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 Аплеев Дмитрий Артурович
Факультет прикладной информатики
Группа K3239
Направление подготовки 09.03.03 Прикладная информатика
Образовательная программа Мобильные и сетевые технологии
2023 **Преподаватель** Говорова Марина Михайловна

Санкт-Петербург
2024/2025

СОДЕРЖАНИЕ

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. Допустимо использование IN/OUT параметров. Допустимо создать авторские процедуры.
2. Создать триггеры для индивидуальной БД

3. Выполнение ЛР5

3.1 Процедуры и функции

Процедура №1: Изменение статуса акции

```
   CREATE OR REPLACE PROCEDURE change_stst_prom(p_promo_id INTEGER)
   LANGUAGE plpgsql AS $$
   BEGIN
     UPDATE promotion
        SET is_active = NOT is_active
      WHERE promotion_id = p_promo_id;
   END;
``` $$;
```

```
Hotel4.0=# CALL change_stst_prom(5);
CALL
Hotel4.0=# SELECT * FROM promotion;
 promotion_id | type_id | discount_value | is_active | start_date | end_date
--------------+---------+----------------+-----------+------------+------------
            1 | 2 | 10.00 | t | 2025-06-01 | 2025-08-31
            2 | 4 | 15.00 | t | 2025-12-15 | 2026-01-15
            3 | 1 | 5.00 | f | 2025-03-01 | 2025-05-31
            4 | 5 | 20.00 | t | 2025-09-01 | 2025-11-30
            5 | 3 | 12.50 | t | 2026-04-01 | 2026-05-15
(5 rows)

Hotel4.0=# CALL change_stst_prom(5);
CALL
Hotel4.0=# SELECT * FROM promotion;
 promotion_id | type_id | discount_value | is_active | start_date | end_date
--------------+---------+----------------+-----------+------------+------------
            1 | 2 | 10.00 | t | 2025-06-01 | 2025-08-31
            2 | 4 | 15.00 | t | 2025-12-15 | 2026-01-15
            3 | 1 | 5.00 | f | 2025-03-01 | 2025-05-31
            4 | 5 | 20.00 | t | 2025-09-01 | 2025-11-30
            5 | 3 | 12.50 | f | 2026-04-01 | 2026-05-15
(5 rows)
```
```

Процедура №2: Изменение зарплаты

```
Hotel4.0=# CREATE OR REPLACE PROCEDURE change_salary(
Hotel4.0(#     p_post_id INTEGER,
Hotel4.0(#     p_zp NUMERIC)
Hotel4.0-# LANGUAGE plpgsql AS $$
Hotel4.0$# BEGIN
Hotel4.0$#   UPDATE post_list
Hotel4.0$#      SET salary = salary + p_zp
Hotel4.0$#    WHERE post_id = p_post_id;
Hotel4.0$# END; $$;
```

```
``CREATE PROCEDURE
```

```
Hotel4.0=# SELECT * FROM post_list;
 post_id |     title     | salary
---------+---------------+--------
       1 | Manager       |  90000
       2 | Administrator |  65000
       3 | Housekeeper   |  45000
       4 | Porter        |  42000
       5 | Security      |  40000
       6 | Chef          |  95000
(6 rows)

Hotel4.0=# call change_salary(1,10000);
CALL
Hotel4.0=# SELECT * FROM post_list;
 post_id |     title     | salary
---------+---------------+--------
       2 | Administrator |  65000
       3 | Housekeeper   |  45000
       4 | Porter        |  42000
       5 | Security      |  40000
       6 | Chef          |  95000
       1 | Manager       | 100000
(6 rows)

Hotel4.0=# call change_salary(1,-10000);
CALL
Hotel4.0=# SELECT * FROM post_list;
 post_id |     title     | salary
---------+---------------+--------
       2 | Administrator |  65000
       3 | Housekeeper   |  45000
       4 | Porter        |  42000
       5 | Security      |  40000
       6 | Chef          |  95000
       1 | Manager       |  90000
(6 rows)
```

