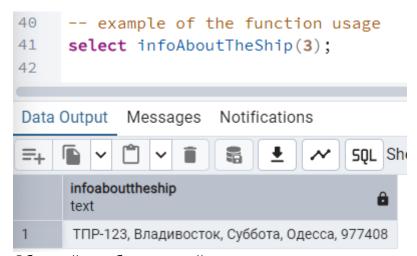
Отчёт по лабораторной работе №4 по управлению данными

Гайфиев Иван Алексеевич студент 23КНТ-1 НИУ ВШЭ НН

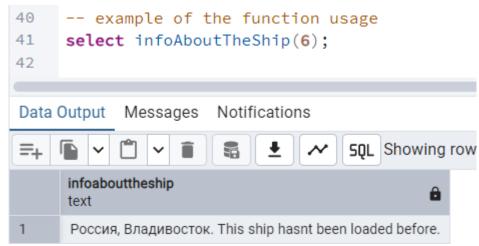
Формулировки заданий и решения:

1. Реализовать хранимую процедуру, возвращающую текстовую строку, содержащую информацию о судне (название, порт приписки, дата, место и стоимость последней погрузки). Обработать ситуацию, когда судно новое, и еще ни разу не грузилось.

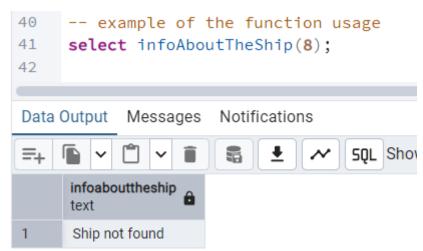
```
1 v create or replace function infoAboutTheShip(my_ship_id int)
 2 returns text as $$
 3
    declare
        result_message text; -- text string with info about the ship
        number_of_loadings int; -- how many times this ship has been loaded
 5
       last_load_id int; -- id of the last loading
 7
        ship name varchar(30);
       h_port varchar(30);
8
9
       last_load_date varchar(30);
10
        last_load_port varchar(30);
11
        last_load_price int;
12 v begin
from loading join ship on ship.id = loading.ship_id where my_ship_id = ship.id;
14
15 🗸
        select max(statement_number) into last_load_id -- id of the last loading the exact ship had
           from loading where ship_id = my_ship_id;
16
17
18 🗸
        if my_ship_id not in (select id from ship) then -- situation when ship with this id doesn't exist
           raise notice 'Ship with this id doesnt exist!';
19
20
            return 'Ship not found';
21
        end if:
22
23 🗸
        if my_ship_id in (select id from ship) and number_of_loadings = 0 then -- this ship is new and hasn't been loaded before
24
            select name, home_port into ship_name, h_port
25
            from ship where id = my_ship_id;
26
            result_message = ship_name || ', ' || h_port || '. This ship hasnt been loaded before.';
27 v
28
            select name, home_port into ship_name, h_port
29
            from ship where id = my_ship_id;
           select loading.date, loading_place.port, loading.price
30 🗸
31
           into last_load_date, last_load_port, last_load_price
32
            from loading join loading_place on loading_place_id = loading_place.id
33
            where loading.statement_number = last_load_id;
            result_message = ship_name || ', ' || h_port || ', ' || last_load_date || ', '
34 🗸
35
            || last_load_port || ', ' || last_load_price;
36
        end if;
37
        return result_message;
39 $$ language plpgsql;
     -- example of the function usage
41 select infoAboutTheShip(3);
```



