**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ**

**“ЛЬВІВСЬКА ПОЛІТЕХНІКА”**

**Кафедра систем штучного інтелекту**



**Розрахунково-графічна робота**

з дисципліни:

**«Організація баз даних та знань»**

на тему:

**«Проектування БД для проекту RESTAVRATSIA»**

**Виконав студент групи** КН-209 Качмар О.І.

|  |  |
| --- | --- |
| Балів | Дата |
|  |  |

**Викладач** Мельникова Н. І.

**Львів – 2020р**

**Тема проекту:**

Створення веб-сервісу, призначеного для швидкої, зручної та безболісної комунікації та укладення угоди про реставрацію будь-якого виду об’єктів між замовником та компанією-виконавцем.Створення на платформі цього сервісу форуму відгуків та репортів про діяльність конкретної реставраційної компанії.

**Зміст**

Вступ………………………………………………………………………………3

1. Логічна схема…………………………………………………………………. 5

2. Опис структури………………………………………………………………...6

3. Фізична модель………………………………………………………………...8

4. Ділова модель………………………………………………………………….12

5. Запити…………………………………………………………………………..15

6.Висновки………………………………………………………………………..25

7.Список використаної літератури……………………………………………...26

2

**Вступ**

Кожен з нас має старовинні речі,які нам дорогі.Чи це антикварний кулон,який дістався вам від бабусі,чи годинник,що передав вам дідусь,який пройшов війну або ж пам’ятка вашого улюбленого міста,де ви народились і живете,що має надзвичайно вагому культурну та історичну цінність.

До прикладу, я громадянин Львова.Наше місто славиться багатьма перлинами архітектури,які занесені до пам’ятки ЮНЕСКО.Це старовинні будівлі,речі які в них знаходяться , пам’ятники, які ,на щастя, неможливо приватизувати.

На жаль.ніщо не вічне.І старовинні речі, пам’ятні для нас потребують не тільки постійного догляду ,але й відновлення, конкретніше кажучи, реставрації.

Реставрація – певні дії спрямовані на відновлення об’єктів та запобігання подальшому їх знищенню та забезпечення їх подальшої довговічності.

Так як, об’єкти, що підлягають реставрації дуже дорогі нам саме за їх пам’ять, антикварність тому, звісно ж реставраційні роботи проводяться з мінімальним втручанням в історичний матеріал об’єкта.Хороший реставратор береже культурну спадщину, тісно співпрацюючи з ученими, (технічними) мистецтвознавцями, археологами та антропологами. Автентичність, цілісність та стійкість - основні цінності, що формують догляд за спадщиною це міра того, наскільки успішною вона передається в майбутнє покоління.

Звісно ж , фахівці витрачають багато зусиль, щоб зберегти пам'ятки мистецтва і архітектури, так як кожен з них частина нашої історії,а як це зробить аматор?

Як людина, що не має жодного уявлення про процес,який дійсно можна назвати мистецтвом,володіючи річчю що має неоціненне культурно-історичне значення.

Уявіть собі,що станеться , якщо відбудеться втручання аматора в оригінальну давньоримську архітектуру або що станеться,коли дилетант почне якось змінювати старовинні скульптури доби італійського відродження.

Ми переконані, що нічо з нічого не виникає і безслідно не зникає.

Саме тому ми командно створили незалежний сервіс за допомогою якого як звичайний обиватель,який володіє старовинною , дорогою його серцю річчю так і цілі громади,спільноти людей, що потребують реставрації об’єктів будь-якого типу ,будь-якого значення можуть знайти справжніх майстрів,справжніх професіоналів своєї справи, які тільки і шукають, як щось комусь відреставрувати.

Нащ проект сервіс має user-friendly інтерфейс, отже його вікова категорія людей є необмежена.Все, що потрібно зробити зі сторони замовника – це перейти до ресурсу,зареєструватись та сформувати своє оголошення про потребу реставрації певного об’єкта з описом і фотографією та чекати.Через деякий час на замовлення подадуть заявку чимало майстрів,з пропозицією про співпрацю в якій вони вкажуть ціну за реставрацію вашого зазначеного в оголошенні об’єкта.Вам залишиться тільки обрати того кому довіритись і погодите пропозицію в ціновому аспекті.

Якщо ж ви виконавець, а не замовник - ви так само маєте зареєструватись на нашому веб-ресурсі та здійснити пошук оголошень.При знаходженні цікавого вам оголошення перейдіть за оголошенням,огляньте його і підпишіться.

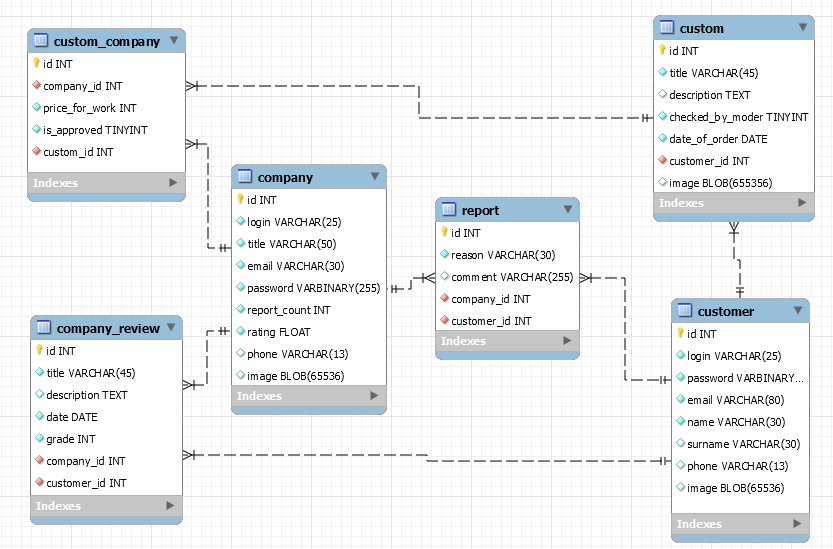
Ясна річ, що як і в реальному житті , так і в віртуальному середовищі є чимало шахраїв.Ці люди або не виконують своєї роботи,яка була домовлена, або виконують не приділяючи достатньо уваги.Ми не толеруємо таку поведінку і як хороші розробники розробили протидію шахраям. 3

Якщо вас на нашому ресурсі було обмануто чи вас чимось образила компанія, що виконувала ваше замовлення – ви можете сформувати скаргу(репорт) і надіслати її адміністраторам.Також залишіть негативний відгук про цю компанію, це знизить її статус і ,можливо її жертвами стане суттєва менша кількість людей.

Вдалого користування нашим ресурсом!

