Технічні науки

УДК 004.054

Токарський Андрій Олегович

студент

Національного технічного університету України

«Київський політехнічний інститут імені Ігоря Сікорського»

Токарский Андрей Олегович

студент

Национального технического университета Украины

«Киевский политехнический институт имени Игоря Сикорского»

Tokarskyi Andrii

Student of the

National technical university of Ukraine

«Igor Sikorsky Kyiv Polytechnic Institute»

КОНЦЕПЦІЯ ОБ'ЄКТНО-ОРІЄНТОВАНИХ БАЗ ДАНИХ
КОНЦЕПЦИЯ ОБЪЕКТНО-ОРИЕНТИРОВАННЫХ БАЗ ДАННЫХ
THE OBJECT-ORIENTED DATABASES CONCEPT

Анотація: Використання об'єктно-орієнтованих баз даних в інформаційних системах.

Ключові слова: SQL, ООБД, Дані, ООП, База данных.

Аннотация. Использование обёектно-ориентированных баз данных в информационных системах.

Ключевые слова: SQL, ООБД, Дані, ООП, База данных.

Summary. Use of object-oriented databases in information systems.

Key words: SQL, OODB, Data, OOP, Database.

На початку 1980-х років почалися перші дослідження і розробка концепцій об'єктно-орієнтованих баз даних.

Об'єктно-орієнтовані бази даних - бази даних, в яких інформація представлена у вигляді об'єктів, як в об'єктно-орієнтованих мовах програмування [1].

Причиною появи систем об'єктно-орієнтованих баз даних була потреба в більш адекватному уявленні і моделюванні сутностей реального світу, оскільки ООБД забезпечують набагато більш розвинену модель даних, ніж традиційні реляційні бази даних.

Об'єктно-орієнтовані бази даних зазвичай рекомендовані для тих випадків, коли потрібна високопродуктивна обробка даних, що мають складну структуру.

Основні труднощі об'єктно-орієнтованого моделювання даних є наслідком того, що такого розвиненого математичного апарату, на який могла б спиратися загальна об'єктно-орієнтована модель даних, не існує.

Розгляд особливостей ООБД привели до визначення ООБД, яке було представлено на Першій міжнародній конференції з дедуктивним і об'єктно-орієнтованим баз даних у вигляді маніфесту 1989 року. У ньому представлені можливості об'єктно-орієнтованих баз даних.

Обов'язкові можливості ООБД:

1. Підтримка складних об'єктів

   Механізм складних об'єктів дозволяє об'єкту мати атрибут, що можуть теж бути об'єктами.
2. Ідентифікація об'єктів
3. Інкапсуляція
4. Підтримка структур і класів [2]
5. Наслідування і поліморфізм

   Наслідування - одне з базових понять в об'єктно-орієнтованому програмуванні. Пов'язано з тим, що клас може мати клас-спадкоємця, який має ті ж властивості, що і базовий клас, але також і мати нові властивості.

6. Паралелізм
7. Відновлення
8. Спеціальна система запитів
9. Можливість додавання нових типів даних

Об'єктно-орієнтовані СУБД

В принципі, ООСУБД - це об'єктно-орієнтована база даних, що надає можливості СУБД об'єктам, що були створені з використанням об'єктно-орієнтованої мови програмування.

Прикладні програми, що використовують об'єктно-орієнтовану базу даних, пишуться на об'єктно-орієнтованої мови програмування, а можливості СУБД існують завдяки спеціальному API, в якому є клас бази даних або базовий клас Persistent, в яких реалізовані функції для роботи з даними.

ООСУБД дозволяє працювати з об'єктами баз даних так само, як з об'єктами в програмуванні на об'єктно-орієнтованих мовах. ООСУБД розширює мови програмування, прозоро вводячи довгострокові дані, управління паралелізмом, відновлення даних, асоціативні запити й інші можливості. Деякі об'єктно-орієнтовані бази даних розроблені для щільної взаємодії з такими об'єктно-орієнтованими мовами програмування як Python, Java, C #, Visual Basic .NET, C ++, Objective-C і Smalltalk; інші мають свої власні мови програмування. ООСУБД використовують точно таку ж модель, що і об'єктно-орієнтовані мови програмування [3].

Порівняння концепцій ООБД і реляційних БД

Таблиця 1

Співвідношення між базовими поняттями ООБД і РБД [4]

ООБД	Реляційні БД
Об'єкт	Рядок таблиці
Поле	Стовпчик таблиці
Ієрархія класів	Схема бази даних
Наслідування	Один з окремих випадків відносини один-до-одного, де основний ключ таблиці-потомка є одночасно зовнішнім ключем на основний ключ таблиці-предка
Метод	-
Поліморфізм	-
Інкапсуляція	-

У таблиці 1 вказані співвідношення між базовими поняттями, що використовуються у об'єктно-орієнтованих та реляційних базах даних.

