

Міністерство освіти і науки України  
Національний університет «Львівська політехніка»  
Інститут комп'ютерних наук та інформаційних технологій  
Кафедра «Системи штучного інтелекту»



Звіт до лабораторної роботи №12  
З дисципліни «Організація Баз Даних»

Виконав:  
студент групи КН-208  
Дерев'янний Андрій

Прийняла:  
Мельникова Н.І

Львів-2020

**Тема:** Розробка та застосування тригерів

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

**Завдання:**

Розробити SQL запити, які моделюють роботу тригерів: каскадне знищення, зміна та доповнення записів у зв'язаних таблицях

## Виконання завдання:

Почнемо виконання даної лабораторної роботи з створенні в таблиці worker поля insurance\_cost і заповнимо його нулями:

```
3 • alter table worker
4   add column insurance_cost int(10) default 0;
5
6 • select * from worker;
```

id	name	surname	date_of_birth	adress	degree	contacts	passport_number	insurance_cost
1	lynn	gunn	1994-05-15	some_home_adress1	some_degree1	some_contacts	1	0
2	alex	albon	1996-03-23	some_adress2	some_degree2	some_contacts	2	0
3	hayley	williams	1988-12-27	some_adress3	some_degree3	some_contacts	3	0
4	billie	eilish	2001-12-18	some_adress4	some_degree4	some_contacts	4	0
5	matty	healy	1989-04-08	some_adress5	some_degree5	some_contacts	5	0
6	chloe	price	1994-05-11	some_adress6	some_degree6	some_contacts	6	0
7	finneas	o'connell	1997-07-30	some_adress7	some_degree7	some_contacts	7	0
8	daniel	raddcliffe	1989-07-23	some_adress8	some_degree8	some_contacts	8	0
9	tom	felton	1987-09-22	some_adress9	some_degree9	some_contacts	9	0
10	max	caulfield	1995-09-21	some_adress10	some_degree10	some_contacts	10	0

Перший тригер буде активується при внесенні нових даних. В таблиці Healthcare є поле cost – вартість лікарняного для певного робітника. При додаванні значень в це поле, поле insurance\_cost також буде набувати нового значення:

```
3 • CREATE TRIGGER insurance AFTER
4   INSERT ON db.healthcare FOR EACH ROW
5   UPDATE healthcare INNER JOIN worker
6   SET worker.insurance_cost = worker.insurance_cost + NEW.cost
7   WHERE NEW.worker_id = worker.id;
```

Перевіримо чи він працює:

```
9 • insert into healthcare (id, worker_id, illness, start_of_illness, recovery_period, cost )
10 values (12, 10, 'time traveller', '2013-10-13', 5, 100);
11
12 • select * from healthcare;
13
```

id	worker_id	illness	start_of_illness	recovery_period	cost
9	9	NULL	NULL	NULL	NULL
10	10	NULL	NULL	NULL	NULL
11	9	brok...	2020-03-12	3	500
12	10	time ...	2013-10-13	5	100

```
14 • select * from worker;
```

```
15
```

Result Grid								
Filter Rows: <input type="text"/>								
Edit:      Export/Import:   Wrap Cell Content:								
id	name	surname	date_of_birth	adress	degree	contacts	passport_number	insurance_cost
8	daniel	raddcliffe	1989-07-23	some_adress8	some_degree8	some_contacts	8	0
9	tom	felton	1987-09-22	some_adress9	some_degree9	some_contacts	9	500
10	max	caulfield	1995-09-21	some_adress10	some_degree10	some_contacts	10	100

Як ми бачимо, при внесенні нових даних в поле cost таблиці healthcare, ці самі дані вводяться і в поле insurance\_cost таблиці worker.

Тепер перейдемо до оновлення(зміни) даних. Будемо оперувати в тих самих таблицях healthcare і worker – тригер активується при зміні даних поля cost, поле insurance\_cost також буде набувати нового значення:

```
5 • CREATE TRIGGER insurance_update AFTER
6   UPDATE ON db.healthcare FOR EACH ROW
7   UPDATE worker INNER JOIN healthcare
8   SET worker.insurance_cost = worker.insurance_cost - OLD.cost + NEW.cost
9   WHERE NEW.worker_id = worker.id;
```

Перевіримо чи він працює:

```
11 • UPDATE `db`.`healthcare` SET `cost` = '200' WHERE (`id` = '12');
```

```
12
```

```
13 • select * from healthcare;
```

```
14
```