4

**Логічна схема БД проекту**

****

На схемі представлено 6 сутностей:

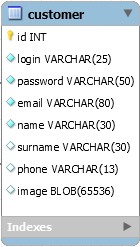
customer,company,report,custom,custom\_company,company\_review.

Цясхема представлена в нотації Баркера[6] ,бо сутності в ній представлені прямокутником,який містить ім’я сутності,її атрибути.Обов’язкові для вказання атрибути є NOT NULL.Реляційні зв’язки один-до-багатьох тут позначені відповідними стрілками.Для зв’язку багато-до-багатьох створені проміжні таблиці.

5

**Опис структури БД**

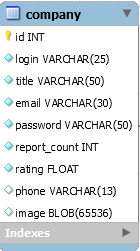
Задана БД restavratsia формується з наступних таблиць:

 Таблиця `customer`

В таблиці customer містяться дані про сутність замовник.Сутність[1] характеризується строковими атрибутами,які є NOT NULL ,це: login, password,email,name(ім’я замовника).

Строкові атрибути,які є NULL - це surname(прізвище),phone.

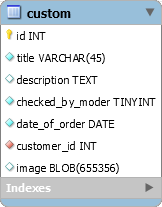
Крім того в цій таблиці зберігається інформація про особисте зображення замовника.Зображення є також NULL,типу BLOB з обмеженням даних на 64КБ.

 Таблиця `company`

Ця таблиця містить дані про сутність компанія-виконавець замовлення.Виконавець має наступні строкові NOT NULL[2] атрибути : login, title(назва компанії),email,password.

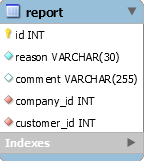
Атрибут phone – телефон компанії строкового типу NULL.

Атрибути report\_count(кількість скарг про порушення норм) є цілочисельного типу NOT NULL, та rating(рейтинг компанії) типу числа з плаваючою комою NOT NULL .Значення цих атрибутів обчислюється відповідними тригерами.

 Таблиця `custom`

Custom описує сутність замовлення.Замовлення характеризується строковими NOT NULL атрибутами заголовок,опис замовлення.Кожне замовлення має дату публікації ,цей атрибут ми назвали date\_of\_order типу DATE NOT NULL .При виведенні всіх замовлень ми використовували TINYINT(BOOL) поле checked\_by\_moder – перевірка на схвалення модератором такого замовлення,.Кожне замовлення має замовника,тому ми вказали customer\_id типу INT ,зовнішній ключ на таблицю `customer`(customer реалізує зв’язок один-до-багатьох з таблицею custom).

Таблиця `report`



Report описує сутність репортів(скарг).Скарга має строкові NOT NULL атрибути reason(причина) та comment.Так як репорт про компанію формує замовник , тому таблиця `report` має атрибути-зовнішні ключі ‘company\_id’ та ‘customer\_id’ типу INT ,таким чином реалізували зв’язок багато-до-багатьох між таблицями Company та Customer.

6

Таблиця `custom\_company`

В таблиці custom\_company зберігається інформація про сутність замовлення-компанія.Таблиця зберігає пропозиції аплікантів на певне замовлення з пропонованою ціною, price\_for\_work INT NOT NULL, за виконання реставраційної роботи.Отже,доречно буде використати атрибути-зовнішні ключі ‘company\_id’ та ‘custom\_id’ на відповідні таблиці,таким чином формуючи зв’язок багато-до-багатьох між компанією,що ‘подається’ на замовлення.Якщо замовник затвердить компанію ,то в значення атрибуту is\_approved TINYINT NOT NULL відповідного запису таблиці буде значення 1,інакше 0.

Таблиця `company\_review`

В цій таблиці зберігається інформація про сутність відгуку про компанію.Користувач може конфігурувати свій відгук задаючи значення для строкових NOT NULL атрибутів - title,description.Дата створення відгуку буде збережено для атрибуту date типу DATE NOT NULL.Оцінка компанії буде визначатись полем grade INT NOT NULL. Так як відгук про компанію формує замовник, тому таблиця `company\_review` має атрибути-зовнішні ключі ‘company\_id’ та ‘customer\_id’ типу INT ,таким чином реалізували зв’язок багато-до-багатьох між таблицями Company та Customer.

Крім того,що всі зовнішні ключі ми зробили індексованими(всі первинні за замовчуванням індексовані PRIMARY),що мало б пришвидшити операцію JOIN для таблиць ,ми встановили для всіх Foreign Key ON DELETE CASCADE ON UPDATE CASCADE.Тобто при видаленні запису в таблиці company відповідний запис з таблиці company\_review буде видалятись,так само при зміні Id запису з company відповідний foreign key з таблиці company\_review зміниться.

Таблиці нашої БД нормалізовані за першими 3-ома формами нормалізації[2]:

1)Всі вони мають унікальний ключ , в нашому випадку id,всі значення атомарні.

2)Відповідність першій формі,незалежність частинно від складеного ключа(так як таких наші таблиці не містять,тому відповідає).

3)Відповідність другій формі, незалежність неключових полів.[5]

7

**Фізична модель БД**

CREATE DATABASE IF NOT EXISTS restavratsia DEFAULT CHARACTER SET utf8 ;

USE restavratsia ;

use restavratsia;

CREATE TABLE IF NOT EXISTS company (

id INT NOT NULL AUTO\_INCREMENT,

login VARCHAR(25) NOT NULL,

title VARCHAR(50) NOT NULL,

email VARCHAR(30) NOT NULL,

password VARCHAR(25) NOT NULL,

report\_count INT NOT NULL,

rating FLOAT NOT NULL,

phone VARCHAR(13) NULL,

image BLOB(65536) NULL,

PRIMARY KEY ( id ))

ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS customer (

id INT NOT NULL AUTO\_INCREMENT,

login VARCHAR(25) NOT NULL,

password VARCHAR(25) NOT NULL,

email VARCHAR(30) NOT NULL,

name VARCHAR(30) NOT NULL,

surname VARCHAR(30) NULL,

phone VARCHAR(13) NULL,

image BLOB(65536) NULL,

PRIMARY KEY ( id ))

ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS custom (

id INT NOT NULL AUTO\_INCREMENT, 8

title VARCHAR(45) NOT NULL,

description TEXT NULL,

checked\_by\_moder TINYINT NOT NULL,

date\_of\_order DATE NOT NULL,

customer\_id INT NOT NULL,

image BLOB(655356),

PRIMARY KEY ( id ),

INDEX fk\_custom\_customer1\_idx ( customer\_id ASC),

CONSTRAINT fk\_custom\_customer1

FOREIGN KEY ( customer\_id )