Приклад використання ООБД

Припустимо, перед нами стоїть завдання розробки інформаційної системи для деякого підприємства. Підприємство працює з клієнтами, тому база даних нашої інформаційної системи зберігає дані про клієнтів. Також з клієнтами працюють співробітники, кожен співробітник даної інформаційної системи може робити маніпуляції з інформацією про клієнтів від свого імені. Тому в програмі створена наступна ієрархія класів. У базовому класі Person зберігається загальна інформація про персону: ПІБ, список адрес електронної пошти. Його потомки - класи Client та Employee. У першому зберігається якась клієнтська інформація, у другому - дані для входу. На рисунку 1 зображена архітектура класів цієї інформаційної системи.

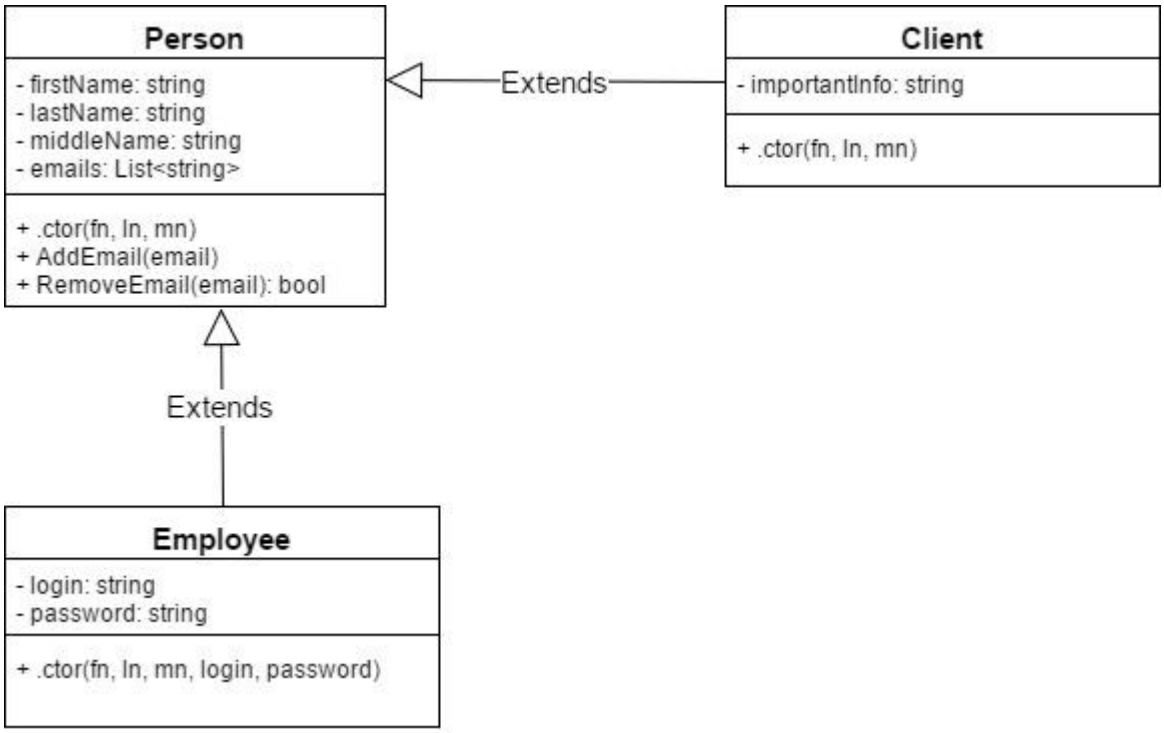

Рис. 1. UML-діаграма класів інформаційної системи

У випадку з використанням реляційних баз даних в нашій інформаційній системі схема бази даних виглядала б так, як на рисунку 2. Наслідування відповідає відносинам один-до-одного між таблицями. Також виникла необхідність додати ще одну таблицю Email з складовим основним ключем, першою частиною якого є зовнішній ключ на таблицю Person, а другий – сама електронна адреса.

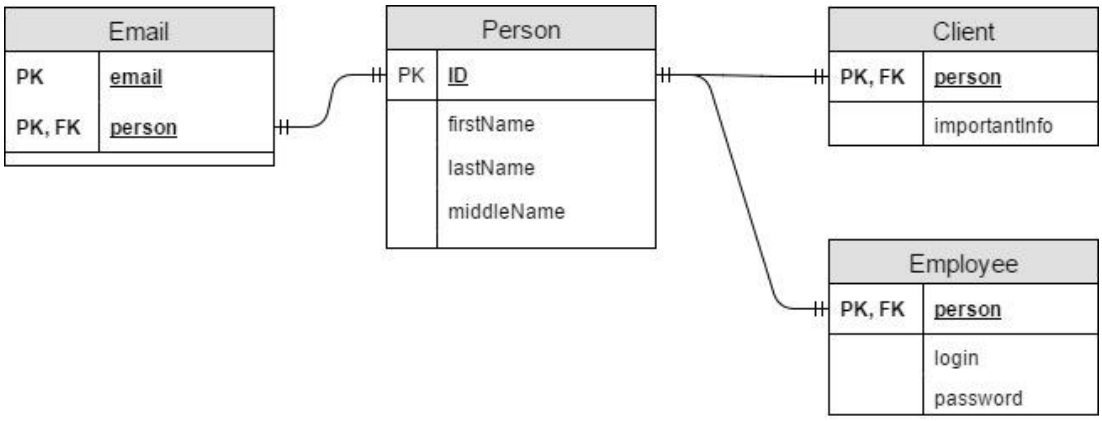

Рис. 2. Реляційне представлення даних інформаційної системи

Тепер порівняємо, яким чином відбувається відправка запитів зі сторони користувача.

