Rozdział 10b
Obsługa błędów wykonania programu PL/SQL

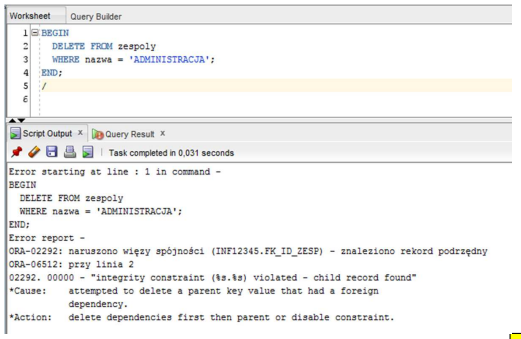
Wyjątki predefiniowane i użytkownika, zgłaszanie i obsługa wyjątków

Komunikaty o błędach serwera Oracle

- **Wykonanie niepoprawnej operacji w trakcie działania programu PL/SQL powoduje wygenerowanie błędu.**
- **Jeśli błąd nie zostanie obsłużony przez programistę, kończy działanie programu a na konsoli wyświetlany jest odpowiedni komunikat.**
- **Komunikat zawiera:**
  - **prefiks określający produkt, którego dotyczy błąd (np. ORA),**
  - **numer błędu,**
  - **opis błędu.**

Komunikaty o błędach serwera Oracle - przykłady

- **Przykładowe komunikaty:**
  - **ORA-00001 – naruszenie klucza podstawowego lub klucza unikalnego podczas definiowania/modyfikowania rekordu,**
  - **ORA-02292 – naruszenie klucza obcego podczas usuwania rekordu,**
  - **ORA-01476 – próba wykonania operacji dzielenia z dzielnikiem równym 0.**



Procedura RAISE_APPLICATION_ERROR

- **Umożliwia programiście zakończenie działania programu z wyświetleniem własnego komunikatu o błędzie.**
- **Parametry wywołania:**
  - **numer błędu z przedziału od -20999 do -20000,**
  - **komunikat.**

```
DECLARE
  CURSOR c_prac IS ...
BEGIN
  FOR prac_record IN c_prac LOOP
    ...
    IF (prac_record.placa_pod < 300 ) THEN
      RAISE_APPLICATION_ERROR(-20010, 'Pracownik ze zbyt niską pensją!');
    ELSE
      ...
    END IF;
  END LOOP;
END;
```

```
END LOOP;
END;
Error report -
ORA-20010: Pracownik ze zbyt niską pensją!
ORA-06512: przy linia 4
```

Obsługa błędu wykonania programu PL/SQL

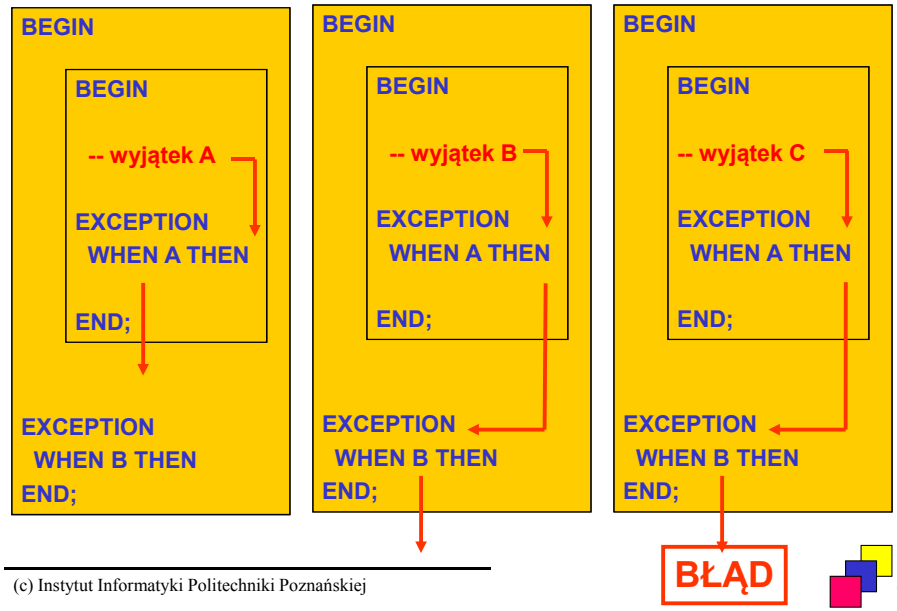
- **Możliwa tylko dla tych błędów wykonania, dla których zdefiniowano tzw. wyjątki (ang. *exception*).**
- **Wyjątek jest identyfikowany wyłącznie przez nazwę, np.:**
  - NO_DATA_FOUND,
  - TOO_MANY_ROWS,
  - ZERO_DIVIDE.
- **Wyjątki predefiniowane (systemowe):**
  - **nie wymagają zadeklarowania w programie,**
  - **zdefiniowane dla najczęściej obsługiwanych błędów serwera,**
  - **generowane automatycznie podczas wystąpienia w programie błędu, z którym są związane.**
- **Wyjątki użytkownika:**
  - **wymagają zadeklarowania w programie przez użytkownika,**
  - **najczęściej wywoływane ręcznie za pomocą polecenia RAISE,**
  - **możliwe jest związanie wyjątku użytkownika z błędem serwera.**