REFERENCES restavratsia . customer ( id )

ON DELETE CASCADE #[2]

ON UPDATE CASCADE)

ENGINE = InnoDB; #[3]

CREATE TABLE IF NOT EXISTS customcompany (

id INT NOT NULL AUTO\_INCREMENT,

company\_id INT NOT NULL,

price\_for\_work INT NOT NULL,

is\_approved TINYINT NOT NULL,

custom\_id INT NOT NULL,

PRIMARY KEY ( id ),

INDEX company\_id\_idx ( company\_id ASC),

INDEX fk\_customcompany\_custom1\_idx ( custom\_id ASC),

CONSTRAINT fk\_Company\_company\_id

FOREIGN KEY ( company\_id )

REFERENCES restavratsia . company ( id )

ON DELETE CASCADE

ON UPDATE CASCADE,

CONSTRAINT fk\_customcompany\_custom1

FOREIGN KEY ( custom\_id )

REFERENCES restavratsia . custom ( id )

9

ON DELETE CASCADE

ON UPDATE CASCADE)

ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS company\_review (

id INT NOT NULL,

title VARCHAR(45) NOT NULL,

description TEXT NULL,

date DATE NOT NULL,

grade INT NOT NULL,

company\_id INT NOT NULL,

customer\_id INT NOT NULL,

PRIMARY KEY ( id ),

INDEX fk\_company\_review\_company1\_idx ( company\_id ASC),

INDEX fk\_company\_review\_customer1\_idx ( customer\_id ASC),

CONSTRAINT fk\_company\_review\_company1

FOREIGN KEY ( company\_id )

REFERENCES restavratsia . company ( id )

ON DELETE CASCADE

ON UPDATE CASCADE,

CONSTRAINT fk\_company\_review\_customer1

FOREIGN KEY ( customer\_id )

REFERENCES restavratsia . customer ( id )

ON DELETE CASCADE

ON UPDATE CASCADE)

ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS report (

id INT NOT NULL,

reason VARCHAR(30) NOT NULL,

comment VARCHAR(255) NULL, 10

company\_id INT NOT NULL,

customer\_id INT NOT NULL,

PRIMARY KEY ( id ),

INDEX fk\_report\_company1\_idx ( company\_id ASC),

INDEX fk\_report\_customer1\_idx ( customer\_id ASC),

CONSTRAINT fk\_report\_company1

FOREIGN KEY ( company\_id )

REFERENCES restavratsia . company ( id )

ON DELETE CASCADE

ON UPDATE CASCADE,

CONSTRAINT fk\_report\_customer1

FOREIGN KEY ( customer\_id )

REFERENCES restavratsia . customer ( id )

ON DELETE CASCADE

ON UPDATE CASCADE)

ENGINE = InnoDB;

11

**Ділова модель**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Таблиці  Функції | Компанія-виконавець | Замовник | Замовлення | Заяви  (custom\_company) | Репорти | Відгуки |
| Хешування паролю при створенні користувачів | \* | \* |  |  |  |  |
| Реєстрація,  Видалення,  Апдейт,селекція даних  замовника |  | \* |  |  |  |  |
| Реєстрація,  Видалення,  Апдейт,селекція даних  виконавця | \* |  |  |  |  |  |
| Створення,  Видалення,  Апдейт,селекція даних конкретного замовлення |  |  | \* |  |  |  |
| Вибірка замовлень певного замовника |  | \* | \* |  |  |  |
| Подача заявок на виконання реставраційної роботи. |  |  |  | \* |  |  |
| Перерахунок рейтингу компанії(тригер) | \* |  |  |  |  | \* |
| Створення відгуків користувачами про компанії |  |  |  |  |  | \* |
| Вибірка компаній які подали заяву на конкретне оголошення | \* | \* |  | \* |  |  |
| Конфігурація репорту про компанію |  |  |  |  | \* |  |
| Перерахунок кількості репортів (тригер) | \* |  |  |  | \* |  |
| Вибірка за допомогою курсору відгуків про конкретну компанію | \* | \* |  |  |  | \* |

1) Хешування паролю при створенні користувачів.

Ця операція буде виконана за допомогою тригера,який перед вставкою даних про користувача буде хешувати його пароль.За типом події – insert.За часом опрацювання-before.Працює на рівні рядка. Працює в тій же транзакції,а не автономно.

2)Реєстрація,Видалення,Апдейт,селекція даних замовника.

Це відповідно Insert , Delete ,Update та Select операції ,які виконуються з таблицею customer,стосуються всіх атрибутів таблиці .Операції update може також бути здійснена вкладеним select для update за іменем login замовника.

3)Реєстрація,Видалення,Апдейт,селекція даних компанії-виконавця.

Це відповідно Insert , Delete ,Update та Select операції ,які виконуються з таблицею company,стосуються всіх атрибутів таблиці.

4) Створення,Видалення,Апдейт,селекція даних конкретного замовлення

Це відповідно Insert , Delete та Select операції ,які виконуються з таблицею custom,стосуються всіх атрибутів таблиці.Update здійснюється при передачі всіх даних про замовлення.(так працював серверний вбудований метод).

5) Вибірка замовлень певного замовника

Це операція селекції замовлення з таблиці custom , де значення атрибуту customer\_id , замовника , що створив замовлення буде рівне тому,яке є в запиті.

6) Подача заявки на виконання реставраційної роботи

Це insert запит,який спрацьовує при подачі компанією заявки про виконання реставраційної роботи.Створює запис в таблиці custom\_company.Отже,вказуються значення атрибутів company\_id,custom\_id.Крім того, компанія вказує ціну за виконання роботи для price\_for\_work-атрибуту.Також вказується при insert значення атрибуту is\_approved=false,коли ж замовник схвалює компанію is\_approved змінюється на true.

7)Перерахунок рейтингу компанії-

Ця операція здійснюється за рахунок тригера,який при додаванні відгуку про компанію здійснює перерахунок оцінки компанії,знаходячи середнє арифметичне атрибутів grade таблиці company\_review та виконує update поля rating таблиці company.За типом події – Update.За часом дії – after.За об’єктом бд – на рівні рядка.В цій же транзакції.

13

8) Створення відгуку про компанію.

Це insert-запит, який створює запис в таблиці company\_review,відгук замовника про компанію,отже мають бути вказані значення атрибутів company\_id,customer\_id та тих , які формують відгук :title,description,date,grade.

9) Вибірка компаній , які подали заяву на конкретне оголошення.

