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»**

Автор: Игнатьев А.А.

Факультет: ИКТ

Группа: К3240

Преподаватель: Говорова М.М.

Санкт-Петербург 2023

Оглавление

Цель работы:

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

Вариант 2 (max - 8 баллов)

1. Создать процедуры/функции согласно индивидуальному заданию (часть 4).

2.1. Модифицировать триггер (триггерную функцию) на проверку корректности входа и выхода сотрудника (см. Практическое задание 1 Лабораторного практикума (Приложение)) с максимальным учетом «узких» мест некорректных данных по входу и выходу).

2.2. Создать авторский триггер по варианту индивидуального задания.

Указание. Работа выполняется в консоли SQL Shell (psql).

Вариант 19. БД «Издательство компьютерной литературы»

Описание предметной области:

Описание предметной области: Издательство занимается выпуском литературы по различным областям ИТ. Покупатели (юридические лица) приобретают книги на базе издательства.

Когда на базе заканчиваются книги, издается дополнительный тираж.

В каждом заказе заказчик может заказать разную литературу. Для покупки заключается договор, который сопровождает менеджер издательства. По каждому проекту составляется договор с Заказчиком (в 2-х экземплярах для каждой стороны). По каждому договору оформляется два счета – на предоплату и остаток. После выполнения проекта подписывается Акт выполненных работ (в 2-х экземплярах для каждой стороны).

Каждое издание относится к определенной области ИТ, имеет тип (учебник, учебное пособие и т.п.), номер издание (если есть), может иметь одного или нескольких авторов, выпускаться под редакцией одного или нескольких авторов и т.п. При формировании списка авторов или списка “под редакцией” важен порядок авторов.

На каждое издание составляется Техническое задание, в котором могут участвовать несколько редакторов, один из которых является главным редактором. На каждую книгу может быть несколько ТЗ, в зависимости от переплета, типа бумаги, наличия иллюстраций и т.д.

БД должна содержать следующий минимальный набор сведений: Фамилия автора. Имя автора. Отчество автора. Код автора. E-mail автора. Код ISBN. Название книги. Количество страниц. Наличие иллюстраций. Код категории книги. Категория книги. Количество страниц. Год начала издания. Розничная цена книги. Тираж. Дата тиража.

Количество экземпляров на базе издательства. Код заказчика. Фамилия заказчика. Имя заказчика. Отчество заказчика. Адрес заказчика. Телефон заказчика. Код заказа. Дата заказа. Срок заказа. Количество экземпляров книги в заказе. Статус заказа.

Выполнение

Создать процедуры

1 - Для снижения цен на книги, которые находятся на базе в количестве, превышающем 1000 штук.

```
CREATE OR REPLACE PROCEDURE decrease_price(amount integer)
language plpgsql
AS $$
BEGIN
    UPDATE "Circulation" set
    price = price - amount where "left_count" > 1000;
    COMMIT;
END;
$$;
```

CREATE OR REPLACE PROCEDURE decrease_price(amount integer)
language plpgsql
AS $$
BEGIN
UPDATE "Circulation" set
price = price - amount where "left_count" > 1000;
COMMIT;
END;
$$;

```
Book_edition=# select * from "Circulation";
 circulation_code |    date    | made_count | left_count |   isbn_code   | price
------------------+------------+------------+------------+---------------+-------
                1 | 2023-10-10 |      10000 |       1200 | 1234567891230 |   100
                2 | 2023-11-10 |     200000 |       1500 | 0123456789123 |   120
                3 | 2023-10-11 |        100 |         10 | 0123456789123 |   130
(3 ёЄЁюъш)


Book_edition=# call decrease_price(100);
CALL
Book_edition=# select * from "Circulation";
 circulation_code |    date    | made_count | left_count |   isbn_code   | price
------------------+------------+------------+------------+---------------+-------
                3 | 2023-10-11 |        100 |         10 | 0123456789123 |   130
                1 | 2023-10-10 |      10000 |       1200 | 1234567891230 |     0
                2 | 2023-11-10 |     200000 |       1500 | 0123456789123 |    20
(3 ёЄЁюъш)


Book_edition=#
```

2 - Для ввода новой книги.

```
CREATE OR REPLACE PROCEDURE add_new_book(
    BookName character varying(255),
    WriteYear integer,
    PublishYear integer,
    CategoryCode integer,
    AuthorCode integer
)
language plpgsql
AS $$
DECLARE
    BookNumber integer;
BEGIN
    INSERT into "Book" (book_number, name, write_year, publish_year) values (DEFAULT, BookName, WriteYear, PublishYear);
    BookNumber := (select "book_number" from "Book" where "name" = BookName);
    INSERT into "Book_category" (book_category_id, book_number, category_code, priority) values (DEFAULT, BookNumber, CategoryCode, 1);
    INSERT into "Book_author" (book_author_id, book_number, author_code, author_position) values (DEFAULT, BookNumber, AuthorCode, 1);
    COMMIT;
END;
$$;
```