Result Grid						
Filter Rows: <input type="text"/>						
Edit:      Export/Import:   Wrap Cell Content:						
id	worker_id	illness	start_of_illness	recovery_period	cost	
9	9	NULL	NULL	NULL	NULL	
10	10	NULL	NULL	NULL	NULL	
11	9	brok...	2020-03-12	3	600	
12	10	time ...	2013-10-13	5	200	
NULL	NULL	NULL	NULL	NULL	NULL	

```
15 • select * from worker
```

Result Grid								
Filter Rows: <input type="text"/>								
Edit:      Export/Import:   Wrap Cell Content:								
id	name	surname	date_of_birth	adress	degree	contacts	passport_number	insurance_cost
8	daniel	raddcliffe	1989-07-23	some_adress8	some_degree8	some_contacts	8	0
9	tom	felton	1987-09-22	some_adress9	some_degree9	some_contacts	9	600
10	max	caulfield	1995-09-21	some_adress10	some_degree10	some_contacts	10	200

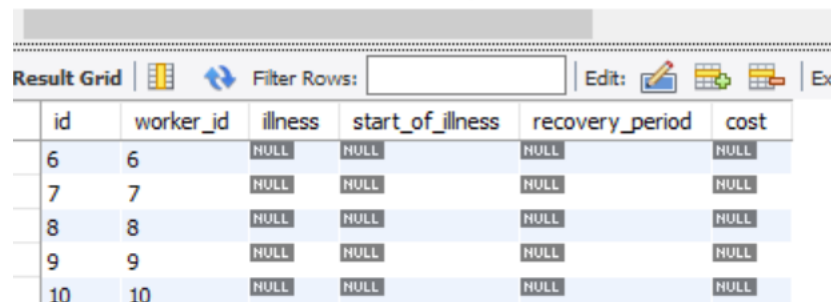
Як ми бачимо, при зміні даних в полі cost таблиці healthcare, ці самі дані змінюються і в полі insurance\_cost таблиці worker.

Залишається розробити тригер видалення даних. Суть його роботи проста - при видаленні даних з поля `cost`, поле `insurance_cost` також буде втрачати ці значення:

```
3 • CREATE TRIGGER clear_insurance AFTER
4   DELETE ON healthcare FOR EACH ROW
5   UPDATE worker INNER JOIN healthcare
6     SET worker.insurance_cost = worker.insurance_cost - OLD.cost
7     WHERE old.worker_id = worker.id;
```

Перевіримо чи він працює:

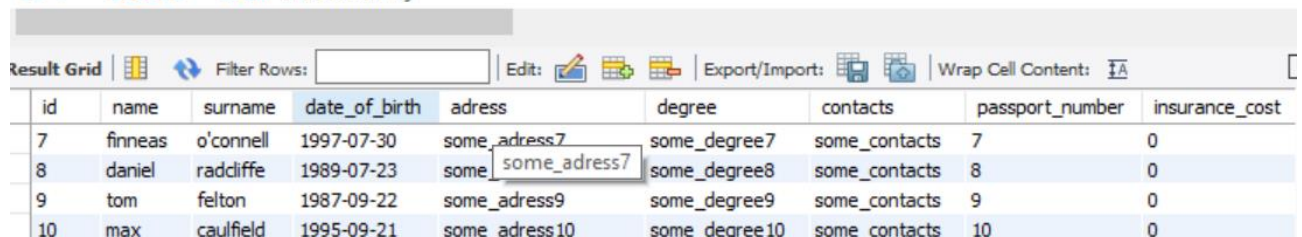
```
9 • DELETE FROM `db`.`healthcare` WHERE (`id` = '12');
10
11 • select * from healthcare;
```



The screenshot shows a database client interface with a 'Result Grid' tab selected. The grid displays the results of the query 'select \* from healthcare;'. The columns are: id, worker\_id, illness, start\_of\_illness, recovery\_period, and cost. The data is as follows:

id	worker_id	illness	start_of_illness	recovery_period	cost
6	6	NULL	NULL	NULL	NULL
7	7	NULL	NULL	NULL	NULL
8	8	NULL	NULL	NULL	NULL
9	9	NULL	NULL	NULL	NULL
10	10	NULL	NULL	NULL	NULL

```
11 • select * from healthcare;
```