Ця операція здійснюється злиттям таблиць custom,company та custom\_company за id.Результатом є вибірка значень атрибутів: title,decription таблиці custom і price\_for\_work таблиці custom\_company,тобто вивід даних компанія,опис замовлення і ціну реставраційної роботи.

10) Перерахунок кількості репортів.

Ця операція здійснюється за рахунок тригера,який при додаванні репорту про компанію здійснює перерахунок кількості репортів на компанію(кількість значеннь конкретної компанії атрибуту company\_id) і виконує update company. За типом події – Update.За часом дії – after.За об’єктом бд – на рівні рядка.В цій же транзакіції

11) Конфігурування репорту про компанію.

Це insert-запит, який створює запис в таблиці report,скарга замовника на компанію,отже мають бути вказані значення атрибутів company\_id,customer\_id та тих , які конфігурують скаргу : reason(причини:спам,ненормативна лексика і ін.),comment(докладний коментар скарги).

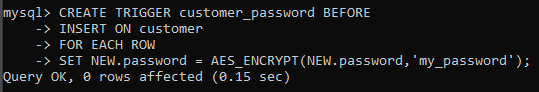
12) Вибірка за допомогою курсору відгуків про конкретну компанію

Виконуючи цю операцію, було створено procedure ,в якій було задекларовано курсор та локальні змінні : title\_review, description\_review, title\_company, full\_name\_user, report\_counter.В ці змінні при циклічному зчитуванні з курсору (попередньо його відкривши)записувались результати query.Як тільки в змінні було зчитано дані результат виводився в консоль,курсор закривався , процедура завершувалась.Запит стосувався таблиць company\_review,company,custom – виведення інформації про company\_review.title as review,company\_review.description as description,company.title as company,Concat(name," ",surname) as user\_full\_name,report\_count за допомогою декількох straight\_join . З групуванням за компанією,сортуванням за датою відгуку. Розумію, що використання курсорів[4] знижує продуктивність прикладних програм[4] , але ми використали подібний за суттю реалізації код в нашому проекті.Детальніший опис біля відповідної квері.

14

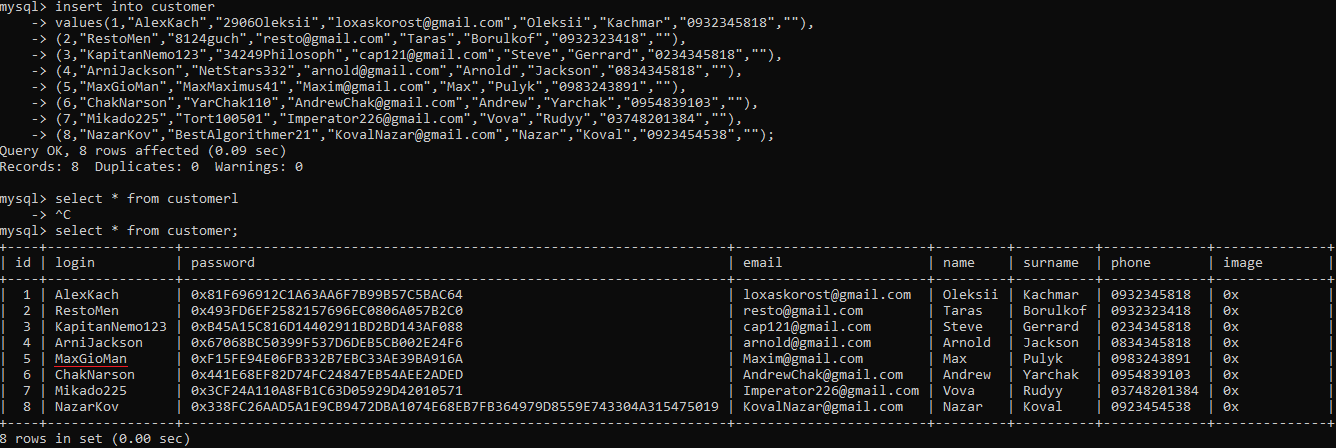
**Запити до БД**

1) Хешування паролю при створенні користувачів.

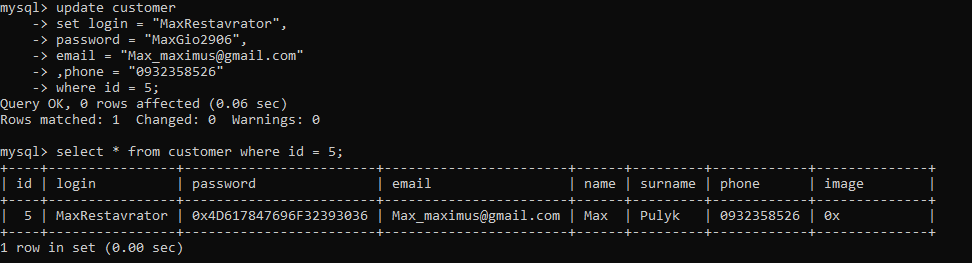


2)Реєстрація,Видалення,Апдейт,селекція даних замовника[7].

Реєстрація замовників

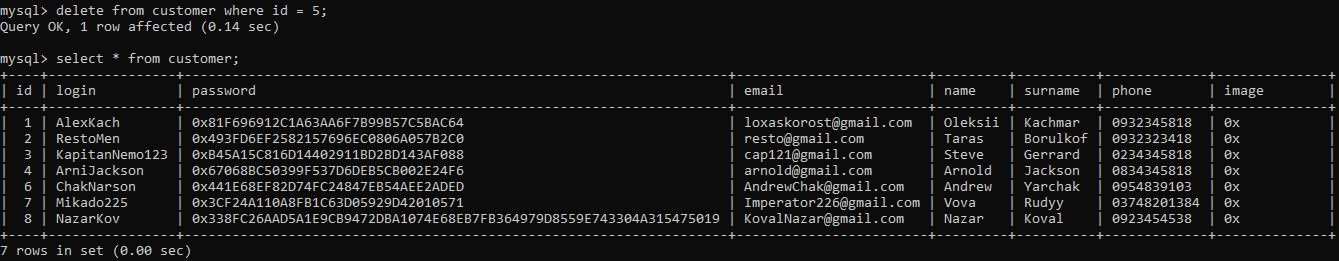


Апдейт,селекція замовника за id.



