Бази даних та інформаційні системи

Тема 17.Тригери

СумДУ, каф. КН

2020

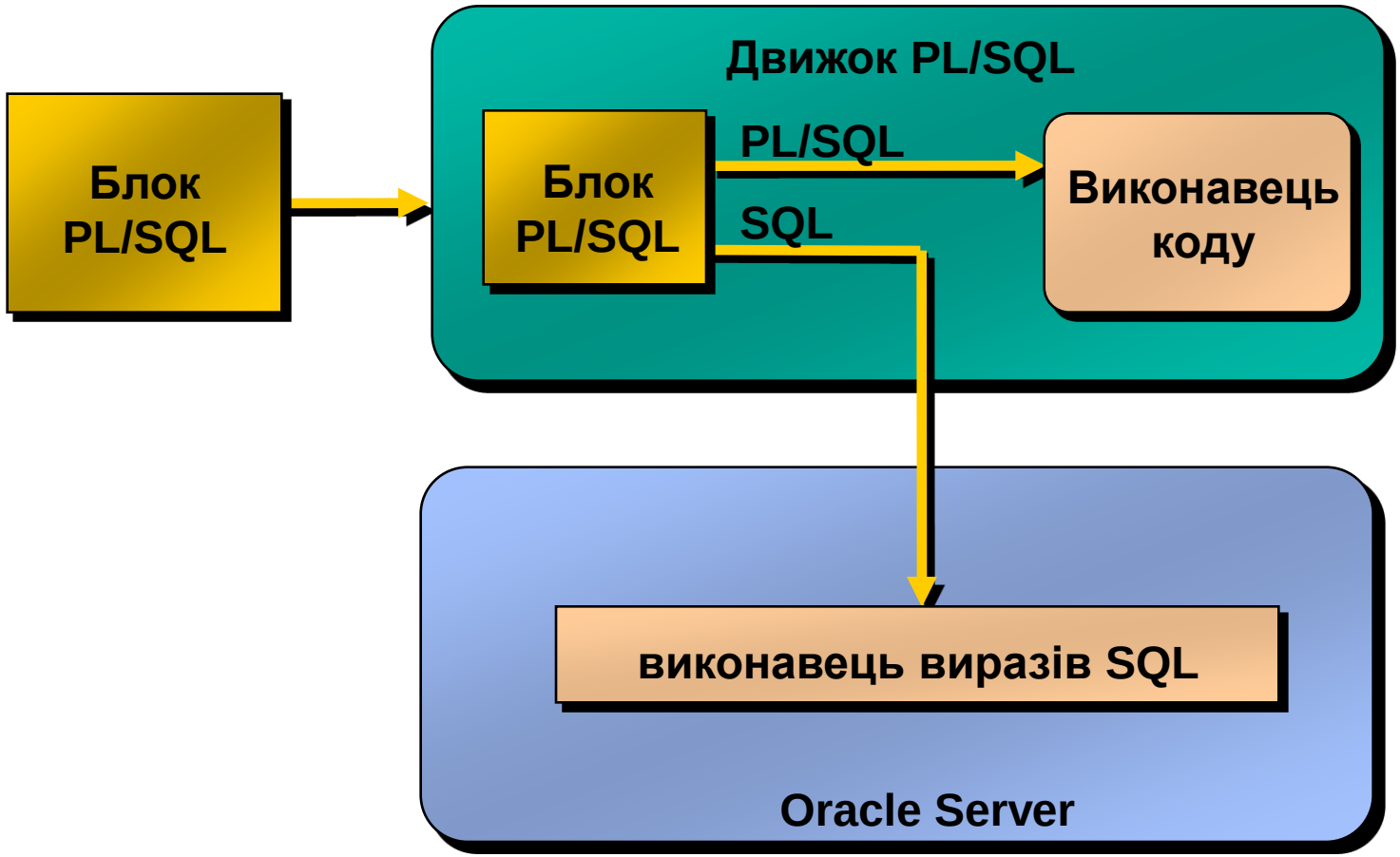
Дещо про PL / SQL

PL / SQL означає Procedural Language / **SQL** (процедурна мова / **SQL**).

