Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО
Факультет инфокоммуникационных технологий

ОТЧЁТ
О ЛАБОРАТОРНОЙ РАБОТЕ №3
Процедуры, функции, триггеры в PostgreSQL
по дисциплине:
ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ БАЗ ДАННЫХ

Выполнил: Пятыго Д. А.
Группа: K32421

Проверил: Говорова М. М.

Санкт-Петербург
2023

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

Практическое задание:

Вариант 2.2:

1. Создать процедуры/функции согласно индивидуальному заданию (вариант 11 — БД «Автомастерская»).
2. Создать авторский триггер по варианту индивидуального задания.

Выполнение:

Создать хранимые процедуры:

1. Повышения цены деталей для автомобиля “Ford” на 10%.

   Прежде всего, необходимо хранить ID обновлённых деталей, чтобы их цена не повышалась несколько раз. Для этого заведём следующую таблицу:

   ```
   CREATE TABLE "Updated_Details" (
     "ID_детали" INTEGER PRIMARY KEY
   );
   ```

   Код процедуры:

   ```
   CREATE OR REPLACE PROCEDURE increase_ford_details_price()
   LANGUAGE plpgsql
   AS $$
   BEGIN
     UPDATE "Деталь"
     SET "Стоимость" = "Стоимость" * 1.1
     WHERE "ID_детали" IN (
       SELECT DISTINCT "Состав_детали"."ID_детали"
       FROM "Состав_детали"
       JOIN "Состав_заказа" USING("ID_состава_заказа")
       JOIN "Заказ" USING("Номер_заказа")
       JOIN "Автомобиль" USING("Госномер")
   ```

```
  WHERE "Автомобиль"."Марка" = 'Ford' AND "Состав_детали"."Предоставлена_клиентом" = false
  )
  AND "ID_детали" NOT IN (SELECT "ID_детали" FROM "Updated_Details");

  INSERT INTO "Updated_Details" ("ID_детали")
  SELECT DISTINCT "Состав_детали"."ID_детали"
  FROM "Состав_детали"
  JOIN "Состав_заказа" USING("ID_состава_заказа")
  JOIN "Заказ" USING("Номер_заказа")
  JOIN "Автомобиль" USING("Госномер")
  WHERE "Автомобиль"."Марка" = 'Ford' AND "Состав_детали"."Предоставлена_клиентом" = false
  AND "Состав_детали"."ID_детали" NOT IN (SELECT "ID_детали" FROM "Updated_Details");
END;
$$;
```

Результат успешного создания процедуры приведён на рисунке 1.

```
automaster$#     FROM "Состав_детали"
automaster$#     JOIN "Состав_заказа" USING("ID_состава_заказа")
automaster$#     JOIN "Заказ" USING("Номер_заказа")
automaster$#     JOIN "Автомобиль" USING("Госномер")
automaster$#     WHERE "Автомобиль"."Марка" = 'Ford' AND "Состав_детали"."Предос
тавлена_клиентом" = false
automaster$#   )
automaster$#   AND "ID_детали" NOT IN (SELECT "ID_детали" FROM "Updated_Details"
);
automaster$#
automaster$#   INSERT INTO "Updated_Details" ("ID_детали")
automaster$#   SELECT DISTINCT "Состав_детали"."ID_детали"
automaster$#   FROM "Состав_детали"
automaster$#   JOIN "Состав_заказа" USING("ID_состава_заказа")
automaster$#   JOIN "Заказ" USING("Номер_заказа")
automaster$#   JOIN "Автомобиль" USING("Госномер")
automaster$#   WHERE "Автомобиль"."Марка" = 'Ford' AND "Состав_детали"."Предоста
влена_клиентом" = false
automaster$#   AND "Состав_детали"."ID_детали" NOT IN (SELECT "ID_детали" FROM "
Updated_Details");
automaster$# END;
automaster$# $$;
CREATE PROCEDURE
```

Рисунок 1. Результат создания процедуры 1

В базе есть Ford, которому требуется обновить масляный фильтр. На рисунке 2 приведена таблица с деталями до вызова процедуры и после.