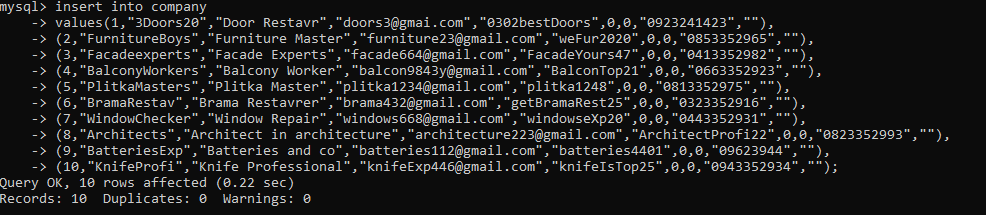
15

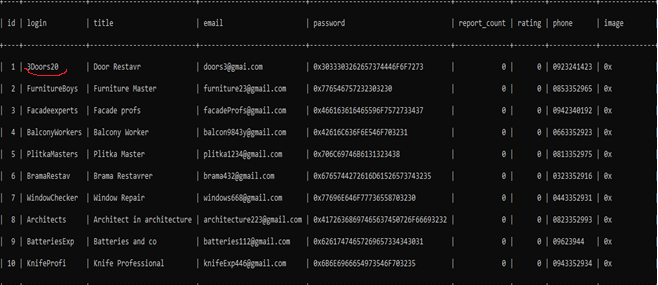
Видалення замовника за id-полем.



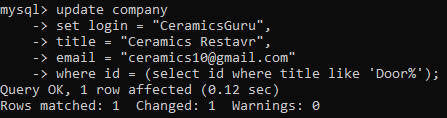
3)Реєстрація,Видалення,Апдейт,селекція даних компанії-виконавця.

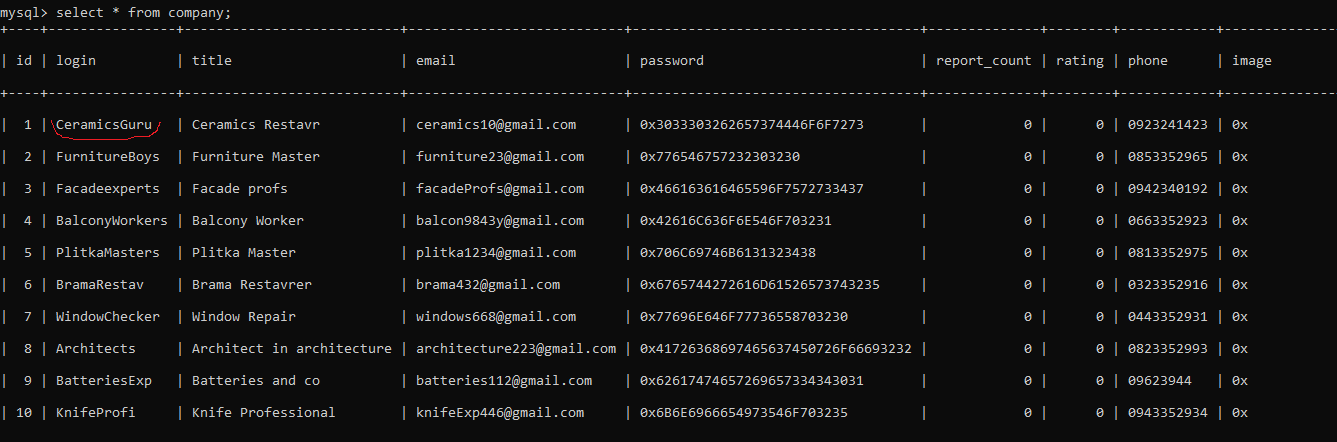
Реєстрація компаній-виконавців.



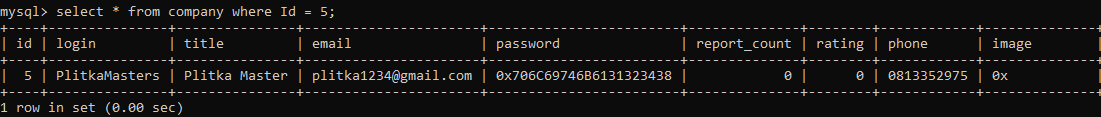


Оновлення даних про конкретну компанію(що спеціалізується по певній області)

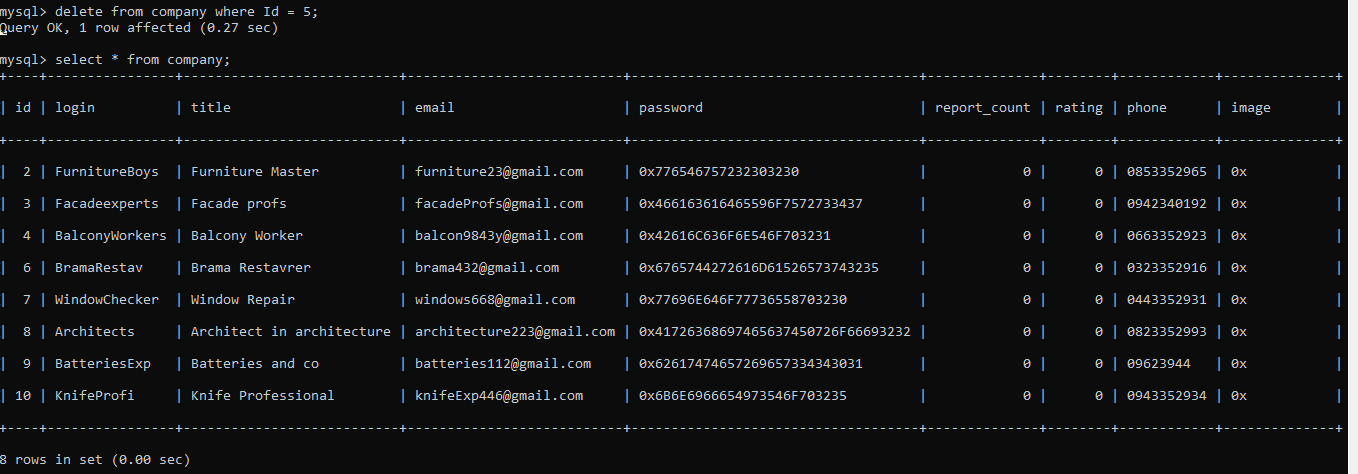
 16



Вибірка даних про компанію відбувається з вказуванням Id.

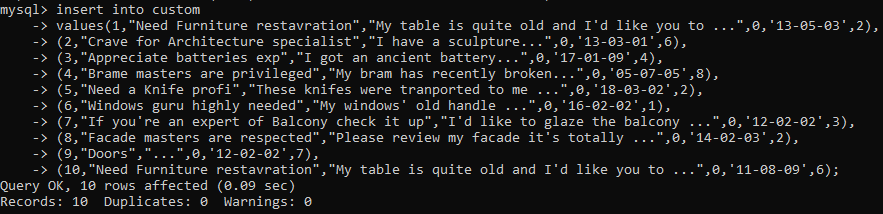


Delete відбувався теж за id.