CREATE OR REPLACE PROCEDURE add_new_book(
    BookName character varying(255),
    WriteYear integer,
    PublishYear integer,
    CategoryCode integer,
    AuthorCode integer
)
language plpgsql
AS $$
DECLARE
    BookNumber integer;
BEGIN

INSERT into "Book" (book_number, name, write_year, publish_year) values (DEFAULT, BookName, WriteYear, PublishYear);

BookNumber := (select "book_number" from "Book" where "name" = BookName);

INSERT into "Book_category" (book_category_id, book_number, category_code, priority) values (DEFAULT, BookNumber, CategoryCode, 1);

INSERT into "Book_author" (book_author_id, book_number, author_code, author_position) values (DEFAULT, BookNumber, AuthorCode, 1);

COMMIT;

END;

$$;

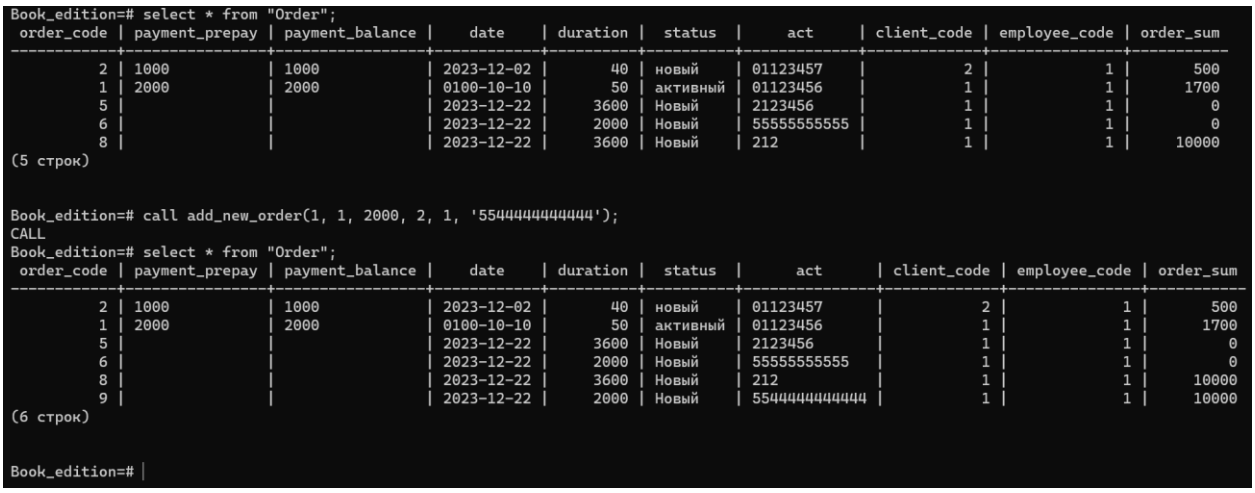
```
Book_edition=# select * from "Book";
 book_number |       name         | write_year | publish_year
-------------+--------------------+------------+--------------
           1 | Книга о волке2     |       1000 |         2023
           2 | проектирование бд  |       1002 |         2023
           4 | Тест               |       2000 |         2020
(3 строки)


Book_edition=# call add_new_book('Тест2', 2000, 2020, 1, 1);
CALL
Book_edition=# select * from "Book";
 book_number |       name         | write_year | publish_year
-------------+--------------------+------------+--------------
           1 | Книга о волке2     |       1000 |         2023
           2 | проектирование бд  |       1002 |         2023
           4 | Тест               |       2000 |         2020
           5 | Тест2              |       2000 |         2020
(4 строки)


Book_edition=#
```

3 - Для ввода нового заказа.

```
CREATE OR REPLACE PROCEDURE add_new_order(
    ClientCode integer,
    ManagerCode integer,
    Duration integer,
    BooksCount integer,
    CirculationCode integer,
    ActOrder character varying (50)
)
language plpgsql
AS $$
DECLARE
    OrderCode integer;
    BookPrice integer;
    CurrentDate date;
BEGIN
    CurrentDate := CURRENT_DATE;
    BookPrice := (select "price" from "Circulation" where "circulation_code"= CirculationCode);
    INSERT into "Order" (order_code, date, duration, status, client_code, employee_code, order_sum, act)
    values (DEFAULT, CurrentDate, Duration, 'Новый', ClientCode, ManagerCode, BookPrice * BooksCount, ActOrder);
    OrderCode := (select "order_code" from "Order" where "act" = ActOrder);
    INSERT into "Order_creation" (order_creation_id, order_code, circulation_code, count) values (DEFAULT, OrderCode, CirculationCode, BooksCount);
    COMMIT;
END;
$$;
```

