

Державний університет «Одеська Політехніка»
Інститут комп'ютерних систем
Кафедра інформаційних систем

Лабораторна робота №6
з дисципліни «Організація баз даних та знань»
Тема: «**Маніпулювання даними. Тригери**»
Варіант: 11

Виконав:
студент групи АІ-205
Светашов Д.В.

Перевірили:
Глава М.Г
Дрозд М.О.

Одеса 2021

Завдання:

Запишіть SQL-запити для маніпулювання даними з таблиць, що створені у лабораторній роботі 1.

В роботі обов'язково відобразити:

- 1) тригер на операцію модифікації даних INSERT;
- 2) тригер на операцію модифікації даних DELETE;
- 3) тригер на операцію модифікації даних UPDATE;
- 4) користувальницьку функцію (збережену процедуру), яка викликається оператором select.

Хід роботи

1) Тригер на INSERT;

Коли ми намагаємось додати до таблиці “Власник” (owner) нового власника, то перевіряємо, щоб його номер телефону був відсутнім у таблиці. В моїй базі даних атрибут номер телефону (telefon) має обмеження NOT NULL, тому при додаванні власника ми обов'язково повинні вказувати його номер телефону.






SQL код створення такого триггера:

```
CREATE OR REPLACE FUNCTION new_owner()
RETURNS trigger
AS $$
BEGIN
IF (SELECT telefon FROM owner WHERE telefon=new.telefon) IS NOT NULL
THEN
RAISE EXCEPTION 'Помилка: Власник з таким номером телефону вже існує';
END IF;
RETURN new;
END;$$

LANGUAGE 'plpgsql';
```

```
CREATE OR REPLACE TRIGGER new_owner
BEFORE INSERT ON owner
FOR EACH ROW
EXECUTE PROCEDURE new_owner();
```

Вміст таблиці Вланік (owner):

	 id_owner [PK] integer 	full_name character (40) 	telefon character (20) 	adress character (40) 
1	1	Иваненко Иван Иванович	0507476512	Арнаутська 12 ...
2	2	Фадеев Богдан Гордеевич	0507476512	Арнаутська 12 ...
3	3	Костин Герман Лаврентьевич	0507476324	Канатная 17 ...
4	4	Баранов Владимир Ярославович ...	0507765489	Ришильевская 54 ...
5	5	Мясников Моисей Лаврентьевич ...	0501234245	Подрадна 33 ...
6	6	Гришин Нинель Владиславович ...	0687216634	Щовринса 99 ...
7	7	Гаврилов Мартин Михайлович ...	0683415512	Бреуса 21 ...
8	8	Дементьев Леонтий Серапионович ...	0482819211	Вапнярка 129 ...

Операція вставки даних в таблицю owner:

```
insert into owner values(NEXTVAL('s_owner'),'Бабилунга Оксана
Юрьевна','0507476512','Проспект 64');
```

Такий номер телефону вже існує в таблиці

1	1	Иваненко Иван Иванович	0507476512	Арнаутська 12 ...
---	---	------------------------	------------	-------------------

Результат операції вставки:

```
ERROR: ОШИБКА:  Помилка: Власник з таким номером телефону вже існує
CONTEXT:  функция PL/pgSQL new_owner(), строка 5, оператор RAISE
```

Але якщо ми спробуємо додати цього власника, але з іншим номером, то все буде добре.

```
25 insert into owner values(NEXTVAL('s_owner'),'Бабилунга Оксана Юрьевна','0508924982','Проспект 64');
```

Data Output Explain Messages Notifications

INSERT 0 1

Query returned successfully in 47 msec.

2) Тригер на DELETE;

При видаленні гарантії з таблиці “Гарантія”, в таблиці “Техніка” id_guarantee відповідної гарантії стає дорівнювати NULL

SQL код цього тригера та тригерної функції:

```
CREATE OR REPLACE FUNCTION delete_guarantee()
```

```
RETURNS trigger
```

```
AS $$
```

```
BEGIN
```

```
UPDATE technique SET id_guarantee=NULL where
```

```
id_guarantee=old.id_guarantee;
```

```
RETURN old;
```

```
END;$$
```

```
LANGUAGE 'plpgsql';
```






```
CREATE OR REPLACE TRIGGER delete_guarantee
```

```
BEFORE DELETE ON guarantee
```




```
FOR EACH ROW
```

```
EXECUTE PROCEDURE delete_guarantee();
```