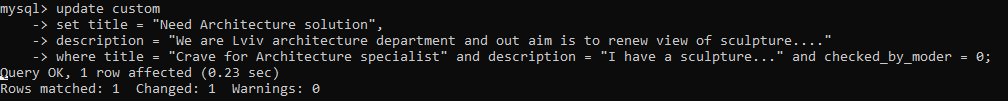
4) Створення,Видалення,Апдейт,селекція даних конкретного замовлення

Створення замовлень

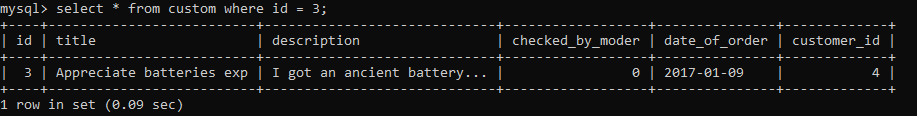


17

Оновлення замовлення,відбувається тільки при передачі даних про попередню його версію.(Так працював серверний метод проекту)



Перехід до конкретного замовлення.

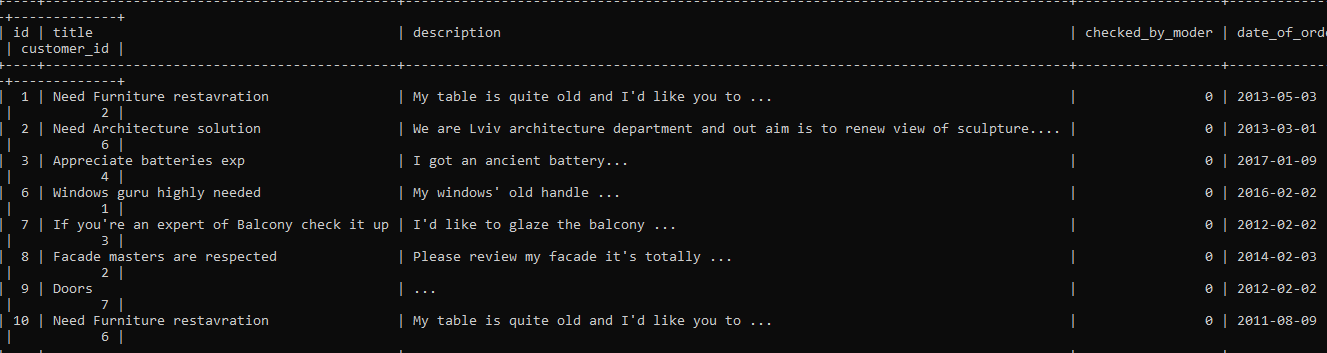


Видалення замовлення можна зробити як зі списку замовлень:



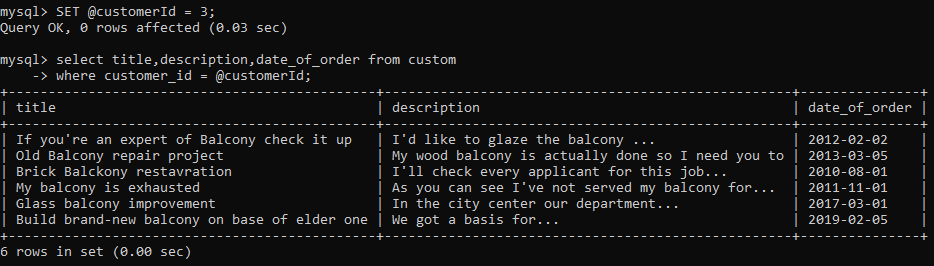
Так і перейшовши до конкретного замовлення і відіславши ріквестом на сервер його id.



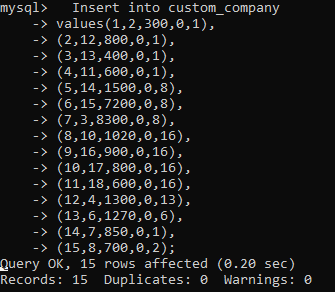


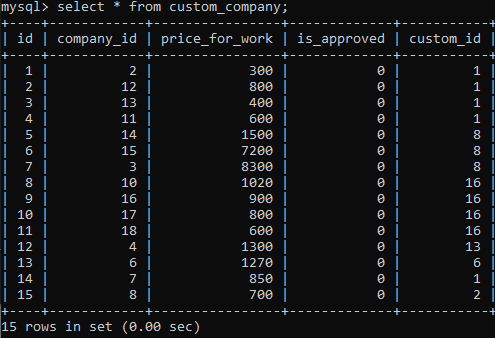
18

5) Вибірка замовлень певного замовника,це дані які будуть показані в списку замовлень користувача.[10]

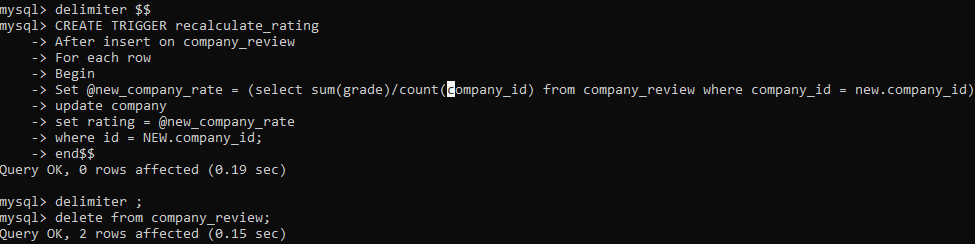


6) Подача заявок на виконання реставраційної роботи.