The screenshot shows a database client interface with a 'Result Grid' tab selected. The grid displays the results of the query 'select \* from healthcare;'. The columns are: id, name, surname, date\_of\_birth, adress, degree, contacts, passport\_number, and insurance\_cost. The data is as follows:

id	name	surname	date_of_birth	adress	degree	contacts	passport_number	insurance_cost
7	finneas	o'connell	1997-07-30	some_adress7	some_degree7	some_contacts	7	0
8	daniel	raddcliffe	1989-07-23	some_adress7	some_degree8	some_contacts	8	0
9	tom	felton	1987-09-22	some_adress9	some_degree9	some_contacts	9	0
10	max	caulfield	1995-09-21	some_adress10	some_degree10	some_contacts	10	0

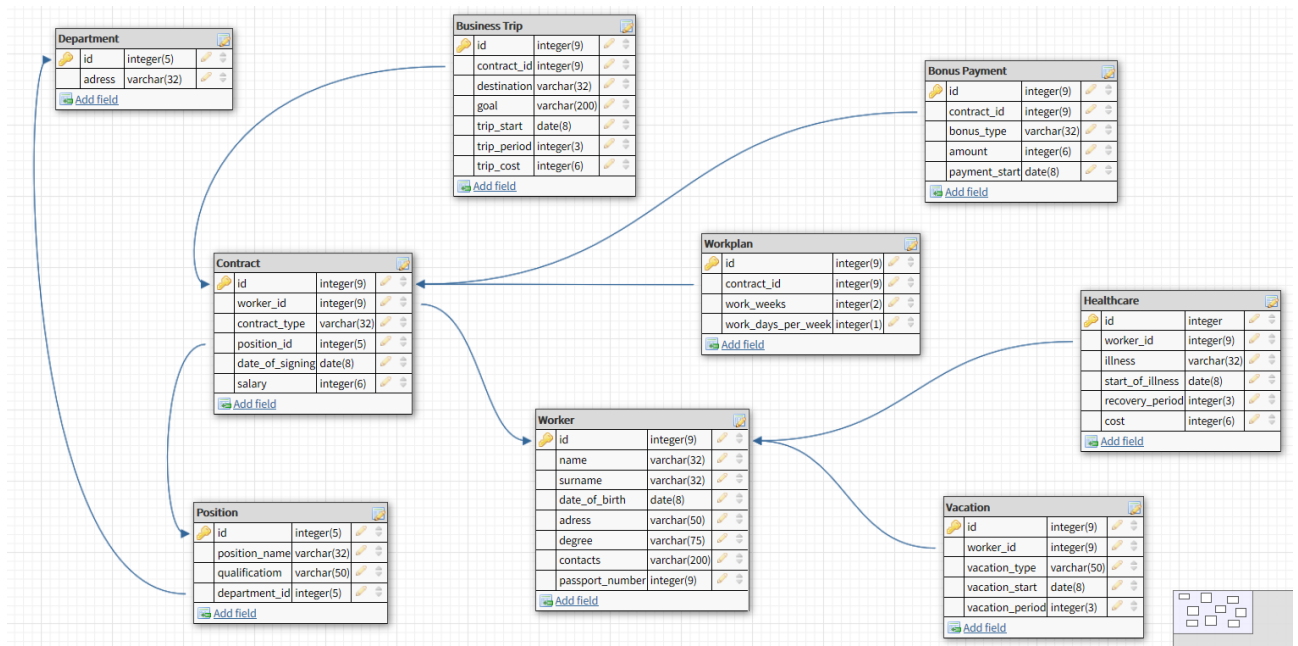
Як ми бачимо, при видаленні даних в полі `cost` таблиці `healthcare`, значення поля `insurance_cost` таблиці `worker` знову стало нулем.

