Лабораторна робота № 1

Тема: Проектування реляційної бази даних

1 Завдання для роботи на *Online*-занятті

Припустимо, що вже існує *UML*-діаграма концептуальних класів, які містять наступні сутності предметної області, пов'язаної з діяльністю торгової компанії:

- співробітник (прізвище, посада, дата зарахування, зарплата, комісійні);
- менеджер, який є співробітником з додатковими функціями управління групою співробітників;
- підрозділ, в якому працюють співробітники (назва);
- місцезнаходження підрозділу (місто).

Також, припустимо, що вже існує реляційна модель БД, яка відповідає *UML*-діаграмі концептуальних класів.

На етапі супроводу у замовника з’явилася додаткова вимога до включення нових сутностей:

– *закупка продукції (найменування, ціна закупки);*

– *продаж продукції (найменування, ціна продажи).*

Завдання 1.1 Перепроектування *UML*-діаграми концептуальних класів

1.1.1. Визначити імена класів.

1.1.2. Визначити імена атрибутів класів.

1.1.3. Визначити зв’язки між класами. Для зв’язків-асоціацій визначити:

– назви;

– кратність;

– можливість агрегації.

Розширити *UML*-діаграму на аркуші паперу або в будь-якому редакторі, наприклад, http://draw.io

Завдання 1.2 Перепроектування реляційної моделі БД

1.2.1 На основі нової *UML*-діаграми концептуальних класів розширити реляційну модель БД на аркуші паперу або в будь-якому редакторі, наприклад, в http://draw.io

2 Індивідуальні завдання для роботи після завершення *Online*-заняття

Нехай задано предметну область з приблизним описом сутностей у відповідності з вашим варіантом з додатку.

Завдання 2.1 Уточення опису предметної області

Для кожної сутності необхідно самостійно визначити не менше 2 атрибутів.

Також потрібно врахувати наявність не менше ніж одного атрибуту символьного типу, типу «дата» та числового типу.

Завдання 2.2 Проектування *UML*-діаграми концептуальних класів

2.2.1 Визначити імена класів.

2.2.2 Визначити імена атрибутів класів.

2.2.3 Забезпечити зв’язки між класами. Для існуючих зв’язків-асоціацій визначити:

– назви;

– кратність;

– можливість агрегації.

Спроектувати *UML*-діаграму на аркуші паперу або в будь-якому редакторі, наприклад, http://draw.io

2.2.4 Зберегти результат проектування у файлі з ім'ям *Surname_uml.jpg, де Surname* – ваше прізвище.

Завдання 2.3 Проектування реляційної моделі БД

2.3.1 На основі *UML*-діаграми концептуальних класів спроектувати реляційну модель на аркуші паперу або в будь-якому редакторі, наприклад, в http://draw.io

2.3.2 Зберегти результат проектування у файлі з ім'ям *Surname_rel.jpg*

Завдання 2.4 Робота з *GitHub-project*

Для контролю виконання завдань з лабораторної роботи викладачами створено:

– *Git*-репозиторій, який розміщено на веб-сервісі *GitHub*;

– *GitHub-project*, який зв'язано з *Git*-репозиторієм.

2.4.1 В *Git*-репозиторії завдання з лабораторної роботи розміщено в *Issue* з назвою «*tasks-of-laboratory-work-1*» та посиланням на файл із завданням - https://drive.google.com/file/d/1GcDoUTSWc_8NiglgBttLlxRXcxXaxWW0/view

Це *Issue* можна побачити через *GitHub-project* на *Scrum*-дошці «*Todo*» у вигляді *Scrum*-картки.

Розпочинаючи роботу над завданнями, необхідно:

1) змінити статус *Issue* з «*Todo*» на «*In progress*», автоматично перевівши *Scrum*-картку з цим *Issue* на *Scrum*-дошку «*In progress*»;

2) у своєму *Git*-репозиторію на веб-сервісі *GitHub* або в локальній *Git*-копії репозиторію створити нову гілку *Git*-репозиторію з назвою, співпадаючою з назвою *Issue*, наприклад, «*tasks-of-laboratory-work-1*».

На веб-сервісі *GitHub* гілку можна створити безпосередньо на сторінці *Issue*, використовуючи посилання «*Create a branch*» у правому розділі «*Development*».

Після створення гілки перейти до цієї гілки для створення оновлень файлів *Git*-репозиторію.

2.4.2 В новій гілці *Git*-репозиторію створити каталог з назвою «*1.1-ConceptualClasses*» (кнопка *«Add file» - «Create new file»*), при створені якого одночасно створити файл *README.md* з першим рядком «*UML-diagram of Conceptual Classes*» зі стилем «Заголовок 3-го рівня» мови розмітки *Markdown* (три символи решітка ###).

Після завершення редагування файлу виконати фіксацію змін:

– кнопка *«Commit new file»* – якщо була перше оновлення файлу;

– кнопка *«Propose changes»* – якщо було повторне оновлення файлу.