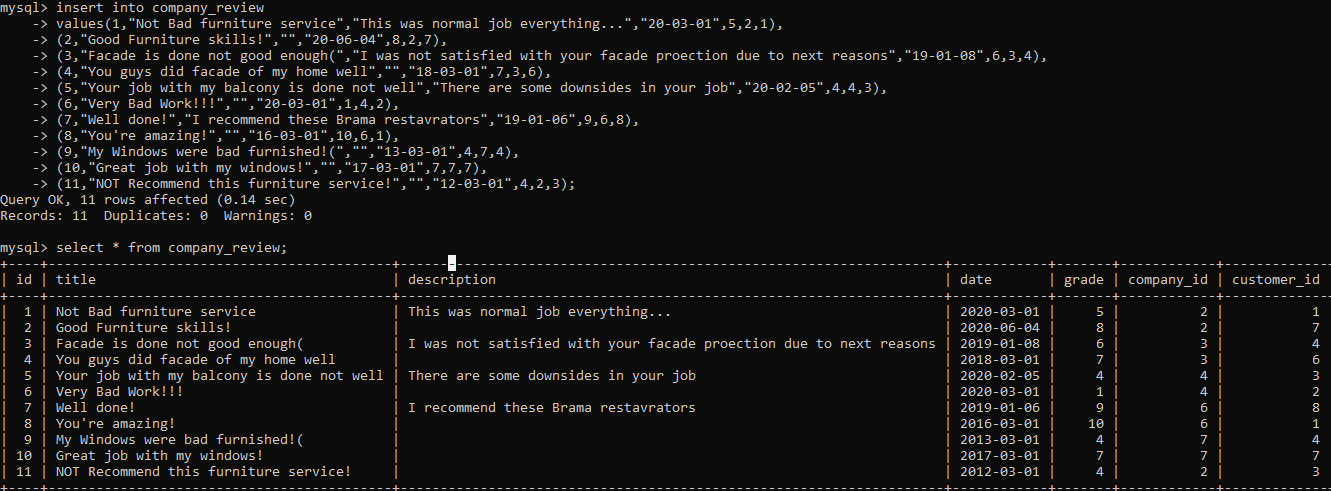


 19

7)Перерахунок рейтингу компанії.

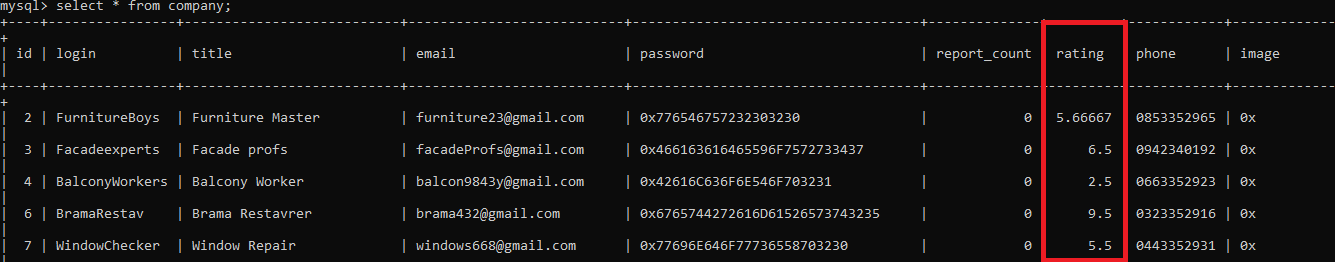


8)Створення відгуків користувачами про компанії.

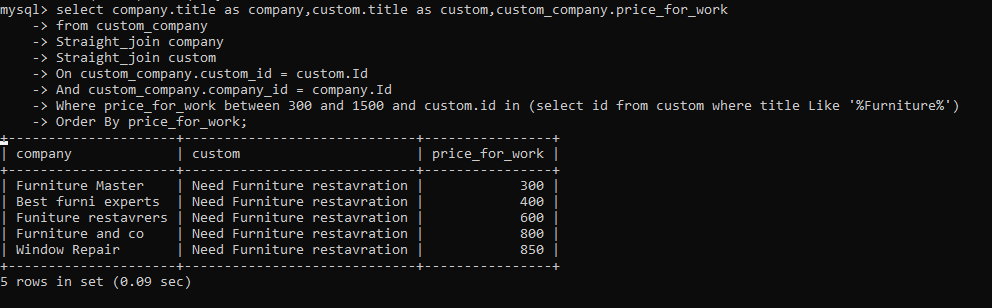


20

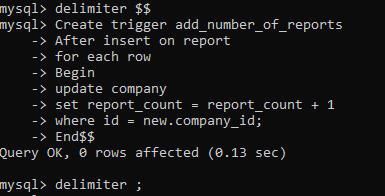
Оцінки Компанії після відгуків



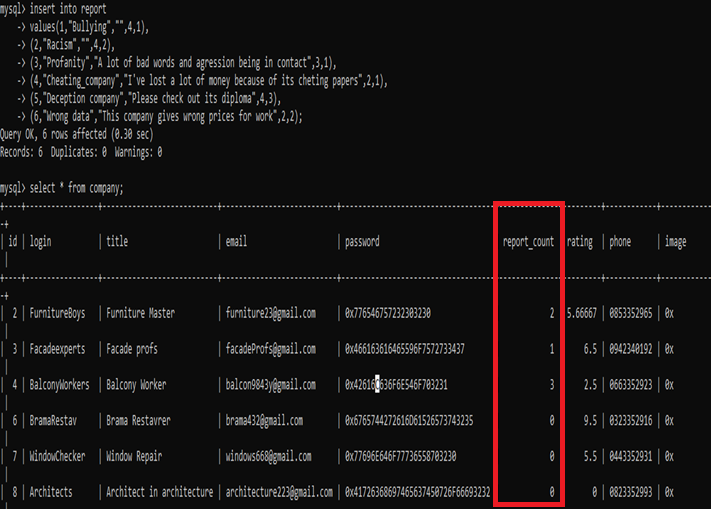
8) Вибірка компаній які подали заяву на конкретне оголошення.(вибірка тих, ціна роботи яких від 300 до 1500 з сортуванням.)[8]



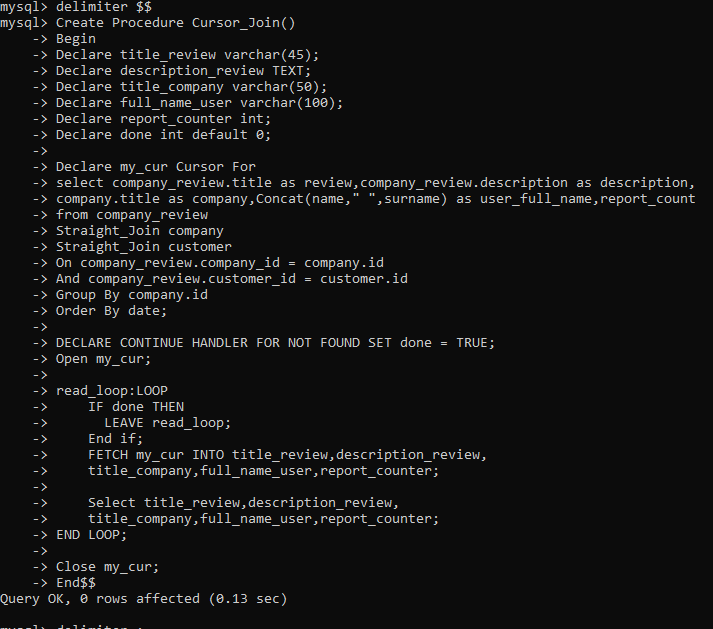
Конфігурування репорту про компанію.Перерахунок кількості репортів.