Процедура №3: Изменение стоимости типа номера

```
Hotel4.0=# CREATE OR REPLACE PROCEDURE upd_room_prc( p_type_id INTEGER, p_new_base NUMERIC)

LANGUAGE plpgsql
AS $$
BEGIN
  UPDATE room_type
     SET base_price = p_new_base,
         current_price = GREATEST(current_price, p_new_base)
   WHERE type_id = p_type_id;
END;
$$;
CREATE PROCEDURE
Hotel4.0=# SELECT * FROM room_type_price;
 room_type_price_id | type_id | start_date |  end_date  |  price
--------------------+---------+------------+------------+----------
                261 |       1 | 2025-06-01 | 2025-08-31 |  7000.00
                262 |       2 | 2025-06-01 | 2025-08-31 |  9200.00
                263 |       3 | 2025-06-01 | 2025-08-31 |  8800.00
                264 |       4 | 2025-06-01 | 2025-08-31 | 14000.00
                265 |       5 | 2025-06-01 | 2025-08-31 | 18000.00
                266 |       1 | 2025-09-01 | 2025-11-30 |  6500.00
                267 |       2 | 2025-09-01 | 2025-11-30 |  8800.00
                268 |       3 | 2025-09-01 | 2025-11-30 |  8200.00
                269 |       4 | 2025-09-01 | 2025-11-30 | 13200.00
                270 |       5 | 2025-09-01 | 2025-11-30 | 17000.00
                271 |       1 | 2025-12-01 | 2026-02-28 |  6000.00
                272 |       2 | 2025-12-01 | 2026-02-28 |  8000.00
                273 |       3 | 2025-12-01 | 2026-02-28 |  7600.00
                274 |       4 | 2025-12-01 | 2026-02-28 | 12500.00
                275 |       5 | 2025-12-01 | 2026-02-28 | 16000.00
                276 |       1 | 2026-03-01 | 2026-05-31 |  6800.00
                277 |       2 | 2026-03-01 | 2026-05-31 |  9000.00
                278 |       3 | 2026-03-01 | 2026-05-31 |  8400.00
                279 |       4 | 2026-03-01 | 2026-05-31 | 13500.00
                280 |       5 | 2026-03-01 | 2026-05-31 | 17500.00
(20 rows)

Hotel4.0=# call upd_room_prc(1,1000);
CALL
Hotel4.0=# SELECT * FROM room_type_price;
 room_type_price_id | type_id | start_date |  end_date  |  price
--------------------+---------+------------+------------+----------
                262 |       2 | 2025-06-01 | 2025-08-31 |  9200.00
                263 |       3 | 2025-06-01 | 2025-08-31 |  8800.00
                264 |       4 | 2025-06-01 | 2025-08-31 | 14000.00
                265 |       5 | 2025-06-01 | 2025-08-31 | 18000.00
                267 |       2 | 2025-09-01 | 2025-11-30 |  8800.00
                268 |       3 | 2025-09-01 | 2025-11-30 |  8200.00
                269 |       4 | 2025-09-01 | 2025-11-30 | 13200.00
                270 |       5 | 2025-09-01 | 2025-11-30 | 17000.00
                272 |       2 | 2025-12-01 | 2026-02-28 |  8000.00
                273 |       3 | 2025-12-01 | 2026-02-28 |  7600.00
                274 |       4 | 2025-12-01 | 2026-02-28 | 12500.00
                275 |       5 | 2025-12-01 | 2026-02-28 | 16000.00
                277 |       2 | 2026-03-01 | 2026-05-31 |  9000.00
                278 |       3 | 2026-03-01 | 2026-05-31 |  8400.00
                279 |       4 | 2026-03-01 | 2026-05-31 | 13500.00
                280 |       5 | 2026-03-01 | 2026-05-31 | 17500.00
                261 |       1 | 2025-06-01 | 2025-08-31 |  1000.00
                266 |       1 | 2025-09-01 | 2025-11-30 |  1000.00
                271 |       1 | 2025-12-01 | 2026-02-28 |  1000.00
                276 |       1 | 2026-03-01 | 2026-05-31 |  1000.00
(20 rows)
```

3.2 Триггеры

Триггер №1: Авто-закрытие просроченного контракта

```
Hotel4.0=# CREATE OR REPLACE FUNCTION trgfn_autoclose_contract()
Hotel4.0-# RETURNS trigger
Hotel4.0-# LANGUAGE plpgsql
Hotel4.0-# AS $$
Hotel4.0$# BEGIN
Hotel4.0$#   IF NEW.contract_end_date IS NOT NULL
Hotel4.0$#      AND NEW.contract_end_date < CURRENT_DATE THEN
Hotel4.0$#      NEW.status := FALSE;
Hotel4.0$#   END IF;
Hotel4.0$#   RETURN NEW;
Hotel4.0$# END;
Hotel4.0$# $$;
CREATE FUNCTION
Hotel4.0=#
```