	ID_детали	Название	Стоимость
1	1	Масляный фильтр	500
2	2	Тормозные колодки	2000
3	3	Свечи зажигания	1000
4	4	Провода высоковольтные	1500
5	5	Датчик температуры двигателя	3000
6	6	Радиатор охлаждения	5000
7	7	Ремень ГРМ	4000
8	8	Аудиосистема	8000

	ID_детали ^ 1	Название	Стоимость
1	1	Масляный фильтр	550
2	2	Тормозные колодки	2000
3	3	Свечи зажигания	1000
4	4	Провода высоковольтные	1500
5	5	Датчик температуры двигателя	3000
6	6	Радиатор охлаждения	5000
7	7	Ремень ГРМ	4000
8	8	Аудиосистема	8000

Рисунок 2. Изменение таблицы до и после вызова процедуры

При повторном вызове процедуры стоимость масляного фильтра не поменялась.

2. Для повышения разряда тех мастеров, которые отремонтировали больше 3 автомобилей.

   Создадим вспомогательную таблицу для отслеживания сотрудников и автомобилей, которые уже были учтены при повышении разряда:

```
CREATE TABLE "Promoted_Masters" (
    "Табельный_номер" INTEGER,
    "Госномер" VARCHAR(255),
    PRIMARY KEY ("Табельный_номер", "Госномер")
);
```

   Код процедуры:

```
CREATE OR REPLACE PROCEDURE promote_masters()
LANGUAGE plpgsql
```

```
AS $$
DECLARE
    v_master_id INTEGER;
    v_new_position_id INTEGER;
BEGIN
    FOR v_master_id IN
        SELECT DISTINCT "Сотрудник"."Табельный_номер"
        FROM "Сотрудник"
        JOIN "Состав_заказа" USING("Табельный_номер")
        JOIN "Заказ" USING("Номер_заказа")
        WHERE "Состав_заказа"."Статус_исполнения" = 'Завершён'
        AND NOT EXISTS (
            SELECT 1
            FROM "Promoted_Masters"
            WHERE "Табельный_номер" = v_master_id
            AND "Госномер" = "Заказ"."Госномер"
        )
        GROUP BY "Сотрудник"."Табельный_номер"
        HAVING COUNT(DISTINCT "Заказ"."Госномер") > 3
    LOOP
        SELECT "new_position"."ID_должности" INTO v_new_position_id
        FROM "Сотрудник"
        JOIN "Должность" AS "old_position" USING("ID_должности")
        JOIN "Должность" AS "new_position" USING("Специализация")
        WHERE "Сотрудник"."Табельный_номер" = v_master_id
        AND "new_position"."Разряд" = (
            SELECT "next_rank"
            FROM (
                VALUES
                ('I', 'II'),
```

```
            ('II', 'III'),
            ('III', 'IV'),
            ('IV', 'IV')
        ) AS "Ranks"("current_rank", "next_rank")
        WHERE "current_rank" = "old_position"."Разряд"
    );

    UPDATE "Сотрудник"
    SET "ID_должности" = v_new_position_id
    WHERE "Табельный_номер" = v_master_id;

    INSERT INTO "Promoted_Masters" ("Табельный_номер",
"Госномер")
    SELECT "Сотрудник"."Табельный_номер", "Заказ"."Госномер"
    FROM "Сотрудник"
    JOIN "Состав_заказа" USING("Табельный_номер")
    JOIN "Заказ" USING("Номер_заказа")
    WHERE "Сотрудник"."Табельный_номер" = v_master_id
    AND "Состав_заказа"."Статус_исполнения" = 'Завершён'
    AND NOT EXISTS (
        SELECT 1
        FROM "Promoted_Masters"
        WHERE "Табельный_номер" = v_master_id
        AND "Госномер" = "Заказ"."Госномер"
    );
  END LOOP;
END;
$$;
```

Результат успешного создания процедуры представлен на рисунке 3.

```
automaster$#
automaster$#         UPDATE "Сотрудник"
automaster$#         SET "ID_должности" = v_new_position_id
automaster$#         WHERE "Табельный_номер" = v_master_id;
automaster$#
automaster$#         INSERT INTO "Promoted_Masters" ("Табельный_номер", "Госноме
р")
automaster$#         SELECT "Сотрудник"."Табельный_номер", "Заказ"."Госномер"
automaster$#         FROM "Сотрудник"
automaster$#         JOIN "Состав_заказа" USING("Табельный_номер")
automaster$#         JOIN "Заказ" USING("Номер_заказа")
automaster$#         WHERE "Сотрудник"."Табельный_номер" = v_master_id
automaster$#         AND "Состав_заказа"."Статус_исполнения" = 'Завершён'
automaster$#         AND NOT EXISTS (
automaster$#             SELECT 1
automaster$#             FROM "Promoted_Masters"
automaster$#             WHERE "Табельный_номер" = v_master_id
automaster$#             AND "Госномер" = "Заказ"."Госномер"
automaster$#         );
automaster$#     END LOOP;
automaster$# END;
automaster$# $$;
CREATE PROCEDURE
```

Рисунок 3. Результат создания процедуры 2

В базе данных сотрудник с табельным номером 3 отремонтировал 4 различных автомобиля. Его должность — Механик I (id = 2), она должна поменяться на Механик II (id = 3). Изменения отражены на рисунке 4.