PL / SQL розширює можливості **SQL**, додає в нього такі конструкції процедурних мов, як:

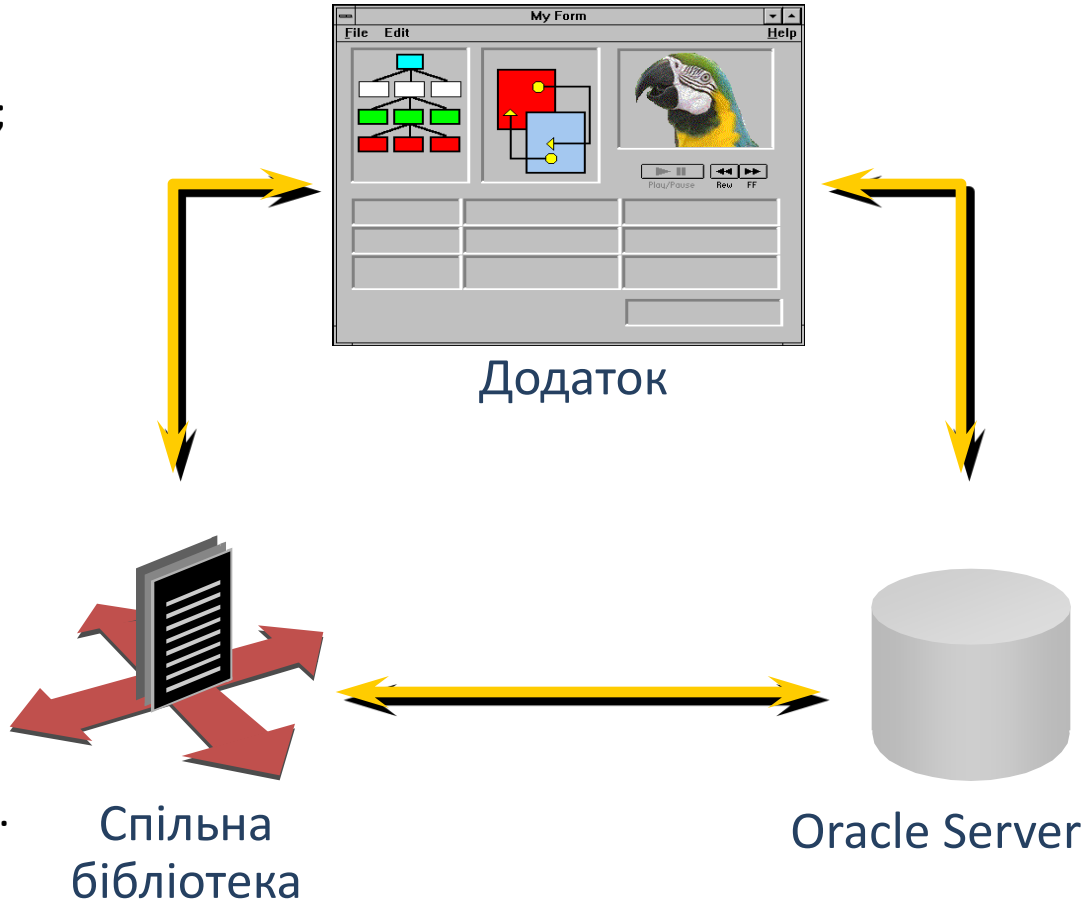
- Змінні і типи даних (як визначені попередньо , так і ті, що визначаються користувачами);
- Керуючі структури, такі як умовні оператори та цикли;
- Процедури і функції.