CREATE OR REPLACE PROCEDURE add_new_order(

```
    ClientCode integer,
    ManagerCode integer,
    Duration integer,
    BooksCount integer,
    CirculationCode integer,
    ActOrder character varying (50)
)
language plpgsql
AS $$
DECLARE
    OrderCode integer;
    BookPrice integer;
    CurrentDate date;
BEGIN
    CurrentDate := CURRENT_DATE;
    BookPrice := (select "price" from "Circulation" where "circulation_code"= CirculationCode);
    INSERT into "Order" (order_code, date, duration, status, client_code, employee_code, order_sum, act)
    values (DEFAULT, CurrentDate, Duration, 'Новый', ClientCode, ManagerCode, BookPrice * BooksCount, ActOrder);
    OrderCode := (select "order_code" from "Order" where "act" = ActOrder);
    INSERT into "Order_creation" (order_creation_id, order_code, circulation_code, count) values (DEFAULT, OrderCode, CirculationCode, BooksCount);
    COMMIT;
END;
$$;
```



```
Book_edition=# select * from "Order";
 order_code | payment_prepay | payment_balance |    date    | duration |  status  |     act     | client_code | employee_code | order_sum
------------+----------------+-----------------+------------+----------+----------+-------------+-------------+---------------+-----------
          2 | 1000           | 1000            | 2023-12-02 |       40 | новый    | 01123457    |           2 |             1 |       500
          1 | 2000           | 2000            | 0100-10-10 |       50 | активный | 01123456    |           1 |             1 |      1700
          5 |                |                 | 2023-12-22 |     3600 | Новый    | 2123456     |           1 |             1 |         0
          6 |                |                 | 2023-12-22 |     2000 | Новый    | 55555555555 |           1 |             1 |         0
          8 |                |                 | 2023-12-22 |     3600 | Новый    | 212         |           1 |             1 |     10000
(5 строк)


Book_edition=# call add_new_order(1, 1, 2000, 2, 1, '554444444444');
CALL
Book_edition=# select * from "Order";
 order_code | payment_prepay | payment_balance |    date    | duration |  status  |     act      | client_code | employee_code | order_sum
------------+----------------+-----------------+------------+----------+----------+--------------+-------------+---------------+-----------
          2 | 1000           | 1000            | 2023-12-02 |       40 | новый    | 01123457     |           2 |             1 |       500
          1 | 2000           | 2000            | 0100-10-10 |       50 | активный | 01123456     |           1 |             1 |      1700
          5 |                |                 | 2023-12-22 |     3600 | Новый    | 2123456      |           1 |             1 |         0
          6 |                |                 | 2023-12-22 |     2000 | Новый    | 55555555555  |           1 |             1 |         0
          8 |                |                 | 2023-12-22 |     3600 | Новый    | 212          |           1 |             1 |     10000
          9 |                |                 | 2023-12-22 |     2000 | Новый    | 554444444444 |           1 |             1 |     10000
(6 строк)


Book_edition=#
```

Создать триггер

Уменьшение количества книг на складе при добавлении заказа

```
CREATE OR REPLACE FUNCTION UpdateCount()
RETURNS TRIGGER AS $$
BEGIN
    UPDATE "Circulation" set "left_count" = "left_count" - NEW.count
    WHERE "circulation_code" = NEW.circulation_code;
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

Create trigger UpdateCountTrigger
AFTER INSERT ON "Order_creation"
FOR EACH ROW
EXECUTE FUNCTION UpdateCount();
```

CREATE OR REPLACE FUNCTION UpdateCount()
RETURNS TRIGGER AS $$
BEGIN
    UPDATE "Circulation" set "left_count" = "left_count" - NEW.count
    WHERE "circulation_code" = NEW.circulation_code;
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

Create trigger UpdateCountTrigger
AFTER INSERT ON "Order_creation"
FOR EACH ROW

EXECUTE FUNCTION UpdateCount();

```
Book_edition=# select * from "Circulation";
 circulation_code |    date    | made_count | left_count |   isbn_code   | price
------------------+------------+------------+------------+---------------+-------
                3 | 2023-10-11 |        100 |         10 | 0123456789123 |   130
                2 | 2023-11-10 |     200000 |       1500 | 0123456789123 |    20
                1 | 2023-10-10 |      10000 |       1198 | 1234567891230 |  5000
(3 строки)


Book_edition=# call add_new_order(1, 1, 2000, 2, 1, '5544444444');
CALL
Book_edition=# select * from "Circulation";
 circulation_code |    date    | made_count | left_count |   isbn_code   | price
------------------+------------+------------+------------+---------------+-------
                3 | 2023-10-11 |        100 |         10 | 0123456789123 |   130
                2 | 2023-11-10 |     200000 |       1500 | 0123456789123 |    20
                1 | 2023-10-10 |      10000 |       1196 | 1234567891230 |  5000
(3 строки)


Book_edition=#
```

Вывод

В ходе лабораторной работы были написаны процедуры, функции, триггеры для PostgresSQL