	1 ФИО	ID_автомастерской	Заработная_плата	ID_должности
1	1 Антипов Андрей Викторович	1	70000	1
2	2 Кузьмин Аркадий Андреевич	2	70000	1
3	3 Агеев Дмитрий Александрович	1	0	2
4	4 Агеев Дмитрий Иванович	2	500	2
5	5 Агеев Иван Александрович	1	0	3
6	6 Агеев Иван Иванович	2	1000	3
7	7 Агеев Сергей Александрович	1	0	4
8	8 Агеев Сергей Иванович	2	0	4
9	9 Васин Дмитрий Александрович	1	2500	5
10	10 Васин Дмитрий Иванович	2	0	5
11	11 Васин Иван Александрович	1	0	6
12	12 Васин Иван Иванович	2	0	6
13	13 Васин Сергей Александрович	1	0	7
14	14 Васин Сергей Иванович	2	0	7

	1 ФИО	ID_автомастерской	Заработная_плата	ID_должности
1	1 Антипов Андрей Викторович	1	70000	1
2	2 Кузьмин Аркадий Андреевич	2	70000	1
3	3 Агеев Дмитрий Александрович	1	0	3
4	4 Агеев Дмитрий Иванович	2	500	2
5	5 Агеев Иван Александрович	1	0	3
6	6 Агеев Иван Иванович	2	1000	3
7	7 Агеев Сергей Александрович	1	0	4
8	8 Агеев Сергей Иванович	2	0	4
9	9 Васин Дмитрий Александрович	1	2500	5
10	10 Васин Дмитрий Иванович	2	0	5
11	11 Васин Иван Александрович	1	0	6
12	12 Васин Иван Иванович	2	0	6
13	13 Васин Сергей Александрович	1	0	7
14	14 Васин Сергей Иванович	2	0	7

Рисунок 4. Изменение таблицы до и после вызова процедуры

3. Сколько автомобилей отремонтировал каждый механик за истёкший квартал.

Код функции:

```
CREATE OR REPLACE FUNCTION cars_repaired_last_quarter(p_date DATE)
RETURNS TABLE (
    "Табельный_номер" INTEGER,
    "ФИО" VARCHAR,
    "Количество_ремонтов" BIGINT
)
LANGUAGE plpgsql
AS $$
DECLARE
    v_start_date DATE;
BEGIN
    SELECT
        p_date - interval '3 months' INTO v_start_date;

    RETURN QUERY
    SELECT
        "Сотрудник"."Табельный_номер",
        "Сотрудник"."ФИО",
        COUNT(DISTINCT "Заказ"."Госномер") AS "Количество_ремонтов"
    FROM "Сотрудник"
    LEFT JOIN "Состав_заказа" USING("Табельный_номер")
    LEFT JOIN "Заказ" USING("Номер_заказа")
    WHERE ("Состав_заказа"."Статус_исполнения" = 'Завершён' AND "Состав_заказа"."Дата_исполнения" BETWEEN v_start_date AND p_date)
```

```
    OR "Состав_заказа"."Табельный_номер" IS NULL
GROUP BY "Сотрудник"."Табельный_номер", "Сотрудник"."ФИО";
END;
$$;
```