Для вивантаження інформації про клієнта потрібно пов'язати таблиці Client і Person із допомогою ключового слова JOIN, і з допомогою конструкції WHERE визначити умови, які повинні виконуватись для клієнта, інформацію про якого ми шукаємо (рисунок 3). У нашому випадку ми шукаємо клієнта по його ідентифікатору.

```
1. SELECT Person.firstName,
2.        Person.lastName,
3.        Person.middleName,
4.        Client.importantInfo
5.   FROM Person
6.        JOIN Client
7.          ON Client.person = Person.id
8.  WHERE Person.id = 25;
```

Рис. 3. Вивантаження даних з допомогою SQL-запиту

Додавання нових даних здійснюється із допомогою INSERT-запитів, як зорбажено на рисунку 4. Для того, щоб додати нового клієнта, необхідно спочатку додати нову запис в таблицю Person. Потім в таблицю Client додаємо нову запис, ідентифікатором якої є ідентифікатор щойно доданої записи в таблиці Person. Виконання цих запитів повинне бути нерозривним, тому вони обернуті в обну транзакцію.

Недолік даного підходу у збереженні інформації полягає у тому, що дані в таблицях Person і Client є не нерозривними. Можливо, наприклад, видалити запис із таблиці Client і не видалити запис із цим ідентифікатором із таблиці Person.

```
1. BEGIN TRANSACTION;
2. INSERT INTO Person (firstName, lastName, middleName)
3.      VALUES ('Andrew', 'Tokarskiy', NULL);
4. INSERT INTO Client (person, importantInfo)
5.      VALUES ((SELECT MAX(Person.id) FROM Person), '!!Some info!!');
6. COMMIT;
```

Рис. 4. Додавання даних з допомогою SQL-запитів

У випадку із необхідністю видалення запису із таблиці Person потрібно видалити або змінити всі записи із інших таблиць, що на ідентифікатор цієї людини посилаються, на рисунку 5 продемонстрована операція видалення клієнта. Всі операції видалення повинні бути обернуті в транзакцію.

```
1. BEGIN TRANSACTION;
2. DECLARE @id INT;
3. SET @id = 25;
4. DELETE FROM Email WHERE person = @id;
5. DELETE FROM Client WHERE person = @id;
6. DELETE FROM Person WHERE id = @id;
7. COMMIT;
```

Рис. 5. Видалення даних з допомогою SQL-запитів

Як бачимо, зберігання таких даних у реляційних базах даних є достатньо складним для розробника. Розглянемо, яким чином ці ж дані зберігаються у об'єктних базах даних.

Операція вивантаження клієнтів в ООБД відрізняється від операції вивантаження клієнтів в реляційних БД тим, що немає необхідності використовувати JOIN для зв'язування сутностей. Той факт, що Client являється потомком Person, базі даних вже відомий. На рисунку 6 продемонстроване вивантаження даних із ООБД.

```
1. Client GetClientById(Database db, int id)
2. {
3.     return db.AsQueryable<Client>()
4.              .First(person => person.Id == id);
5. }
```

Рис. 6. Вивантаження даних з допомогою ООБД

Це ж стосується і додавання нової інформації, як видно із рисунка 7. Також немає необхідності створювати в базі даних нову сутність для зберігання у ній електронних адресів. Вони вже зберігаються у полі emails, що має тип даних List<string>.

```
1. var client = new Client
2. {
3.     FirstName = "John",
4.     LastName = "Smith",
5.     MiddleName = null,
6.     ImportantInfo = "!!Some info!!"
7. };
8. client.AddEmail("smith@email.com");
9.
   db.Persist(client);
```

Рис. 7. Додавання даних з допомогою ООБД

Видалення об'єкта із об'єктно-орієнтованої бази даних є дуже простим. Достатньо лише вказати об'єкт, що потрібно видалити.

Література

1. Объектно-ориентированные базы данных - основные концепции, организация и управление: краткий обзор. – Режим доступа: http://citforum.ru/database/articles/art_24.shtml/. – Дата доступа: 11.07.2017
2. Thomas A. Mueck. Index Data Structures in Object-Oriented Databases / Thomas A. Mueck – Luxembourg : Springer Science & Business Media. – 1997
3. Architecture In Object Oriented databases. – Режим доступа: http://citeseerx.ist.psu.edu/viewdoc/download?rep=rep1&type=pdf&doi=10.1.1.116.434/. – Дата доступа: 29.06.2017
4. David Maier. Object-Orienter Database Theory. An Introduction / David Maier – Muenchen : TU Muenchen. – 2001