Триггер №2: Планирование уборки после выезда

```
Hotel4.0=# CREATE OR REPLACE FUNCTION trgfn_schedule_cleaning()
Hotel4.0-# RETURNS trigger
Hotel4.0-# LANGUAGE plpgsql
Hotel4.0-# AS $$
Hotel4.0$# BEGIN
Hotel4.0$#   IF TG_OP = 'UPDATE'
Hotel4.0$#      AND OLD.booking_status <> 'checked_out'
Hotel4.0$#      AND NEW.booking_status  = 'checked_out' THEN
Hotel4.0$#     INSERT INTO cleaning(room_id, scheduled_at, status)
Hotel4.0$#     VALUES (NEW.room_id,
Hotel4.0$#             NEW.departure_date + INTERVAL '1 hour',
Hotel4.0$#             'scheduled');
Hotel4.0$#   END IF;
Hotel4.0$#   RETURN NEW;
Hotel4.0$# END;
Hotel4.0$# $$;
CREATE FUNCTION
Hotel4.0=# CREATE TRIGGER trg_schedule_cleaning
Hotel4.0-#   AFTER UPDATE OF booking_status
Hotel4.0-#   ON order_item
Hotel4.0-#   FOR EACH ROW
Hotel4.0-#   EXECUTE FUNCTION trgfn_schedule_cleaning();
CREATE TRIGGER
Hotel4.0=#
```

Триггер №3: Пересчёт общей суммы заказа

```
Hotel4.0=# CREATE OR REPLACE FUNCTION trgfn_recalc_order_total()
Hotel4.0-# RETURNS trigger
Hotel4.0-# LANGUAGE plpgsql
Hotel4.0-# AS $$
Hotel4.0$# DECLARE
Hotel4.0$#   v_oid integer := COALESCE(NEW.order_id, OLD.order_id);
Hotel4.0$# BEGIN
Hotel4.0$#   UPDATE orders
Hotel4.0$#      SET amount = COALESCE(
Hotel4.0$#            (SELECT SUM(booking_amount)
Hotel4.0$#               FROM order_item
Hotel4.0$#              WHERE order_id = v_oid), 0)
Hotel4.0$#    WHERE order_id = v_oid;
Hotel4.0$#   RETURN NULL;
Hotel4.0$# END;
Hotel4.0$# $$;
CREATE FUNCTION
Hotel4.0=# CREATE TRIGGER trg_recalc_order_total
Hotel4.0-#   AFTER INSERT OR UPDATE OR DELETE
Hotel4.0-#   ON order_item
Hotel4.0-#   FOR EACH ROW
Hotel4.0-#   EXECUTE FUNCTION trgfn_recalc_order_total();
CREATE TRIGGER
Hotel4.0=#
```

Триггер №4: Расчёт стоимости позиции бронирования

```
Hotel4.0=# CREATE OR REPLACE FUNCTION trgfn_calc_item_price()
Hotel4.0-# RETURNS trigger
Hotel4.0-# LANGUAGE plpgsql
Hotel4.0-# AS $$
Hotel4.0$# DECLARE
Hotel4.0$#   v_type_id integer;
Hotel4.0$#   v_price   numeric;
Hotel4.0$#   v_disc    numeric := 0;
Hotel4.0$#   v_days    integer;
Hotel4.0$# BEGIN
Hotel4.0$#   SELECT r.type_id
Hotel4.0$#     INTO v_type_id
Hotel4.0$#     FROM room r
Hotel4.0$#    WHERE r.room_id = NEW.room_id;
Hotel4.0$#
Hotel4.0$#   SELECT price
Hotel4.0$#     INTO v_price
Hotel4.0$#     FROM room_type_price p
Hotel4.0$#    WHERE p.type_id = v_type_id
Hotel4.0$#      AND NEW.arrival_date BETWEEN p.start_date AND p.end_date
Hotel4.0$#    ORDER BY p.start_date DESC
Hotel4.0$#    LIMIT 1;
Hotel4.0$#
Hotel4.0$#   IF v_price IS NULL THEN
Hotel4.0$#      SELECT base_price
Hotel4.0$#        INTO v_price
Hotel4.0$#        FROM room_type
Hotel4.0$#       WHERE type_id = v_type_id;
Hotel4.0$#   END IF;
Hotel4.0$#
Hotel4.0$#   SELECT discount_value
Hotel4.0$#     INTO v_disc
Hotel4.0$#     FROM promotion pr
Hotel4.0$#    WHERE pr.type_id = v_type_id
Hotel4.0$#      AND pr.is_active
Hotel4.0$#      AND NEW.arrival_date BETWEEN pr.start_date AND pr.end_date
Hotel4.0$#    LIMIT 1;
Hotel4.0$#
Hotel4.0$#   v_disc := COALESCE(v_disc, 0);
Hotel4.0$#   v_days := NEW.departure_date - NEW.arrival_date;
Hotel4.0$#   NEW.booking_amount := v_days * v_price * (1 - v_disc / 100);
Hotel4.0$#
Hotel4.0$#   RETURN NEW;
Hotel4.0$# END;
Hotel4.0$# $$;
CREATE FUNCTION
```

