

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Отчет

по Лабораторной Работе № 3

по дисциплине «**Проектирование и реализация баз данных**»

Вариант 12

Автор: Мазеин Никита Олегович

Факультет: ФИКТ

Группа: K32402

Преподаватель: Говорова Марина Михайловна

itmo

Санкт-Петербург

2023

1. Цель работы:

овладеть практическими навыками создания и использования процедур, функций и триггеров в базе данных PostgreSQL.

2. Практическое задание:

БД «Прокат автомобилей»

1. Создать процедуры/функции согласно индивидуальному заданию (часть 4).
2. Создать триггер для логирования событий вставки, удаления, редактирования данных в базе данных PostgreSQL (согласно индивидуальному заданию, часть 5). Допустимо создать универсальный триггер или отдельные триггеры на логирование действий.

3. Выполнение:

Задание 4. Создать хранимые процедуры:

- Выполнить списание автомобилей, выпущенных ранее заданного года.

```
1 CREATE OR REPLACE PROCEDURE sp_WriteOffCarsByYear (IN targetYear INT)
2 AS $$
3 BEGIN
4     ALTER TABLE lab."Auto"
5     ADD COLUMN "Status" BOOLEAN DEFAULT true;
6
7     UPDATE lab."Auto"
8     SET "Status" = false
9     WHERE "Release_Year" < targetYear;
10 END;
11 $$ LANGUAGE plpgsql;
12
```

Data Output Messages Notifications

CREATE PROCEDURE

Query returned successfully in 50 msec.

```
1 CALL sp_WriteOffCarsByYear(2015);
2
```

Data Output Messages Notifications

CALL

Query returned successfully in 100 msec.

```
1 select * from lab."Auto"
2 order by "Status";
```

Data Output Messages Notifications

	Auto_Code [PK] integer	Mod_Code integer	Engine_Num character varying (30)	Date_Last_TS timestamp without time zone	Mileage integer	Body_Num character varying (30)	Release_Year integer	reg_plate character varying (9)	Status boolean
1	721930	789012	6UJTT54253	2019-11-15 00:00:00	7409	WP0AB2A74BL123456	2011	M392XM123	false
2	437892	345678	32VFDK6749	2023-06-01 00:00:00	8053	WBA3D5C53EKX98023	2014	MH523E83	false
3	693246	890123	K6U2315266	2023-06-01 00:00:00	4789	5YJSA1DN8CFP12345	2013	EX437X77	false
4	539481	901234	KAA4427606	2023-06-01 00:00:00	29750	WA1CVAFP5AA098765	2011	Y007YX36	false
5	972841	456789	N97F30E433	2023-06-01 00:00:00	8935	WDDKK5GF5BF123456	2014	A717AP26	false
6	205635	890123	25GFBC2891	2021-05-21 00:00:00	1392	5YJSA1E11HF185161	2019	H0438054	true

- Выдачи автомобиля и расчета стоимости с учетом скидки постоянным клиентам.

```

1 CREATE OR REPLACE PROCEDURE sp_IssueCarToCl (IN cl_Code INT, IN stf_Code INT, IN auto_Code INT, IN rent_h INT)
2 AS $$
3 DECLARE
4     discount DECIMAL(10, 2);
5     rentPrice INT;
6     finalPrice INT;
7 BEGIN
8     IF EXISTS (
9         SELECT 1
10        FROM lab."Bonus_Card"
11        WHERE "Cl_Code" = cl_Code
12    ) THEN
13        discount := 0.05;
14    ELSE
15        discount := 0;
16    END IF;
17
18    SELECT
19        CASE
20            WHEN rent_h < 24 THEN "Price_One_H" * rent_h
21            ELSE "Price_Long_Inter" * (rent_h / 24)
22        END INTO rentPrice
23    FROM lab."Price"
24    WHERE "Mod_Code" = (SELECT "Mod_Code" FROM lab."Auto" WHERE "Auto_Code" = auto_Code)
25        AND "DT_Inter_End" IS NULL;
26
27    finalPrice := CAST(rentPrice * (1 - discount) AS INT);
28
29    INSERT INTO lab."Contract" ("Contr_Code", "Act_Transf_Client", "Act_Transf_Company", "Rent_Price", "DT_Contrac
30    VALUES ((SELECT COALESCE(MAX("Contr_Code"), 0) + 1 FROM lab."Contract"), (SELECT COALESCE(MAX("Act_Transf_C
31
32 END;
33 $$ LANGUAGE plpgsql;