Середовище PL/SQL



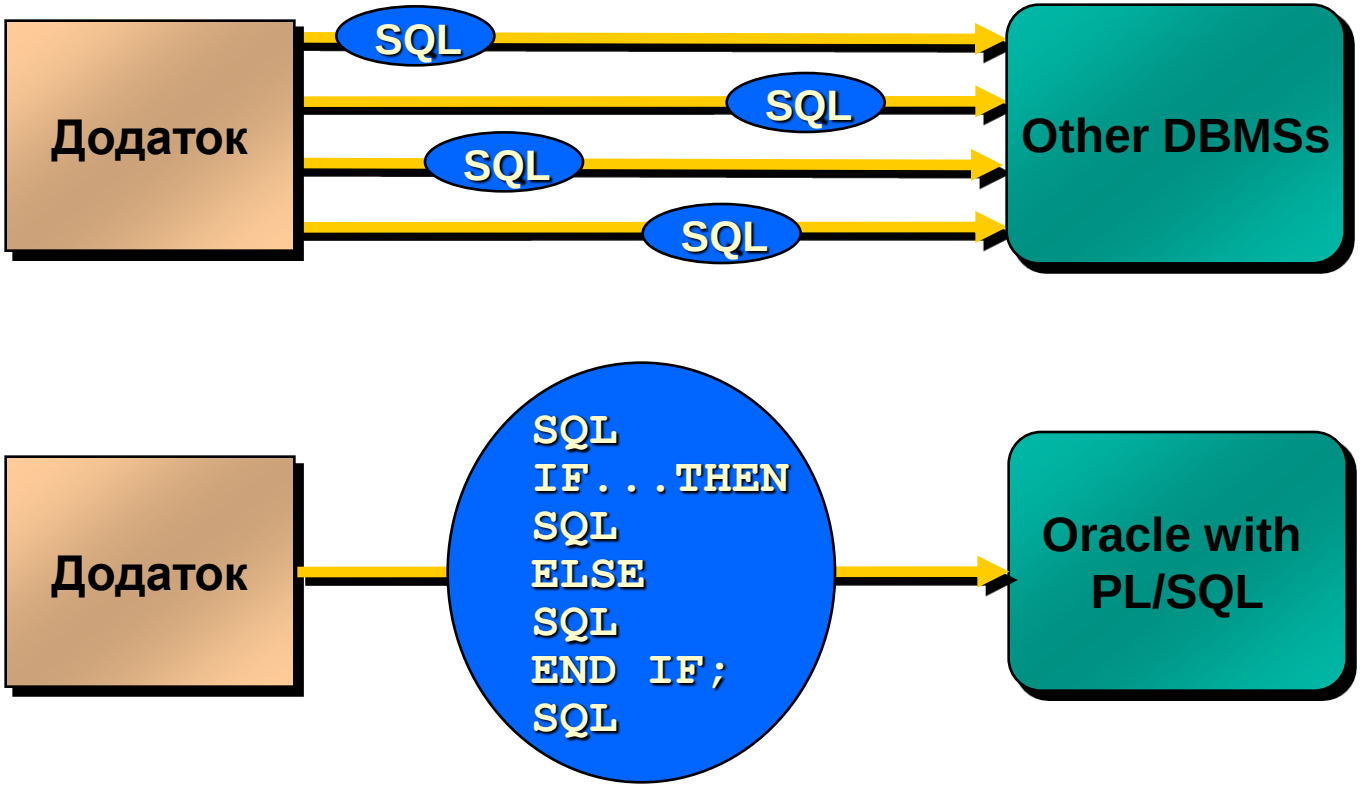
Переваги PL/SQL

- Тісна інтеграція з SQL;
- Висока продуктивність;
- Економія часу та зручність програмування;
- Повна переносимість;
- Високий ступінь безпеки;
- Доступ до попередньо встановлених пакетів;
- Підтримка ООП;
- Підтримка розробки веб-додатків і сторінок.



