
RETURNS TRIGGER AS $$
BEGIN
	IF (SELECT COUNT(*) FROM airport_scheme.ticket WHERE
flight_id = NEW.flight_id) >=
		(SELECT seat_capacity FROM airport_scheme.aircraft
WHERE aircraft_id =
		(SELECT aircraft_id FROM airport_scheme.flight WHERE
flight_id = NEW.flight_id)) THEN
		RAISE EXCEPTION 'Нет мест на этом рейсе';
	END IF;
	RETURN NEW;
END;
$$ LANGUAGE plpgsql;


CREATE TRIGGER trigger_check_seat_availability
BEFORE INSERT ON airport_scheme.ticket
FOR EACH ROW
EXECUTE FUNCTION airport_scheme.check_seat_availability();
```

- Триггер для ограничения количества касс

```
airport_db=# CREATE OR REPLACE FUNCTION airport_scheme.check_cashdesk_limit()
airport_db-# RETURNS TRIGGER AS $$
airport_db$# BEGIN
airport_db$# IF (SELECT COUNT(*) FROM airport_scheme.cash_desk) >= 50 THEN
airport_db$# RAISE EXCEPTION 'Достигнут лимит касс';
airport_db$# END IF;
airport_db$# RETURN NEW;
airport_db$# END;
airport_db$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
airport_db=# CREATE TRIGGER trigger_check_cashdesk_limit
airport_db-# BEFORE INSERT ON airport_scheme.cash_desk
airport_db-# FOR EACH ROW
airport_db-# EXECUTE FUNCTION airport_scheme.check_cashdesk_limit();
CREATE TRIGGER
airport_db=#
```

```
CREATE OR REPLACE FUNCTION
airport_scheme.check_cashdesk_limit()
RETURNS TRIGGER AS $$
BEGIN
	IF (SELECT COUNT(*) FROM airport_scheme.cash_desk) >= 50
THEN
		RAISE EXCEPTION 'Достигнут лимит касс';
	END IF;
	RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trigger_check_cashdesk_limit
```

```
BEFORE INSERT ON airport_scheme.cash_desk
FOR EACH ROW
EXECUTE FUNCTION airport_scheme.check_cashdesk_limit();
```

- TT

Выводы

В данной лабораторной работе были получены основные навыки создания функций, процедур и триггеров , а также навыки по работе в plpgsql.