```

Data Output Messages Notifications

CREATE PROCEDURE

Query returned successfully in 73 msec.

```

INSERT INTO lab."Contract" ("Contr_Code", "Act_Transf_Client", "Act_Transf_Company", "Rent_Price", "DT_Contract",
"DT_Car_Transf_To_Cl", "Factual_DT_Ret", "Late_Fee", "Ret_Mark", "Cl_Code", "Stf_Code", "Auto_Code", "rent_time")

VALUES ((SELECT COALESCE(MAX("Contr_Code"), 0) + 1 FROM lab."Contract"), (SELECT COALESCE(MAX("Act_Transf_Client"), 0) + 1 FROM
lab."Contract"), NULL, finalPrice, CURRENT_TIMESTAMP, NULL, NULL, NULL, false, cl_Code, stf_Code, auto_Code, rent_h);

```

(То, что не вошло)

Арендуем машины с одинаковой ценой сначала через клиента с картой постоянного клиента, потом через клиента без карты.

```

1 CALL sp_IssueCarToCl(12345, 123123, 539481, 10);

```

Data Output Messages Notifications

CALL

Query returned successfully in 75 msec.

```

1 CALL sp_IssueCarToCl(23456, 123123, 776363, 10);

```

Data Output Messages Notifications

CALL

Query returned successfully in 68 msec.

34	100035	128167	[null]	1000	2023-06-08 07:45:03.1	[null]	[null]	[null]	false	12345	123123	539481	10
35	100036	128168	[null]	950	2023-06-08 07:46:01	[null]	[null]	[null]	false	23456	123123	776363	10

1000

950

Скидка применилась!

- Для вычисления количества автомобилей заданной марки.

```
1 CREATE OR REPLACE PROCEDURE sp_get_car_count_by_model(IN mod_Code Integer, OUT carCount Integer)
2 AS $$
3 BEGIN
4     SELECT COUNT(*) INTO carCount
5     FROM lab."Auto" a
6     JOIN lab."Model" m USING ("Mod_Code")
7     WHERE m."Mod_Code" = mod_Code;
8 END;
9 $$ LANGUAGE plpgsql;
10
11 DO $$
12 DECLARE
13     carCount Integer;
14 BEGIN
15     CALL sp_get_car_count_by_model(345678, carCount);
16     RAISE NOTICE 'Car Count Result: %', carCount;
17 END $$;
```

Data Output Messages Notifications

NOTICE: Car Count Result: 7
DO

Query returned successfully in 38 msec.

Задание 5. Создать триггеры:

Триггер на начисление штрафа после возврата автомобиля с просрочкой

```
1 CREATE OR REPLACE FUNCTION calculate_late_fee()
2 RETURNS TRIGGER AS $$
3 DECLARE
4     late_time INT;
5     late_fee REAL;
6 BEGIN
7     IF NEW."Factual_DT_Ret" IS NOT NULL AND OLD."Factual_DT_Ret" IS NULL THEN
8         SELECT EXTRACT(EPOCH FROM (NEW."Factual_DT_Ret" - (OLD."DT_Car_Transf_To_Cl" + INTERVAL '1 hour' * OLD."rent_time"))) / 3600
9         INTO late_time
10        FROM lab."Contract";
11
12        SELECT (late_time * "Price_One_H")
13        INTO late_fee
14        FROM lab."Price" p
15        JOIN lab."Auto" a ON p."Mod_Code" = a."Mod_Code"
16        WHERE p."DT_Inter_End" IS NULL
17              AND a."Auto_Code" = OLD."Auto_Code";
18
19        UPDATE lab."Contract"
20        SET "Late_Fee" = late_fee,
21            "Ret_Mark" = true
22        WHERE "Contr_Code" = NEW."Contr_Code";
23    END IF;
24
25    RETURN NEW;
26 END;
27 $$ LANGUAGE plpgsql;
28
29 CREATE TRIGGER late_fee_trigger
30 AFTER UPDATE ON lab."Contract"
31 FOR EACH ROW
32 EXECUTE FUNCTION calculate_late_fee();
33
```

Data Output Messages Notifications

CREATE TRIGGER

Query returned successfully in 74 msec.

34	100035	128167	[null]	1000	2023-06-08 07:45:03.1	2023-06-08 8:00:00	[null]	[null]	false	12345	123123	539481	10
35	100036	128168	[null]	950	2023-06-08 07:46:01	2023-06-08 8:10:00	[null]	[null]	false	23456	123123	776363	10

Обновим строки и проверим работу триггера:

34	100035	128167	[null]	1000	2023-06-08 07:45:03.1	2023-06-08 08:00:00	2023-06-08 15:30:00	[null]	true	12345	123123	539481	10
35	100036	128168	[null]	950	2023-06-08 07:46:01	2023-06-08 08:10:00	2023-06-08 19:30:00	100	true	23456	123123	776363	10

Триггер сработал!

4. Выводы:

По итогам выполнения лабораторной работы я ознакомился с понятиями процедур, функций и триггеров. Мною были созданы и проверены на работоспособность несколько процедур, также триггер, который значительно упростил работу с базой данных, автоматизировав вычисление штрафа за просрочку возврата автомобиля.