Таблиця *technique* до видалення гарантії з *id_guarantee=1*:

		id_technique [PK] integer 	category character (20) 	id_guarantee integer 	id_owner integer 
1		1	Монитор	4	3
2		2	Комп'ютер	5	6
3		3	Комп'ютер	6	4
4		4	Аудиоподсистема ...	3	5
5		5	Наушники	2	7
6		6	Клавіатура	8	2
7		7	Монитор	7	8
8		8	PS4	1	1






Таблиця *guarantee* до видалення гарантії з *id_guarantee=1*:

		id_guarantee [PK] integer 	date_end_guarantee date 
1		1	2021-12-14
2		2	2021-12-14
3		3	2021-09-12
4		4	2021-08-10
5		5	2022-06-11
6		6	2024-05-01
7		7	2023-01-23
8		8	2021-12-14




Операція видалення гарантії з таблиці *guarantee*:

delete from guarantee where id_guarantee = 1;

Таблиця *technique* після видалення гарантії з *id_guarantee=1*:

		id_technique [PK] integer 	category character (20) 	id_guarantee integer 	id_owner integer 
1		1	Монитор	4	3
2		2	Комп'ютер	5	6
3		3	Комп'ютер	6	4
4		4	Аудиоподсистема ...	3	5
5		5	Наушники	2	7
6		6	Клавіатура	8	2
7		7	Монитор	7	8
8		8	PS4	[null]	1

Таблиця *guarantee* до видалення гарантії з *id_guarantee=1*:

		id_guarantee [PK] integer 	date_end_guarantee date 
1		2	2021-12-14
2		3	2021-09-12
3		4	2021-08-10
4		5	2022-06-11
5		6	2024-05-01
6		7	2023-01-23
7		8	2021-12-14






Як ми можемо помітити, тригер відпрацював повністю коректно.

Напишемо ще один тригер на операцію delete , на це раз він буде видаляти техніку, а з нею і усі замовлення пов'язані з нею, а також хвости в інших SQL код тригера та тригерної функції:









```
CREATE OR REPLACE FUNCTION delete_technique_and_tails()
RETURNS trigger
AS $$
BEGIN
IF(EXISTS (SELECT id_order_info FROM order_info WHERE
id_technique=old.id_technique))
THEN
DELETE FROM master_order WHERE master_order.id_order_info IN
(SELECT id_order_info FROM order_info WHERE
id_technique=old.id_technique);
DELETE FROM order_info WHERE id_order_info IN
(SELECT id_order_info FROM order_info WHERE
id_technique=old.id_technique);
END IF;
RETURN old;
END;$$
LANGUAGE 'plpgsql';
```

```
CREATE OR REPLACE TRIGGER delete_technique_and_tails
BEFORE DELETE ON technique
FOR EACH ROW
EXECUTE PROCEDURE delete_technique_and_tails();
```




Таблиця *technique* до видалення техніки з *id_technique* = 8:

				
	id_technique [PK] integer	category character (20)	id_guarantee integer	id_owner integer
1	1	Монитор	4	3
2	2	Компьютер	5	6
3	3	Компьютер	6	4
4	4	Аудиоподсистема ...	3	5
5	5	Наушники	2	7
6	6	Клавиатура	8	2
7	7	Монитор	7	8
8	8	PS4	[null]	1

Таблиця *order_info* до видалення техніки з *id_technique* = 8:

							
	id_order_info [PK] integer	repair_type character (30)	repair_lenght character (20)	price numeric (50,2)	defect character varying	date_order_complete date	id_technique integer
1	1	Частичный ...	2 недели	1499.00	Не включается	[null]	1
2	2	Полный ...	месяц	3800.00	Очень сильно гудит	[null]	2
3	3	Частичный ...	[null]	999.00	Не включается	2021-12-14	3
4	4	[null]	[null]	1499.00	Не включается	2021-11-12	4
5	5	Полный ...	3 дня	700.00	Слышны помехи	[null]	5
6	6	[null]	2 недели	300.00	Залипают клавиши	[null]	6
7	7	[null]	[null]	1499.00	Не включается	2021-11-13	8
8	8	Частичный ...	5 дней	1000.00	Битые пиксели	[null]	7

Таблиця *master_order* до видалення техніки з *id_technique* = 8 (замовлення 7):

		
	id_order_info [PK] integer	id_master [PK] integer
1	3	2
2	3	1
3	1	2
4	1	3
5	2	4
6	4	5
7	4	6
8	5	6
9	6	3
10	7	1

Видалення техніки з $id_technique=8$;






```
77 delete from technique where id_technique =8;
78
```

Data Output Explain Messages Notifications









DELETE 1

Query returned successfully in 72 msec.