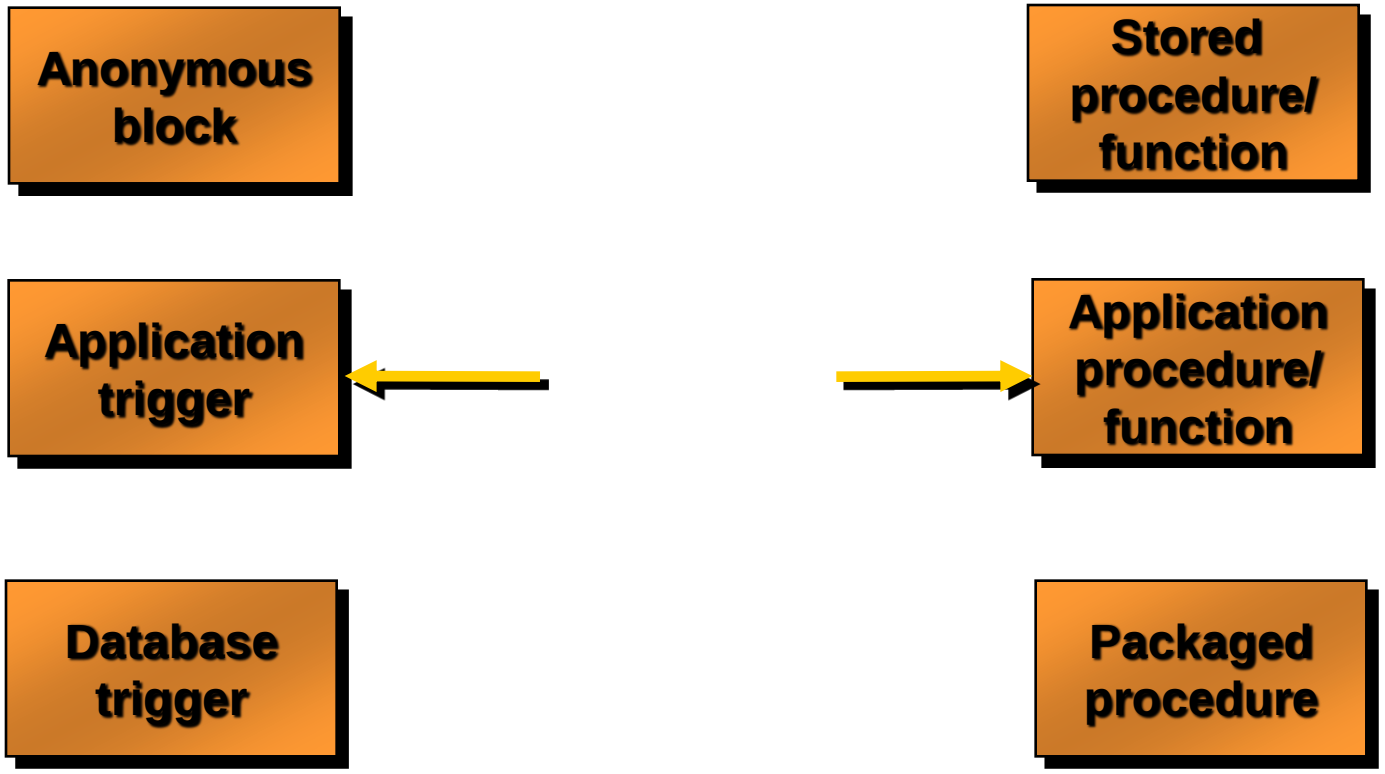
Переваги PL/SQL

Поліпшене виконання



PL/SQL

- Кожна одиниця PL / SQL складається з одного або декількох блоків.
- Блоки можуть бути окремими або вкладеними один в одного.
- Базові блоки (процедури, функції і анонімні блоки), що становлять програму на PL / SQL, є логічними блоками, які можуть містити будь-яку кількість вкладених підблоків.



Тригер

- **Три́гер** — *процедура*, що зберігається (блок PL/SQL);
- спрацьовує за певної події;
- користувач не викликає тригер безпосередньо.

