**МІНІСТЕРСТВО ОСВІТИ І НАУКИ**

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»**

**Інститут комп'ютерних наук та інформаційних технологій**

**Кафедра систем штучного інтелекту**

****

**Звіт до лабораторної роботи №12**

з дисципліни

“Організація баз даних та знань”

**Виконав**:

ст. гр. КН-211

Петров Дмитро

**Викладач:**

Якимишин Х.М.

Львів – 2020

**Тема:** Розробка та застосування тригерів.

**Мета роботи:** Розробити SQL запити, які моделюють роботу тригерів: каскадне знищення, зміна та доповнення записів у зв’язаних таблицях.

#### Хід роботи

Потрібно розробити тригери, які виконуватимуть наступні дії.

1. Каскадне оновлення в таблиці article при зміні таблиці oder.
2. Оновлення даних при видаленні в таблиці article.
3. Тригер для таблиці group, що каскадно видалить усі залежності перед видаленням.

1. Каскадне оновлення в таблиці article при зміні таблиці oder.

delimiter //

CREATE

TRIGGER update\_order\_status

AFTER UPDATE ON db\_sht.order FOR EACH ROW

BEGIN

if (NEW.status = 'accomplished') then

BEGIN

-- set @newArt = (select goods\_id from goods where order\_id = new.id);

update article set amount = amount -1 where id in(select goods\_id from goods where order\_id = new.id) and amount > 0;

END;

END IF;

END//

delimiter ;

* 1. Створимо транзакцію, для перевірки його роботи.

start transaction;

set @ord = 12;

SELECT

goods.id as goods, goods\_id as article, order\_id, article.id as art\_id, goods\_name, article.amount, o.id as ord, o.status

FROM

(goods

INNER JOIN article)

INNER JOIN

db\_sht.order AS o ON goods.goods\_id = article.id

AND o.id = goods.order\_id

WHERE

order\_id = @ord;

-- accomplished

update db\_sht.order as o set o.status = 'accomplished' where id = @ord limit 1;

SELECT

goods.id as goods, goods\_id as article, order\_id, article.id as art\_id, goods\_name, article.amount, o.id as ord, o.status

FROM

(goods

INNER JOIN article)

INNER JOIN

db\_sht.order AS o ON goods.goods\_id = article.id

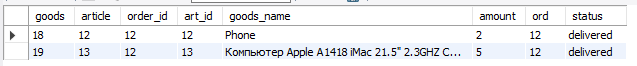
AND o.id = goods.order\_id

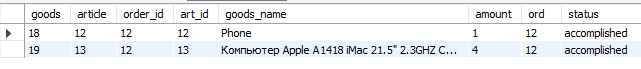
WHERE

order\_id = @ord;

rollback;

* 1. Результати:





1. Оновлення даних при видаленні в таблиці article.

delimiter //

CREATE

TRIGGER delete\_article

BEFORE DELETE ON article FOR EACH ROW

BEGIN

delete article\_group from article\_group where goods\_id = old.id;

-- insert into article value (55, 'test', 15, 1, 'beh');

set @testid = (select id from article where goods\_name = 'TEST' limit 1);

update goods set goods\_id = @testid where goods\_id = old.id;

END//

delimiter ;

* 1. Створимо транзакцію для перевірки тригера

START TRANSACTION;

insert into article value (55, 'test', 15, 1, 'beh');

select \* from article;

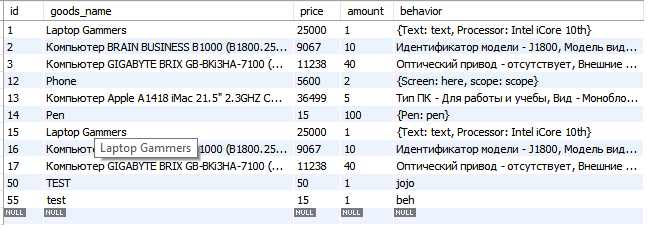
delete article from article where id = 12;

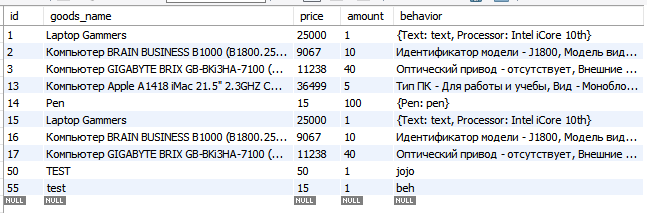
select \* from article;

select \* from goods;

ROLLBACK;

* 1. Результати







1. Тригер для таблиці group, що каскадно видалить усі залежності перед видаленням.

CREATE

TRIGGER delete\_group

BEFORE DELETE ON db\_sht.group FOR EACH ROW

DELETE db\_sht . article\_group FROM db\_sht.article\_group WHERE

group\_id = OLD.id;

* 1. Транзакція для перевірки

START TRANSACTION;

SELECT \* FROM db\_sht.group where id in(18, 19);

SELECT \* FROM article\_group where group\_id in(18, 19);

DELETE db\_sht.group FROM db\_sht.group

WHERE

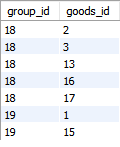
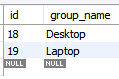
id = 18;

SELECT \* FROM db\_sht.group where id in(18, 19);

SELECT \* FROM article\_group where group\_id in(18, 19);

ROLLBACK;

* 1. Результати





**Висновок** : на цій лабораторній роботі було розглянуто тригери, їх призначення, створення та використання. Було розроблено тригери для таблиць order, article, groups..