2.4.3 Розмістити в каталозі «*1.1-ConceptualClasses*» на *GitHub* файл *Surname_uml.jpg* (кнопка *«Add file» - «Upload files»)*

2.4.4 Відобразити файл *Surname_uml.jpg* через його *URL*-посилання на *GitHub* у файлі *README.md* каталогу «*1.1-ConceptualClasses*» з використанням *MarkDown*-форматування.

Приклад *MarkDown*-форматування:

```
![](https://github.com/OP-NC-EduCentre/student/blob/Laboratory_Work_1/1.1-ConceptualClasses/UML-ConceptClasses.jpg)
```

2.4.5 Створити ще один каталог з назвою «*1.2-RelationDBSchema*», при створені якого одночасно створити файл *README.md* з першим рядком «*Relation DataBase Schema*» зі стилем «Заголовок 3-го рівня» мови розмітки *Markdown*.

2.4.6 Розмістити в каталозі «*1.2-RelationDBSchema*» файл *Surname_rel.jpg*

2.4.7 Відобразити файл *Surname_rel.jpg* у файлі *README.md* каталогу «*1.2-RelationDBSchema*» з використанням *MarkDown*-форматування.

Після завершення редагування файлу виконати фіксацію змін:

– кнопка *«Commit new file»* – якщо було перше оновлення файлу;

– кнопка *«Propose changes»* – якщо було повторне оновлення файлу.

2.4.8 Для злиття всіх оновлень (*Commit*), які було зроблено у новій *Git*-гілці, до основної гілки *Git*-репозиторію (*main*-гілка) необхідно виконати запит *Pull Request (*кнопка *«Compare & pull request»* розділу *Pull request)*, розпочавши процес *Code Review*.

Під час створення *Pull Request* необхідно вказати:

– *Reviewers* (рецензенти) = *Oleksandr Blazhko, Maria Glava*, щоб в процесі *Code Review* в якості рецензентів виступив викладач, який є власником репозиторію (*Oleksandr Blazhko*), або викладач, який є модератором репозиторію (*Maria Glava*);

– *Labels* (мітка) = *enhancement (New feature or request)*;

– *Projects* = посилання на *GitHub-project*.

Процес *Code Review* може мати два варіанти завершення:

1) викладачем вказано коментарі щодо рішень, на які необхідно оновити відповідні файли;

2) викладач схвалює рішення (*Approve*), після чого дозволяє злити оновлення до основної гілки та закрити *Pull Request*.

Схвалення рішення за *Pull Request* переводить його *Scrum*-картку до *Scrum*-дошки «*In Progress*».

Злиття оновлення до основної гілки переводить *Scrum*-картку *Pull Request* до *Scrum*-дошки «*Done*».

2.4.9 Після завершення процесу *Code Review* викладач закриє *Issue*, завершуючи процес виконання завдань з лабораторної роботи.

Додаток. Варіанти завдань

1. Опис сутностей предметної області «Студентський гуртожиток»:
– кімнати та їх характеристики;
– оснащення кімнат (меблеве та технічне).
2. Опис сутностей предметної області «Автостоянка»:
– власники: ПІБ, адреса, телефон;
– машини: марка, номер, колір, рік випуску.
3. Опис сутностей предметної області «Фітнес-клуб»:
– зали та їх оснащення;
– тренери та інші співробітники.
4. Опис сутностей предметної області «Ресторан»:
– меню з зазначенням основних інгредієнтів, часу приготування та вартості;
– столики та їх бронювання постійними клієнтами.
5. Опис сутностей предметної області «Кабельне телебачення»:
– клієнти компанії;
– рахунки та їх сплати.
6. Опис сутностей предметної області «Молочний кіоск»:
– інформація про товар;
– дані по продажах.
7. Опис сутностей предметної області «Оператор мобільного зв’язку»:
– тарифи та їх характеристики;
– контрактні клієнти.
8. Опис сутностей предметної області «Меблевий магазин»:
– товари та їх характеристики з зазначенням наявності або термінів виготовлення;
– клієнти та їх замовлення.
9. Опис сутностей предметної області «Ріелторські послуги»:
– інформація про нерухомість;
– інформація про власників.
10. Опис сутностей предметної області «Ремонт побутової техніки»:
– інформація про техніку на ремонті;
– інформація про власників.
11. Опис сутностей предметної області «Автобусний парк»:
– інформація про автобуси;
– інформація про водіїв.

12. Опис сутностей предметної області «Прокат автомобілів»:

– інформація про автомобілі;

– інформація про прокат.

13. Опис сутностей предметної області «Картотека поліклініки»:

– дані про пацієнтів;

– відомості про проведені обстеження.

14. Опис сутностей предметної області «Аптечна мережа»:

– ліки;

– інформація про поставки ліків.

15. Опис сутностей предметної області «Оптовий склад»:

– продовольчі товари,

– продажі товару.

16. Опис сутностей предметної області «Салон краси»:

– перелік послуг;

– запис на процедури.

17. Опис сутностей предметної області «Прокат туристичного спорядження»:

– характеристика наявного спорядження;

– прокат спорядження.

18. Опис сутностей предметної області «Станція технічного обслуговування»:

– марки авто, що обслуговуються;

– каталог запасних частин.

19. Опис сутностей предметної області «Театри міста»:

– театри міста;

– запропоновані вистави.