Таблиця *technique* після видалення техніки з $id_technique = 8$:

					
	id_technique [PK] integer	category character (20)	id_guarantee integer	id_owner integer	
1	1	Монитор	4	3	
2	2	Комп'ютер	5	6	
3	3	Комп'ютер	6	4	
4	4	Аудиоподсистема ...	3	5	
5	5	Наушники	2	7	
6	6	Клавіатура	8	2	
7	7	Монитор	7	8	

Таблиця *order_info* після видалення техніки з $id_technique = 8$:

								
	id_order_info [PK] integer	repair_type character (30)	repair_lenght character (20)	price numeric (50,2)	defect character varying	date_order_complete date	id_technique integer	
1	1	Частичный ...	2 недели	1499.00	Не включается	[null]		1
2	2	Полный ...	месяц	3800.00	Очень сильно гудит	[null]		2
3	3	Частичный ...	[null]	999.00	Не включается	2021-12-14		3
4	4	[null]	[null]	1499.00	Не включается	2021-11-12		4
5	5	Полный ...	3 дня	700.00	Слышны помехи	[null]		5
6	6	[null]	2 недели	300.00	Залипают клавиши	[null]		6
7	8	Частичный ...	5 дней	1000.00	Битые пиксели	[null]		7

Таблиця *master_order* після видалення техніки з $id_technique = 8$ (замовлення 7):

			
	id_order_info [PK] integer	id_master [PK] integer	
1	3	2	
2	3	1	
3	1	2	
4	1	3	
5	2	4	
6	4	5	
7	4	6	
8	5	6	
9	6	3	
10	8	3	

3) Тригер на UPDATE






Коли ми намагаємось змінити гарантію техніки та вказуємо id гарантії, якої нема в таблиці guarantee, створюється гарантія з вказаним id та вже після цього у техніки змінюється id гарантії не id нещодавно створеної гарантії.

SQL код:




```
CREATE OR REPLACE FUNCTION update_technique()
RETURNS trigger
AS $$
BEGIN
IF(select id_guarantee from guarantee where id_guarantee=new.id_guarantee) IS
NULL
THEN
INSERT INTO guarantee values (new.id_guarantee,NULL);
END IF;
RETURN new;
END;$$
LANGUAGE 'plpgsql';
```

```
CREATE OR REPLACE TRIGGER update_technique
BEFORE UPDATE ON technique
FOR EACH ROW
EXECUTE PROCEDURE update_technique();
```






Таблиця *technique* до зміни *id_guarantee*:

	 id_technique [PK] integer 	category character (20) 	id_guarantee integer 	id_owner integer 
1	1	Монитор	4	3
2	2	Комп'ютер	5	6
3	3	Комп'ютер	6	4
4	4	Аудиоподсистема ...	3	5
5	5	Наушники	2	7
6	6	Клавіатура	8	2
7	7	Монитор	7	8




Таблиця *guarantee* до зміни *id_guarantee* в *technique*:

	 id_guarantee [PK] integer 	date_end_guarantee date 
1	2	2021-12-14
2	3	2021-09-12
3	4	2021-08-10
4	5	2022-06-11
5	6	2024-05-01
6	7	2023-01-23
7	8	2021-12-14

Таблиця *technique* після зміни *id_guarantee*:

		id_technique [PK] integer 	category character (20) 	id_guarantee integer 	id_owner integer 
1		2	Компьютер	5	6
2		3	Компьютер	6	4
3		4	Аудиоподсистема ...	3	5
4		5	Наушники	2	7
5		6	Клавиатура	8	2
6		7	Монитор	7	8
7		1	Монитор	12	3

Таблиця *guarantee* після зміни *id_guarantee* в *technique*:

		id_guarantee [PK] integer 	date_end_guarantee date 
1		2	2021-12-14
2		3	2021-09-12
3		4	2021-08-10
4		5	2022-06-11
5		6	2024-05-01
6		7	2023-01-23
7		8	2021-12-14
8		12	[null]

4) Користувальницька функція (збережена процедура), яка викликається оператором select.