## Висновок:

За час виконання лабораторної роботи я навчився розробляти SQL запити, які моделюють роботу тригерів: каскадне знищення, зміна та доповнення записів у зв'язаних таблицях

## Додаток:

Діаграма бази даних(відділ кадрів підприємства):



SQL-скрипт бази даних:

```
create database db;
```

```
use db;
```

```
CREATE TABLE `Department` (
```

```
    `id` INT(5) NOT NULL AUTO_INCREMENT,
```

```
    `adress` varchar(32) NOT NULL,
```

```
    PRIMARY KEY (`id`)
```

```
);
```

```
CREATE TABLE `Worker` (
```

```
    `id` INT(9) NOT NULL AUTO_INCREMENT,
```

```
    `name` varchar(32) NOT NULL,
```

```
    `surname` varchar(32) NOT NULL,
```

```
    `date_of_birth` DATE NOT NULL,
```

```
    `adress` varchar(50) NOT NULL,
```

```
    `degree` varchar(75) NOT NULL,
```

```
    `contacts` varchar(200) NOT NULL,
```

```
    `passport_number` INT(9) NOT NULL,
```

```

PRIMARY KEY (`id`)

);

CREATE TABLE `Vacation` (
    `id` INT(9) NOT NULL AUTO_INCREMENT,
    `worker_id` INT(9) NOT NULL,
    `vacation_type` varchar(50),
    `vacation_start` DATE,
    `vacation_period` INT(3),
    PRIMARY KEY (`id`),

    CONSTRAINT fk0 FOREIGN KEY (worker_id)
        REFERENCES Worker(id)
        ON DELETE CASCADE ON UPDATE CASCADE
);

```

```

CREATE TABLE `Position` (
    `id` INT(5) NOT NULL AUTO_INCREMENT,
    `position_name` varchar(32) NOT NULL,
    `qualification` varchar(50) NOT NULL,
    `department_id` INT(5) NOT NULL,
    PRIMARY KEY (`id`),

    CONSTRAINT fk1 FOREIGN KEY (department_id)
        REFERENCES Department(id)
        ON DELETE CASCADE ON UPDATE CASCADE
);

```

```

CREATE TABLE `Contract` (
    `id` INT(9) NOT NULL AUTO_INCREMENT,
    `worker_id` INT(9) NOT NULL,
    `contract_type` varchar(32) NOT NULL,
    `position_id` INT(5) NOT NULL,
    `date_of_signing` DATE NOT NULL,
    `salary` INT(6) NOT NULL,
    PRIMARY KEY (`id`),

```

```
CONSTRAINT fk2 FOREIGN KEY (worker_id)
REFERENCES Worker(id)
ON DELETE CASCADE ON UPDATE CASCADE,
```

```
CONSTRAINT fk3 FOREIGN KEY (position_id)
REFERENCES `Position`(id)
ON DELETE CASCADE ON UPDATE CASCADE
```

```
);
```

```
CREATE TABLE `Workplan` (
  `id` INT(9) NOT NULL AUTO_INCREMENT,
  `contract_id` INT(9) NOT NULL,
  `work_weeks` INT(2),
  `work_days_per_week` INT(1),
  PRIMARY KEY (`id`),
```

```
CONSTRAINT fk4 FOREIGN KEY (contract_id)
REFERENCES Contract(id)
ON DELETE CASCADE ON UPDATE CASCADE
```

```
);
```

```
CREATE TABLE `Bonus_Payment` (
  `id` INT(9) NOT NULL AUTO_INCREMENT,
  `contract_id` INT(9) NOT NULL,
  `bonus_type` varchar(32),
  `amount` INT(6),
  `payment_start` DATE,
  PRIMARY KEY (`id`),
```

```
CONSTRAINT fk5 FOREIGN KEY (contract_id)
REFERENCES Contract(id)
ON DELETE CASCADE ON UPDATE CASCADE
```

```
);
```

```
CREATE TABLE `Healthcare` (
```



```
`id` INT NOT NULL AUTO_INCREMENT,  
`worker_id` INT(9) NOT NULL,  
`illness` varchar(32),  
`start_of_illness` DATE,  
`recovery_period` INT(3),  
`cost` INT(6),  
PRIMARY KEY (`id`),
```

```
CONSTRAINT fk6 FOREIGN KEY (worker_id)  
REFERENCES Worker(id)  
ON DELETE CASCADE ON UPDATE CASCADE  
);
```

```
CREATE TABLE `Business_Trip` (  
  `id` INT(9) NOT NULL AUTO_INCREMENT,  
  `contract_id` INT(9) NOT NULL,  
  `destination` varchar(32),  
  `goal` varchar(200),  
  `trip_start` DATE,  
  `trip_period` INT(3),  
  `trip_cost` INT(6),  
  PRIMARY KEY (`id`),  
  
  CONSTRAINT fk7 FOREIGN KEY (contract_id)  
  REFERENCES Contract(id)  
  ON DELETE CASCADE ON UPDATE CASCADE  
);
```