Застосування тригерів

- Розширений аудит дій;
- Запобігання помилкових транзакцій;
- Розширена перевірка цілісності (наприклад між розподіленими даними);
- Реалізація складної бізнес-логіки;
- Реалізація складної авторизації;
- Прозоре ведення журналу подій;
- Автоматичне заповнення стовпців таблиці;
- Відстеження системних подій.

Загальний шаблон

```
CREATE [OR REPLACE] TRIGGER ім’я_тригера

{BEFORE | AFTER | INSTEAD OF }
подія_тригера  [OR подія_2 [OR подія_3]]

ON {таблиця_або_представлення |
[REFERENCING [OLD AS стара][NEW AS нова]]

[FOR EACH ROW ]
[WHEN умова_тригера]
[DECLARE < об’ява змінних >]

BEGIN

тіло_тригера;

END;
```

```
CREATE [OR REPLACE] TRIGGER
ім’я_тригера

{BEFORE | AFTER} подія_тригера

ON [DATABASE | схема]

[WHEN умова_тригера]

[DECLARE <об’ява змінних >]

BEGIN

тіло_тригера;

END;
```

DML-події

Подія	Опис. Подія відбувається тоді, коли…
INSERT	… у таблицю або предаставлення додається рядок
UPDATE	… інструкція UPDATE змінює таблицю або представлення
UPDATE OF і’мя_стовпця	… оновлюється вказаний стовпець таблиці, причому для кожного рядка - окрема подія
DELETE	… рядок видаляється з таблиці або представлення. При виконанні інструкції TRUNCATE ця подія не відбувається

DDL-події

Подія	Опис. Подія відбувається тоді, коли…
ALTER	… інструкція ALTER змінює об'єкт бази даних. У цьому контексті об'єкти подібні таблицям або пакетам (в ALL_OBJECTS). Може застосовуватися до окремої схеми або до всієї бази даних.
ANALIZE	… база даних збирає або видаляє статистику або перевіряє структуру об'єкта бази даних.
ASSOCIATE STATISTICS	… база даних асоціює тип статистики з об'єктом бази даних.
AUDIT	... база даних записує операцію аудиту.
COMMENT	… змінюється коментар до об'єкта бази даних.
CREATE	… створюється об'єкт бази даних. Не відбувається при виконанні інструкції CREATE CONTROLFILE
DIASSOCIATE STATISTICS	… база даних від'єднує тип статистики від об'єкта бази даних.
DROP	… інструкція DROP видаляє об'єкт з БД. У цьому контексті об'єкти подібні таблицям або пакетам (в ALL_OBJECTS). Може застосовуватися до окремої схеми або до всієї бази даних.
GRANT	… призначається привілея системі, ролі або об'єкту.
NOAUDIT	… база даних виконує інструкцію NOAUDIT.
RENAME	… інструкція RENAME змінює ім'я об'єкта БД.
REVOKE	… анулюється привілея системи, ролі або об'єкта.
TRUNCATE	… виконується інструкція TRUNCATE для очищення таблиці або кластера від непотрібних даних.

Події на рівні БД

Подія	Опис. Подія відбувається тоді, коли…
SERVERERROR	... реєструється повідомлення про помилку сервера. <Тільки для тригерів AFTER.>
LOGON	... створюється сеанс (користувач підключається до БД). <Тільки для тригерів AFTER.>
LOGOFF	... закривається сеанс (користувач відключається від БД). <Тільки для тригерів BEFORE.>
STARTUP	... відкривається БД. <Тільки для тригерів AFTER.>
SHUTDOWN	... закривається БД. <Тільки для тригерів BEFORE.>
SUSPEND	... через помилку сервера зависає транзакція.

