БД «Прокат автомобилей»

1. Создание процедур/функций.

2. Создание триггера для логирования событий вставки, удаления, редактирования данных в базе данных PostgreSQL.

Создаём хранимые процедуры:

- Выполнить списание автомобилей, выпущенных ранее заданного года.

```
CREATE OR REPLACE PROCEDURE sp_WriteOffCarsByYear (IN targetYear INT)
AS $$
BEGIN
    ALTER TABLE lab."Auto"
    ADD COLUMN "Status" BOOLEAN DEFAULT true;

    UPDATE lab."Auto"
    SET "Status" = false
    WHERE "Release_Year" < targetYear;
END;
$$ LANGUAGE plpgsql;
```

Data Output | Messages | Notifications

```
CREATE PROCEDURE

Query returned successfully in 50 msec.
```

```
CALL sp_WriteOffCarsByYear(2015);
```

Data Output | Messages | Notifications

```
CALL

Query returned successfully in 100 msec.
```

```
select * from lab."Auto"
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	У007УХ36	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	HO438O54	true

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

```
CREATE OR REPLACE PROCEDURE sp_IssueCarToCl (IN cl_Code INT, IN stf_Code INT, IN auto_Code INT, IN rent_h INT)
AS $$
DECLARE
    discount DECIMAL(10, 2);
    rentPrice INT;
    finalPrice INT;
BEGIN
    IF EXISTS (
        SELECT 1
        FROM lab."Bonus_Card"
        WHERE "Cl_Code" = cl_Code
    ) THEN
        discount := 0.05;
    ELSE
        discount := 0;
    END IF;

    SELECT
        CASE
            WHEN rent_h < 24 THEN "Price_One_H" * rent_h
            ELSE "Price_Long_Inter" * (rent_h / 24)
        END INTO rentPrice
    FROM lab."Price"
    WHERE "Mod_Code" = (SELECT "Mod_Code" FROM lab."Auto" WHERE "Auto_Code" = auto_Code)
        AND "DT_Inter_End" IS NULL;

    finalPrice := CAST(rentPrice * (1 - discount) AS INT);

    INSERT INTO lab."Contract" ("Contr_Code", "Act_Transf_Client", "Act_Transf_Company", "Rent_Price", "DT_Contr
    VALUES ((SELECT COALESCE(MAX("Contr_Code"), 0) + 1 FROM lab."Contract"), (SELECT COALESCE(MAX("Act_Transf_C

END;
$$ 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 <u>сначала через клиента с картой</u> постоянного клиента, потом через клиента без карты.

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

Data Output | Messages | Notifications

```
CALL

Query returned successfully in 75 msec.
```

```
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.

```
CREATE OR REPLACE PROCEDURE sp_get_car_count_by_model(IN mod_Code Integer, OUT carCount Integer)
AS $$
BEGIN
    SELECT COUNT(*) INTO carCount
    FROM lab."Auto" a
    JOIN lab."Model" m USING ("Mod_Code")
    WHERE m."Mod_Code" = mod_Code;
END;
$$ LANGUAGE plpgsql;

DO $$
DECLARE
    carCount Integer;
BEGIN
     CALL sp_get_car_count_by_model(345678, carCount);
     RAISE NOTICE 'Car Count Result: %', carCount;
END $$;
```

Data Output | Messages | Notifications

```
NOTICE:  Car Count Result: 7
DO

Query returned successfully in 38 msec.
```

Создаём триггеры:

Триггер на начисление штрафа после возврата автомобиля с просрочкой

```
CREATE OR REPLACE FUNCTION calculate_late_fee()
RETURNS TRIGGER AS $$
DECLARE
  late_time INT;
  late_fee REAL;
BEGIN
  IF NEW."Factual_DT_Ret" IS NOT NULL AND OLD."Factual_DT_Ret" IS NULL THEN
    SELECT EXTRACT(EPOCH FROM (NEW."Factual_DT_Ret" - (OLD."DT_Car_Transf_To_Cl" + INTERVAL '1 hour' * OLD."rent_time"))) / 3600
    INTO late_time
    FROM lab."Contract";

    SELECT (late_time * "Price_One_H")
    INTO late_fee
    FROM lab."Price" p
    JOIN lab."Auto" a ON p."Mod_Code" = a."Mod_Code"
    WHERE p."DT_Inter_End" IS NULL
      AND a."Auto_Code" = OLD."Auto_Code";

    UPDATE lab."Contract"
    SET "Late_Fee" = late_fee,
        "Ret_Mark" = true
    WHERE "Contr_Code" = NEW."Contr_Code";
  END IF;

  RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER late_fee_trigger
AFTER UPDATE ON lab."Contract"
FOR EACH ROW
EXECUTE FUNCTION calculate_late_fee();
```

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!

Выводы:

Мы ознакомились с понятиями процедур, функций и триггеров. Нами были созданы и проверены на работоспособность несколько процедур, также триггер, который значительно упростил работу с базой данных, автоматизировав вычисление штрафа за просрочку возврата автомобиля.