Результат успешного создания функции представлен на рисунке 5.

```
automaster=# CREATE OR REPLACE FUNCTION cars_repaired_last_quarter(p_date DATE)
automaster-# RETURNS TABLE (
automaster(#     "Табельный_номер" INTEGER,
automaster(#     "ФИО" VARCHAR,
automaster(#     "Количество_ремонтов" BIGINT
automaster(# )
automaster-# LANGUAGE plpgsql
automaster-# AS $$
automaster$# DECLARE
automaster$#     v_start_date DATE;
automaster$# BEGIN
automaster$#     SELECT
automaster$#         p_date - interval '3 months' INTO v_start_date;
automaster$#
automaster$#     RETURN QUERY
automaster$#     SELECT
automaster$#         "Сотрудник"."Табельный_номер",
automaster$#         "Сотрудник"."ФИО",
automaster$#         COUNT(DISTINCT "Заказ"."Госномер") AS "Количество_ремонтов"
automaster$#     FROM "Сотрудник"
automaster$#     LEFT JOIN "Состав_заказа" USING("Табельный_номер")
automaster$#     LEFT JOIN "Заказ" USING("Номер_заказа")
automaster$#     WHERE ("Состав_заказа"."Статус_исполнения" = 'Завершён' AND "Состав_за
каза"."Дата_исполнения" BETWEEN v_start_date AND p_date)
automaster$#         OR "Состав_заказа"."Табельный_номер" IS NULL
automaster$#     GROUP BY "Сотрудник"."Табельный_номер", "Сотрудник"."ФИО";
automaster$# END;
automaster$# $$;
CREATE FUNCTION
```

Рисунок 5. Результат создания функции 3

Посмотреть количество отремонтированных автомобилей за прошедший квартал можно с помощью команды SELECT * FROM cars_repaired_last_quarter(CURRENT_DATE);

Пример вызова функции представлен на рисунке 6.

Табельный_номер	ФИО	Количество_ремонтов
1	Антипов Андрей Викторович	0
2	Кузьмин Аркадий Андреевич	0
3	Агеев Дмитрий Александрович	4
4	Агеев Дмитрий Иванович	1
5	Агеев Иван Александрович	0
6	Агеев Иван Иванович	1
8	Агеев Сергей Иванович	0
9	Васин Дмитрий Александрович	1
10	Васин Дмитрий Иванович	0
11	Васин Иван Александрович	0
13	Васин Сергей Александрович	0
14	Васин Сергей Иванович	0
16	Дорофеев Дмитрий Иванович	0
17	Дорофеев Иван Александрович	0
18	Дорофеев Иван Иванович	0
19	Дорофеев Сергей Александрович	0
20	Дорофеев Сергей Иванович	0
21	Калинин Дмитрий Александрович	0
23	Калинин Иван Александрович	0
24	Калинин Иван Иванович	0
25	Калинин Сергей Александрович	0
26	Калинин Сергей Иванович	0

Рисунок 6. Пример выполнения функции 3

Создание авторского триггера: при изменении очередного статуса исполнения в составе заказа на "Завершён" необходимо добавлять соответствующему мастеру в столбец "Заработная_плата" половину стоимости вида работы.

Триггерная функция:

```
CREATE OR REPLACE FUNCTION update_salary_on_completion()
RETURNS TRIGGER
LANGUAGE plpgsql
AS $$
BEGIN
    IF NEW."Статус_исполнения" = 'Завершён' AND OLD."Статус_исполнения"
!= 'Завершён' THEN
        UPDATE "Сотрудник"
        SET "Заработная_плата" = "Заработная_плата" + (
            SELECT "Стоимость_работы" / 2
            FROM "Вид_работы"
            WHERE "ID_вида_работы" = NEW."ID_вида_работы"
        )
```

```
        WHERE "Табельный_номер" = NEW."Табельный_номер";
    END IF;
    RETURN NEW;
END;
$$;
```

Триггер:

```
CREATE TRIGGER update_salary_on_completion_trigger
AFTER UPDATE OF "Статус_исполнения" ON "Состав_заказа"
FOR EACH ROW
WHEN (NEW."Статус_исполнения" != OLD."Статус_исполнения")
EXECUTE FUNCTION update_salary_on_completion();
```

Результат создания триггера представлен на рисунке 7.

```
automaster=# CREATE OR REPLACE FUNCTION update_salary_on_completion()
automaster-# RETURNS TRIGGER
automaster-# LANGUAGE plpgsql
automaster-# AS $$
automaster$# BEGIN
automaster$#     IF NEW."Статус_исполнения" = 'Завершён' AND OLD."Статус_исполнения" !=
 'Завершён' THEN
automaster$#         UPDATE "Сотрудник"
automaster$#         SET "Заработная_плата" = "Заработная_плата" + (
automaster$#             SELECT "Стоимость_работы" / 2
automaster$#             FROM "Вид_работы"
automaster$#             WHERE "ID_вида_работы" = NEW."ID_вида_работы"
automaster$#         )
automaster$#         WHERE "Табельный_номер" = NEW."Табельный_номер";
automaster$#     END IF;
automaster$#     RETURN NEW;
automaster$# END;
automaster$# $$;
CREATE FUNCTION
automaster=# CREATE TRIGGER update_salary_on_completion_trigger
automaster-# AFTER UPDATE OF "Статус_исполнения" ON "Состав_заказа"
automaster-# FOR EACH ROW
automaster-# WHEN (NEW."Статус_исполнения" != OLD."Статус_исполнения")
automaster-# EXECUTE FUNCTION update_salary_on_completion();
CREATE TRIGGER
```