Обычный корабль, который существует и имеет опыт погрузки



Новый корабль, который не бывал в порту и не перевозил груз



Корабля с таким id не существует

2. Добавить таблицу, содержащую списки возможных грузов для каждого места погрузки. При вводе погрузки проверять, может ли присутствовать данный груз в указанном месте погрузки.

```
1 v create table possible_cargo as -- this table contains info about the place where various cargos are located
 2 select loading_place_id, cargo_id from loading
 3 select * from possible_cargo order by loading_place_id;
 5 v create or replace function exactCargoInLoadingPlaceAvailability()
    returns trigger as $$
    declare
8
        text mes text:
9 v begin
10
         -- check if inputed parameters are correct
        if new.loading_place_id in (select p_c.loading_place_id from possible_cargo as p_c) and new.cargo_id in
11
         (select p_c.cargo_id from possible_cargo as p_c) then
12
13
            -- check if this exact cargo is already in this exact loading place
14
            if exists (select 1 from possible_cargo where new.loading_place_id = loading_place_id and new.cargo_id = cargo_id) then
15
                raise notice 'This cargo is already in the loading place';
16
17 🗸
18
                raise notice 'This cargo is not in the loading place yet';
                return new;
19
20
            end if:
21 🗸
22
            text_mes = 'Cargo or loading_place with these id dont exist';
23
            raise exception '%',text_mes;-- неправильный ввод этих двух параметров
24
25 end:
$$ language plpgsql;
27
28 v create or replace trigger cargoAvailabilityTrigger
29 before insert or update on loading
30
    for each row execute function exactCargoInLoadingPlaceAvailability();
31
   -- check if the trigger works and provides as the information on cargo availability
32 • insert into loading (statement_number, date, ship_id, loading_place_id, cargo_id, number, price)
33 values (1, 'Вторник', 001, 004, 002, 124, 3900000);
```

```
values (1, 'Вторник', 001, 004, 002, 124, 3900000);

Data Output Messages Notifications

ЗАМЕЧАНИЕ: This cargo is already in the loading place
INSERT 0 1
```

Груз уже в месте погрузки означает, что груз по id 002 уже содержится в месте погрузки 004, но это не означает запрет на погрузку, поэтому вставка всё равно разрешается, просто выводится уведомление о том, что такой груз уже есть.

```
28 v insert into loading (statement_number, date, ship_id, loading_place_id, cargo_id, number, price)
29 values (2, 'Вторник', 001, 002, 004, 124, 3900000);

Data Output Messages Notifications

ЗАМЕЧАНИЕ: This cargo is not in the loading place yet
INSERT 0 1
```

Обратная ситуация – груза 004 нету в порту 002, поэтому выводится соответствующее оповещение и происходит вставка

```
28 vinsert into loading (statement_number, date, ship_id, loading_place_id, carg values (2, 'Вторник', 001, 009, 004, 124, 3900000);

Data Output Messages Notifications

ERROR: Cargo or loading_place with these id dont exist

CONTEXT: функция PL/pgSQL exactcargoinloadingplaceavailability(), строка 18, оператор RAISE

ОШИБКА: Cargo or loading_place with these id dont exist

SQL state: P0001
```

Порта 009 не существует, поэтому выводим ошибку.



Так выглядит таблица погрузок после запущенного кода с примерами

3. Реализовать триггер такой, что при вводе строки в таблице погрузок, если сумма не указана, то она вычисляется

```
1 ➤ create or replace function countingSumIfNecessary()
     returns trigger as $$
 3
     declare
4
         loading_sum bigint;
5 v begin
         loading_sum = new.number * (select cargo.price from cargo where cargo.id = new.cargo_id);
 6
         if new.price is null then
 7 🗸
8
             new.price = loading_sum;
9
             return new;
10 🗸
         else
11
             return new;
12
         end if;
13
    end:
    $$ language plpgsql;
14
15
16 v create or replace trigger countingSumTrigger
17
     before insert or update on loading
     for each row execute function countingSumIfNecessary();
18
     -- check if the trigger works and provides as the information on cargo availability
19
20 v insert into loading (statement_number, date, ship_id, loading_place_id, cargo_id, number)
   values (1, 'Вторник', 001, 004, 002, 124)
21
     insert into loading (statement_number, date, ship_id, loading_place_id, cargo_id, number, price)
22
23 values (2, 'Вторник', 001, 004, 002, 124, 2352353);
             19 -- check if the trigger works and provides as the information on cargo availability
             20 v insert into loading (statement_number, date, ship_id, loading_place_id, cargo_id, number)
             21 values (1, 'Вторник', 001, 004, 002, 124)
             Data Output Messages Notifications
             ЗАМЕЧАНИЕ: This cargo is already in the loading place
             INSERT 0 1
                                                    ship_id integer
                   statement_number
                                                             loading_place_id
                                   character varying (50)
                                                                           integer
                                                                                    integer
                                                            integer
                                1 Вторник
                                                                                  2
                                                                                        124
                                                                                                     9920000
              2
                             70204 Понедельник
                                                                         5
                                                                                  2
                                                                                        100
                                                                                                      6516050
                             70205 Понедельник
                                                                                                     977407.5
```