Приклад 1

При оновленні таблиці `Emp_tab` тригер перевіряє значення, і якщо нова зарплата більше 1000, то заносить дані про зміни в таблицю `Emp_log`.

```
CREATE OR REPLACE TRIGGER Log_salary_increase
AFTER UPDATE ON Emp_tab
FOR EACH ROW
WHEN (new.Sal > 1000)
BEGIN
    INSERT INTO Emp_log (Emp_id, Log_date, New_salary, Action)
     VALUES (:new.Empno, SYSDATE, :new.SAL, 'NEW SAL');
END;
```

Приклад 2

При вставці або оновленні даних тригер виводить стару, нову зарплати, різницю між ними:

```
CREATE OR REPLACE TRIGGER Print_salary_changes
BEFORE DELETE OR INSERT OR UPDATE ON Emp_tab
FOR EACH ROW
DECLARE sal_diff number; --оголосили змінну
BEGIN
sal_diff := :new.sal - :old.sal;
dbms_output.put('Old salary: ' || :old.sal);
dbms_output.put('New salary: ' || :new.sal);
dbms_output.put_line('Diff:   '|| sal_diff);
END;
```

Приклад 3 с CASCADE UPDATE

При оновленні стовпця `deptno` таблиці `dept` каскадно оновлюється стовпець `deptno` в таблиці `emp`.

```
CREATE TRIGGER tu_dept
AFTER UPDATE OF deptno ON dept
FOR EACH ROW
BEGIN
if (:old.deptno != :new.deptno)  then
      begin
      update emp
      set   deptno = :new.deptno
      where        emp.deptno = :old.deptno ;
      end;
end if;
END;
```

Приклад 4

```
-- Приклад базується на таблицях:

CREATE TABLE T4 (a INTEGER, b CHAR(10));
CREATE TABLE T5 (c CHAR(10), d INTEGER);

-- Тригер вставляє в таблицю T5 рядки,
-- які вставляє користувач у T4 і які
-- відповідають певній умові:

CREATE TRIGGER trig1
    AFTER INSERT ON T4
    REFERENCING NEW AS newRow
    FOR EACH ROW
    WHEN (newRow.a <= 10)
    BEGIN
        INSERT INTO T5 VALUES(:newRow.b, :newRow.a);
    END;
```

Приклад 5

```
-- при додаванні рядка в таблицю accounts
-- додається рядок в таблицю account_balances

CREATE or REPLACE TRIGGER trig_accounts
after INSERT or UPDATE or DELETE on  accounts
FOR each ROW
BEGIN
  IF  inserting THEN
    INSERT INTO account_balances
        ( ACCOUNT_ID, PREVIOUS_BALANCE )
       VALUES   ( :NEW.ACCOUNT_ID, 0.0);
  ELSIF  updating  THEN NULL;
  ELSIF  deleting  THEN NULL;
END IF;
END;
```

Приклад 6

```
CREATE TABLE logon_tbl (who VARCHAR2(30), WHEN DATE);


-- при вході користувача на сервер дані
-- про це записуються в таблицю logon_tbl

CREATE OR REPLACE TRIGGER trg_logon_db
  after logon ON database
BEGIN
  INSERT INTO logon_tbl (who, WHEN) VALUES (USER,
SYSDATE);
END;
```

Приклад 7

```
-- Створювати нових співробітників можна тільки в
-- робочі дні та в робочий час


CREATE OR REPLACE TRIGGER secure_emp
before INSERT on emp
BEGIN
  IF (TO_CHAR (sysdate,'DY') IN ('SAT','SUN'))
     OR (TO_CHAR(sysdate,'HH24') NOT BETWEEN
     '08' AND '18')
  THEN RAISE_APPLICATION ERROR (-20500,
  'You may only insert into EMP during normal
   hours. ') ;
  END IF;
END;
/
```

Рекомендації до використання тригерів

- Використовуйте тригери, щоб виконати додатковий код при настанні певних подій;
- Не потрібно дублювати тригерами існуючий функціонал Oracle Database. (Наприклад перевірка обмежень);
- Якщо тригер довше 60 рядків - краще реалізувати збережену процедуру, яка буде викликана з тригеру;
- Тригер виконується для події незалежно від користувача, який викликав подію;
- Не створюйте рекурсивні тригери (наприклад AFTER UPDATE тригер, який додає в ту ж таблицю).