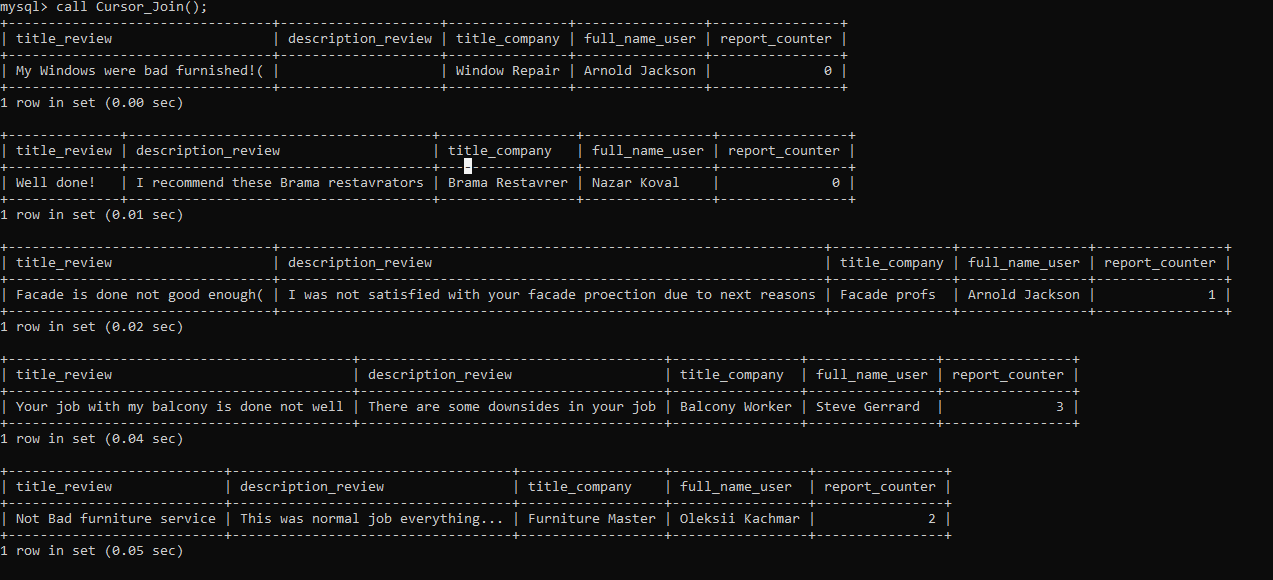
21



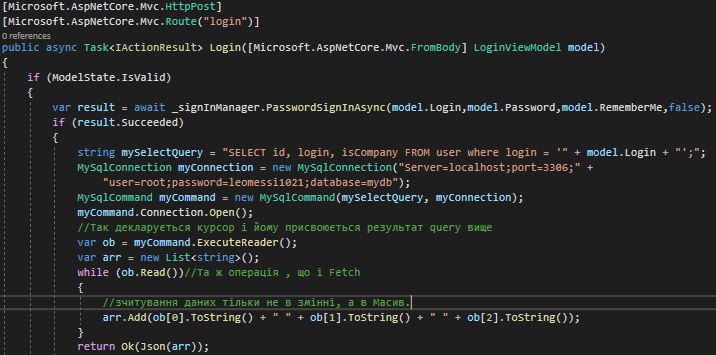
9)Вибірка за допомогою курсору відгуків про конкретну компанію.[9]



Робота процедури з курсором.[11]



Подібний код запиту в нашому проекті на ASP.NET Core

****

Цей код схожий до коду нашої процедури Cursor\_Join, адже ми так само створюєм query,результат роботи якої записується в об’єкт класу MySqlDataReader і цей об’єкт по факту і є операцією declare та open нашого курсора.Ob.Read() –аналог операції Fetch нашого курсору, в цьому коді ми так само циклічно перебираєм результати , записані в курсор.Щоправда, ми додаєм результати в масив, а не записуєм в локальні змінні, як в mysql скрипті.Знищується об’єкт Ob прихованим методом Dispose – це аналог Close для нашого курсора.

24

**Висновки до роботи**

Отже, виконавши цю роботу, я описав основні методи проектування бази даних нашого проекту та сформував відповідні запити для її створення та обробки.Основними сутностями моєї бази даних були:

1)Замовник , описує особу,якій потрібна реставрація

2)Компанія-виконавець, описує компанію,яка виконувала цю реставрацію

3)Замовлення,описувє замовлення,яке формує замовник на своїй сторінці(має виконавця)

4)Репорт\_на\_компанію,характеризує конфігурацію скарги,які складає замовник на компанію(багато-до-багатьох замовника і компанії)

5)Відгуки\_про\_компанію,описує відгук замовника про компанію(багато-до-багатьох замовника і компанії)

6)Компанія\_замовлення,формується з пропертей домовленості між компанією і замовленням(багато-до-багатьох компанії і замовлення).

Основними операціями з моєю базою даних були CRUD-операції з замовником,компанією-виконавцем,замовленням(використав INSERT,UPDATE,DELETE,SELECT до відповідних таблиць);подання компанії на певне замовлення(INSERT до custom\_company);створення скарг на компанію та відгуків про неї(INSERT’и до report,company\_review відповідно);вибірка замовлень,на які підписана компанія(злиттям таблиць company,custom,custom\_company); щоб симулювати код проекту, було створено курсор, в який записувались результати query,що вибирала інформацію про відгуки про компанію.

Також, для підтримки цілісності моєї бд в процесі було створено тригери:хешування паролів користувачів,перерахунок рейтингу компанії при вставці відгуку з оцінкою та підрахунку кількості репортів про компанія,на основі яких адміністратор ресурсу виносить рішення про блокування компанії на нашій сторінці.

25

**Список використаної літератури**

1. Конспект лекцій Мельникова Н.І.
2. Пасічник В.В., Резніченко В.А. Організація баз даних та знань
3. Kroenke D.M., Auer D.J. Database Processing: Fundamentals, Design, and Implementation. 14th ed

##### Powell G. Beginning Database Design

1. <https://habr.com/ru/post/254773/>
2. <http://enisey.name/umk/teis/ch18s04s09.html>
3. <https://www.w3schools.com/sql/default.asp>
4. <https://uk.wikipedia.org/wiki/Join_(SQL)>
5. <https://www.mysqltutorial.org/mysql-cursor/>
6. <http://www1.udel.edu/evelyn/SQL-Class3/SQL3_AggrSyntax.html>
7. https://www.w3resource.com/mysql/mysql-procedure.php

26