УДК 004.65

Бояр Є.М.[1], Рябенко А.Є.[2]

АВТОМАТИЗАЦІЯ ВЗАЄМОДІЇ З БАЗАМИ ДАНИХ ЗА ДОПОМОГОЮ ORM У PYTHON

Вступ. Python є однією з найпопулярніших мов програмування у світі. Він широко використовується у веб-розробці, аналізі даних, машинному навчанні та багатьох інших сферах. Одним із ключових аспектів веб-розробки є робота з базами даних. Прямі SQL-запити, хоча й надають повний контроль, можуть бути складними в реалізації та супроводі, особливо при роботі з великими проектами. Саме тому популярними стають ORM (Object-Relational Mapping) — бібліотеки, що дозволяють працювати з базами даних через об'єктно-орієнтоване програмування.

Сьогодні Python є однією з провідних мов для розробки бекенду веб-додатків. Популярні фреймворки, такі як Django та Flask, широко використовують ORM-рішення, що полегшують взаємодію з базами даних.

Основний матеріал. ORM дає змогу розробникам взаємодіяти з даними через класи та об'єкти, а не через SQL-запити. Це підвищує безпеку, зменшує кількість коду, необхідного для виконання операцій із базою даних, та дозволяє абстрагуватися від конкретної СУБД (системи управління базами даних). Таким чином, ORM забезпечує:

- спрощене управління даними;
- можливість переносимості коду між різними базами даних;
- зменшення помилок, пов'язаних із синтаксисом SQL-запитів;
- кращу інтеграцію з об'єктно-орієнтованими підходами до програмування.

[1] студент групи КНТ-811, НУ «Запорізька політехніка» та Університету Сент-Ендрюса

[2] доцент кафедри САОМ, НУ «Запорізька політехніка», к. ф.-м. н.

Головне завдання ORM — автоматизувати взаємодію з базами даних, зробити її більш гнучкою, безпечною та ефективною. Для цього ORM-бібліотеки надають широкий спектр можливостей:

1. Автоматичне створення таблиць та схем баз даних

ORM дозволяє визначати моделі даних у вигляді класів Python, а потім автоматично створювати відповідні таблиці у базі. Це спрощує процес розробки та змінення структури бази даних без необхідності писати SQL-код.

2. Виконання CRUD-операцій

CRUD (Create, Read, Update, Delete) — це основні операції з базою даних. Використовуючи ORM, можна легко створювати, отримувати, оновлювати та видаляти записи у таблицях за допомогою об'єктів Python, без необхідності працювати з SQL-запитами. Для створення записів у базі даних ORM надає методи, які дозволяють додавати нові записи, використовуючи Python-словники. У нашій реалізації метод insert_entries класу BaseModel дозволяє додавати кілька записів одночасно.

3. Підтримка міграцій

Міграції є важливим компонентом сучасних ORM, оскільки вони дозволяють автоматично оновлювати структуру бази даних відповідно до змін у моделях. Це особливо корисно у великих проектах, де структура бази даних може часто змінюватися.

4. Оптимізація запитів

Однією з ключових переваг ORM є можливість оптимізації запитів до бази даних. Це дозволяє зменшити кількість SQL-запитів, підвищити продуктивність програми та уникнути зайвого навантаження на базу даних. У нашій ORM можна реалізувати кілька механізмів оптимізації, які широко використовуються в сучасних ORM-бібліотеках.

5. Розширюваність

Сучасні ORM-бібліотеки дозволяють додавати власну логіку для обробки запитів, розширювати їх функціонал за допомогою плагінів та middleware, а також реалізовувати додаткові методи для специфічних задач. Нижче наведено кілька аспектів розширюваності та приклади їх реалізації:

Висновки. Розробка власної ORM-бібліотеки може стати складним, але цікавим викликом. Це дозволяє глибше зрозуміти принципи роботи з базами даних, оптимізації запитів та об’єктно-орієнтованого підходу. Завдяки ORM, програмісти отримують потужний інструмент для взаємодії з базами даних, що спрощує розробку, підвищує безпеку коду та зменшує ризик помилок.

Враховуючи зростання популярності Python та необхідність ефективного управління даними, використання ORM стає важливим

аспектом сучасної розробки. Розуміння основних принципів роботи ORM дозволяє створювати масштабовані, ефективні та зручні у підтримці програми.