Триггер №5: Заполнение значений по умолчанию (order_item)

```
Hotel4.0=# CREATE OR REPLACE FUNCTION trgfn_order_item_defaults()
Hotel4.0-# RETURNS trigger
Hotel4.0-# LANGUAGE plpgsql
Hotel4.0-# AS $$
Hotel4.0$# BEGIN
Hotel4.0$#   IF NEW.booking_status IS NULL THEN
Hotel4.0$#      NEW.booking_status := 'booked';
Hotel4.0$#   END IF;
Hotel4.0$#
Hotel4.0$#   IF NEW.departure_date IS NULL THEN
Hotel4.0$#      NEW.departure_date := NEW.arrival_date + 1;
Hotel4.0$#   END IF;
Hotel4.0$#
Hotel4.0$#   RETURN NEW;
Hotel4.0$# END;
Hotel4.0$# $$;
CREATE FUNCTION
Hotel4.0=#
```

Триггер №6: Блокировка заказа с просроченным паспортом

```
Hotel4.0=# CREATE OR REPLACE FUNCTION trgfn_reject_expired_passport()
Hotel4.0-# RETURNS trigger
Hotel4.0-# LANGUAGE plpgsql
Hotel4.0-# AS $$
Hotel4.0$# DECLARE
Hotel4.0$#   v_expiry date;
Hotel4.0$# BEGIN
Hotel4.0$#   SELECT passport_expiry_date
Hotel4.0$#     INTO v_expiry
Hotel4.0$#     FROM passport_data
Hotel4.0$#    WHERE passport_id = NEW.passport_id;
Hotel4.0$#
Hotel4.0$#   IF v_expiry < CURRENT_DATE THEN
Hotel4.0$#      RAISE EXCEPTION
Hotel4.0$#        'Паспорт клиента просрочен (до %). Заказ отклонён.', v_expiry;
Hotel4.0$#   END IF;
Hotel4.0$#
Hotel4.0$#   RETURN NEW;
Hotel4.0$# END;
Hotel4.0$# $$;
CREATE FUNCTION
Hotel4.0=#
```

Триггер №7: Блокировка заказа при неактивном контракте

```
Hotel4.0=# CREATE OR REPLACE FUNCTION trgfn_reject_inactive_contract()
Hotel4.0-# RETURNS trigger
Hotel4.0-# LANGUAGE plpgsql
Hotel4.0-# AS $$
Hotel4.0$# DECLARE
Hotel4.0$#   v_active boolean;
Hotel4.0$# BEGIN
Hotel4.0$#   SELECT status
Hotel4.0$#     INTO v_active
Hotel4.0$#     FROM employment_contract
Hotel4.0$#    WHERE contract_id = NEW.contract_id;
Hotel4.0$#
Hotel4.0$#   IF v_active IS FALSE THEN
Hotel4.0$#      RAISE EXCEPTION
Hotel4.0$#        'Контракт % неактивен. Заказ запрещён.', NEW.contract_i
Hotel4.0$#   END IF;
Hotel4.0$#
Hotel4.0$#   RETURN NEW;
Hotel4.0$# END;
Hotel4.0$# $$;
CREATE FUNCTION
Hotel4.0=#
Hotel4.0=# CREATE TRIGGER trg_reject_inactive_contract
Hotel4.0-#   BEFORE INSERT
Hotel4.0-#   ON orders
Hotel4.0-#   FOR EACH ROW
Hotel4.0-#   EXECUTE FUNCTION trgfn_reject_inactive_contract();
CREATE TRIGGER
Hotel4.0=#
```

``` **Выводы**

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