Рисунок 7. Результат создания триггера

Пример работы триггера показан на рисунках 8 и 9.

	номер_заказа	Табельный_номер	ID_вида_работы	Дата_исполнения	Статус_исполнения
1	1	<null>	3	<null>	Не назначен
2	1	<null>	2	<null>	Не назначен
3	2	22	5	<null>	Назначен
4	2	12	8	<null>	В работе
5	3	9	7	2023-03-17	Завершён
6	4	<null>	7	<null>	Не назначен
7	4	6	2	<null>	Назначен
8	5	15	4	<null>	Назначен
9	5	7	6	<null>	В работе
10	6	4	1	2023-03-17	Завершён
11	6	6	2	2023-03-19	Завершён
12	1	3	1	2023-04-24	Завершён
13	3	3	1	2023-04-24	Завершён
14	5	3	1	2023-04-24	Завершён

	Табельный_н... ^ 1	ФИО	ID_автомастерской	Заработная_плата
1	1	Антипов Андрей Викторович	1	70000
2	2	Кузьмин Аркадий Андреевич	2	70000
3	3	Агеев Дмитрий Александрович	1	0
4	4	Агеев Дмитрий Иванович	2	500
5	5	Агеев Иван Александрович	1	0
6	6	Агеев Иван Иванович	2	1000
7	7	Агеев Сергей Александрович	1	0
8	8	Агеев Сергей Иванович	2	0
9	9	Васин Дмитрий Александрович	1	2500
10	10	Васин Дмитрий Иванович	2	0
11	11	Васин Иван Александрович	1	0
12	12	Васин Иван Иванович	2	0
13	13	Васин Сергей Александрович	1	0
14	14	Васин Сергей Иванович	2	0

Рисунок 8. Содержимое таблиц "Состав_заказа" и "Сотрудник" до изменения данных

	номер_заказа	Табельный_номер	ID_вида_работы	Дата_исполнения	Статус_исполнения
7	5	15	4	<null>	Назначен
8	5	7	6	<null>	В работе
9	6	4	1	2023-03-17	Завершён
10	6	6	2	2023-03-19	Завершён
11	1	3	1	2023-04-24	Завершён
12	3	3	1	2023-04-24	Завершён
13	5	3	1	2023-04-24	Завершён
14	9	3	1	2023-04-24	Завершён
15	2	12	8	2023-04-25	Завершён

	Табельный_н... ^ 1	ФИО	ID_автомастерской	Заработная_плата
1	1	Антипов Андрей Викторович	1	70000
2	2	Кузьмин Аркадий Андреевич	2	70000
3	3	Агеев Дмитрий Александрович	1	0
4	4	Агеев Дмитрий Иванович	2	500
5	5	Агеев Иван Александрович	1	0
6	6	Агеев Иван Иванович	2	1000
7	7	Агеев Сергей Александрович	1	0
8	8	Агеев Сергей Иванович	2	0
9	9	Васин Дмитрий Александрович	1	2500
10	10	Васин Дмитрий Иванович	2	0
11	11	Васин Иван Александрович	1	0
12	12	Васин Иван Иванович	2	2000
13	13	Васин Сергей Александрович	1	0
14	14	Васин Сергей Иванович	2	0

Рисунок 9. Содержимое таблиц "Состав_заказа" и "Сотрудник" после изменения данных

Выводы: в процессе выполнения лабораторной работы были изучены такие темы, как функции и процедуры для базы данных PostgreSQL, а также реализован триггер, который существенно уменьшает количество лишних действий администратора базы данных и возможные ошибки в расчёте заработной платы сотрудников. Было показано, что функции, процедуры и триггеры играют важную роль в проектировании и реализации баз данных, обеспечивая расширяемость, автоматизацию и повышение производительности системы.