

Василь Литвин

ОБ'ЄКТНО-ОРІЄНТОВАНИЙ ПІДХІД ДО ПРОЕКТУВАННЯ БАЗ ДАНИХ

Запропонований більш ніж 20 років тому об'єктний підхід до розробки прикладних програм ніс у собі революційні зміни: зокрема, можливість повторного використання даних, підвищення продуктивності, значну гнучкість процесів проектування систем. Не прийняття об'єктів в їх першому поданні обумовилось бажанням розробників піти шляхом узагальнення раніш напрацьованих підходів та методів.

Нині об'єктний підхід використовується для побудови різнотипних інформаційних систем: зокрема, систем автоматизованого проектування, навчання, баз даних, контролю систем програмного забезпечення, аналіз, операційних систем, інтерфейсних модулів. Ця методологія дозволяє успішно поборювати фактор складності, який присутній при побудові великих проектів та реальних інформаційно-комп'ютерних систем.

Перед створенням об'єктних типів в БД, слід проаналізувати поставлену мету їх використання. Нашим завданням є створення об'єктних типів для побудови складних БД, що опрацьовують дані з максимальним ступенем подібності до предметів реального світу. Потім ми та інші проектувальники можуть використовувати ці об'єкти в таблицях іншої БД.

Основним принципом ОО підходу є умова "об'єктом може бути все, що завгодно". Деякі об'єкти є примітивні і не змінюються, наприклад, числа і символічні рядки. Згідно з традиційною термінологією такі об'єкти відповідають значенням. Інші об'єкти, які задаються користувачем і змінюються, побудовані складнішим чином, наприклад, СЛУЖБОВЦІ, ВІДДІЛ і т.д. Ці більш складні об'єкти відповідають змінним з довільною внутрішньою складністю, тобто такі об'єкти можуть містити будь-яку кількість типів даних, які присутні в звичайних мовах програмування, і конструкторів типів - чисел, рядків, списків, масивів, стеків і т.д.

Наступні приклади ілюструють роботу з об'єктами і те як ми можемо використати їх в проектуванні БД. Багато розробників використовують об'єктні типи для побудови простих БД, які потім

можна використати для побудови реляційної БД, що робиться досить просто. Наприклад, наступний запис створює тип адреса ADDRESS_TYPE:

```
CREATE OR REPLACE TYPE address_type AS OBJECT (  
    street1 VARCHAR2(50),  
    street2 VARCHAR2(50),  
    city VARCHAR2(50),  
    state VARCHAR2(25),  
    zipcode VARCHAR2(10),  
    country VARCHAR2(50));
```

Тому що специфікація ADDRESS_TYPE не містить методів специфікації, то тіло в такого об'єкта відсутнє. Коли специфікація об'єктного типу не містить ні методів специфікації, ні атрибутів, то це є незакінчений об'єктний тип (дефектний). Інколи треба оголошувати дефектні об'єкти перед тим як ми ще точно не знаємо, що ми хочемо зробити. Так як потім ми можемо побудувати залежні об'єкти навколо дефектного.

Коли нам треба визначити адресу в реляційній БД, ми можемо використати об'єкт ADDRESS_TYPE, який автоматично погоджується із загальною директивою:

```
CREATE TABLE sales.customers (  
    id INTEGER PRIMARY KEY,  
    last_name VARCHAR2(50),  
    first_name VARCHAR2(50),  
    address pub.address_type,  
    ...
```

Об'єктно-орієнтована розробка програм є набагато вигідніша, тому що потребує менше затрат на побудову інформаційних систем. Об'єкти можна використовувати повторно, не задумуючись над тим як вони працюють. Це є перспективний етап на шляху розвитку технологій керування БД.

1. Г. Буч Объектно-ориентированное проектирование с примерами применения. - Киев: Диалектика, 1992. 2. Objects in high gear. Oracle magazine, january/february. 1997. P. 84-96. 3. Extending Oracle 8 with objects. Oracle magazine, september/october. 1997. P. 123-128. 4. Date C.J. An Introduction to Database Systems. - Addison-Wesley Longman, Inc., 1995.