Створюємо функцію, яка буде видаляти усі гарантії з датами, які раніше ніж вказана дата (треба не забувати про те, що id видаляємої гарантії може бути в таблиці technique)

```
select * from guarantee where date_end_guarantee < '14.12.2021';
```

	id_guarantee [PK] integer		date_end_guarantee date
1		3	2021-09-12
2		4	2021-08-10

Як ми можемо побачити, існує 2 гарантії в таблиці guarantee, які вже закінчились. Отже напишемо функцію.

Ми маємо тригер , який при видаленні гарантії з таблиці “Гарантія” , в таблиці “Техніка” id_guarantee відповідної гарантії стає дорівнювати NULL, а також маємо тригер на Update таблиці техніка. Отже, при видаленні гарантії з таблиці guarantee, спочатку спрацює тригер, метою якого є видалення id_guarantee з таблиці technique, а через це спрацює тригер Update , бо ми змінюємо дані в таблиці technique. Тому щоб функція працювала коректно, треба спочатку трішки змінити тригерну функцію (update_technique()), а саме

```
CREATE OR REPLACE FUNCTION update_technique()
RETURNS trigger
AS $$
BEGIN
IF(new.id_guarantee IS NULL)
THEN
RETURN new;
END IF;
IF(select id_guarantee from guarantee where id_guarantee=new.id_guarantee) IS NULL
THEN
```

якщо ми бажаємо змінити гарантію техніки встановлюючи NULL, ми не будемо спочатку шукати в таблиці guarantee рядок з id_guarantee=NULL, та якщо його

нема (а його нема бо на первинний ключ встановлено обмеження NOT NULL) , будемо створювати рядок з id_guarantee = NULL (такого бути не може). Тому ми вказуємо, якщо id_guarantee в таблиці technique встановлюємо як NULL, ми пропускаємо тригер update_technique.

Отже, таблиця technique до роботи функції:

	id_technique [PK] integer	category character (20)	id_guarantee integer	id_owner integer
1	2	Компьютер	5	6
2	3	Компьютер	6	4
3	4	Аудиоподсистема ...	3	5
4	5	Наушники	2	7
5	6	Клавиатура	8	2
6	7	Монитор	7	8
7	1	Монитор	12	3

Таблиця guarantee до роботи функції:

	id_guarantee [PK] integer	date_end_guarantee date
1	2	2021-12-14
2	3	2021-09-12
3	4	2021-08-10
4	5	2022-06-11
5	6	2024-05-01
6	7	2023-01-23
7	8	2021-12-14
8	12	[null]

SQL код функції:

Як аргумент вказується дата. Всі гарантії , які мають дати менш ніж вказана - видаляються.





```
CREATE OR REPLACE FUNCTION delete_old_guarantee(date_end date)
RETURNS INTEGER
AS $$
BEGIN
DELETE FROM guarantee where id_guarantee
IN
(SELECT id_guarantee from guarantee where date_end_guarantee < date_end);
RETURN 1;
END;$$
LANGUAGE 'plpgsql'
```

select delete_old_guarantee('14.12.2021');



Результат запису:

delete_old_guarantee	
integer	
1	1

Отже, таблиця *technique* після роботи функції:

	 id_technique [PK] integer	 category character (20)	 id_guarantee integer	 id_owner integer
1	2	Компьютер	5	6
2	3	Компьютер	6	4
3	5	Наушники	2	7
4	6	Клавиатура	8	2
5	7	Монитор	7	8
6	1	Монитор	12	3
7	4	Аудиоподсистема ...	[null]	5

Таблиця *guarantee* після роботи функції:

	 id_guarantee [PK] integer	 date_end_guarantee date
1	2	2021-12-14
2	5	2022-06-11
3	6	2024-05-01
4	7	2023-01-23
5	8	2021-12-14
6	12	[null]

Цією функцією я хотів показати, що при створенні функцій необхідно пам'ятати про створенні тригери, бо вони можуть сильно вплинути на роботу функції.

Висновок:

Під час лабораторної роботи я ознайомився з тригерами та власними функціями. На основі цих знань створив свої тригери на операції модифікації даних та власну функцію.