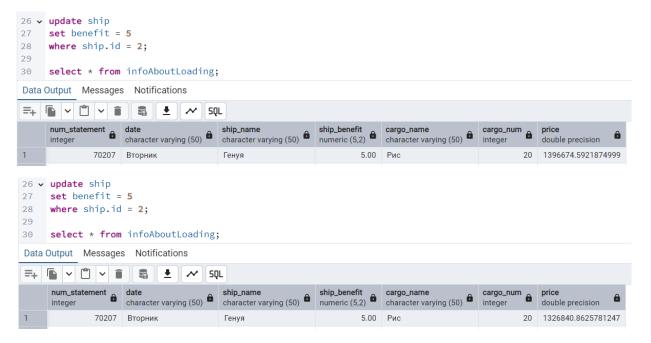
Цена не указана, но вставка всё равно произошла и цена автоматически вычислилась

22	, , , , , , , , , , , , , , , , , , , ,							
Data	Output Messages N	Notifications						
3AMEYAHUE: This cargo is already in the loading place INSERT 0 1								
	statement_number [PK] integer	date character varying (50)	ship_id integer	loading_place_id integer	cargo_id integer	number integer	price double precision	
1	1	Вторник	1	4	2	124	9920000	
2	2	Вторник	1	4	2	124	2352353	
3	70204	Понедельник	1	5	2	100	6516050	

Указывать цену тоже можно

4. Создать представление (view), содержащее поля: номер ведомости, дата, название судна, льгота, название груза, количество, стоимость. Обеспечить возможность изменения предоставленной льготы. При этом должна быть пересчитана стоимость.

```
1 v create view infoAboutLoading(num_statement, date, ship_name, ship_benefit, cargo_name, cargo_num, price) as
     (select loading.statement_number, loading.date, ship.name, ship.benefit, cargo.name, loading.number, loading.price
     from loading join ship on ship.id = loading.ship_id join cargo on cargo.id = loading.cargo_id)
    create or replace function changeBenefitAndRecountprice ()
 6
    returns trigger as $$
 7
    declare
8
9
10
        if new.benefit is null or new.benefit < 0 or new.benefit > 100 then
11
            raise exception 'Value of benefit must be between 1 and 100';
12 🗸
            update loading
13
            set price = (1 - new.benefit * 0.01) * price
14
15
            from ship
16
            where ship_id = new.id;
17
            return new;
18
        end if;
19
    end:
20
    $$ language plpgsql;
21
22 v create or replace trigger changeBenefitTrigger
23
    before update on ship
24
    for each row execute function changeBenefitAndRecountprice ();
25
26 v update ship
27
    set benefit = 5
    where ship.id = 2;
28
29
    select * from infoAboutLoading;
30
```



Id 2 соответствует кораблю Генуя. После запуска кода на 5% снижается стоимость погрузки у всех записей по указанному id.

```
26 v update ship
27 set benefit = 500
28 where ship.id = 2;
29
30 select * from infoAboutLoading;

Data Output Messages Notifications

ERROR: Value of benefit must be between 1 and 100

CONTEXT: функция PL/pgSQL changebenefitandrecountprice(), строка 6, оператор RAISE

ОШИБКА: Value of benefit must be between 1 and 100

SQL state: P0001
```

Проверка на неправильно введённое значение