Sekcja EXCEPTION w bloku PL/SQL

- **Umożliwia zdefiniowanie w programie kodu reagującego na wystąpienie błędów wykonania.**
- **Składa się z podsekcji, każda podsekcja reaguje najczęściej na jeden błąd (jednak możliwa jest również obsługa wielu błędów w jednej sekcji).**
- **Błąd, obsługiwany w podsekcji, wskazuje się przez podanie nazwy związanego z nim wyjątku.**
- **Podsekcja WHEN OTHERS – obsługuje wszystkie niewymienione błędy, sekcja opcjonalna.**

```
DECLARE
  ...
BEGIN
  ...
EXCEPTION
  WHEN <wyjątek_1> THEN
    sekwencja poleceń
  WHEN <wyjątek_2> THEN
    sekwencja poleceń
  WHEN <wyjątek_3> OR <wyjątek_4> THEN
    sekwencja poleceń
  WHEN OTHERS THEN
    sekwencja poleceń
END;
```


Propagacja obsługi wyjątku



Wyjątki predefiniowane

Nazwa wyjątku	Numer błędu	Wartość SQLCODE
CURSOR_ALREADY_OPEN	ORA-06511	-6511
DUP_VAL_ON_INDEX	ORA-00001	-1
INVALID_CURSOR	ORA-01001	-1001
INVALID_NUMBER	ORA-01722	-1722
LOGIN_DENIED	ORA-01017	-1017
NO_DATA_FOUND	ORA-01403	100
NOT_LOGGED_ON	ORA-01012	-1012
PROGRAM_ERROR	ORA-06501	-6501
STORAGE_ERROR	ORA-06500	-6500
TIMEOUT_ON_RESOURCE	ORA-00051	-51
TOO_MANY_ROWS	ORA-01422	-1422
VALUE_ERROR	ORA-06502	-6502
ZERO_DIVIDE	ORA-01476	-1476

Wyjątki predefiniowane – przykład

```
DECLARE
   v_id_zesp zespoly.id_zesp%TYPE;
   v_nazwa zespoly.nazwa%TYPE := '&nazwa';
BEGIN
   SELECT id_zesp INTO v_id_zesp FROM zespoly
   WHERE nazwa = v_nazwa;

   DBMS_OUTPUT.PUT_LINE('Pracownicy z zespole '|| v_nazwa);

   FOR prac IN (SELECT nazwisko FROM pracownicy
                WHERE id_zesp = v_id_zesp ORDER BY nazwisko) LOOP
     DBMS_OUTPUT.PUT_LINE(prac.nazwisko);
   END LOOP;
EXCEPTION
   WHEN no_data_found THEN
      DBMS_OUTPUT.PUT_LINE('Nie istnieje zespół o nazwie '|| v_nazwa);
END;
```


Funkcje SQLCODE i SQLERRM

- **Służą do identyfikacji błędu.**
- **SQLCODE – zwraca numer błędu, który wystąpił.**
  - **numer ujemny, jedyne wartości dodatnie to: 100 dla wyjątku NO_DATA_FOUND i 1 dla wyjątków użytkownika.**
- **SQLERRM – zwraca komunikat błędu, który wystąpił.**
- **Funkcje wykonane w sytuacji braku błędu zwracają:**
  - **SQLCODE: 0**
  - **SQLERRM:** ORA-0000: normal, successful completion
- **Najczęściej stosowane w podsekcji WHEN OTHERS.**

```
...
EXCEPTION
   ...
   WHEN OTHERS THEN
     DBMS_OUTPUT.PUT_LINE('Wystąpił błąd numer: ' || SQLCODE);
     DBMS_OUTPUT.PUT_LINE(SQLERRM);
END;
```


Wyjątki użytkownika

- **Najczęściej służą do przekazania do środowiska wywołującego informacji, że w programie wystąpiła jakaś sytuacja błędna.**
  - **rozwiązanie alternatywne dla procedury RAISE_APPLICATION_ERROR.**
- **Muszą zostać jawnie zadeklarowane.**

```
DECLARE
        v_liczba NUMBER := 0;
        ex_moj_wyjatek EXCEPTION;
...
```

- **Wywoływane z użyciem polecenia RAISE.**
- **Wywołanie wyjątku przerywa działanie programu, system przystępuje do wyszukiwania procedury obsługi wyjątku (w identyczny sposób jak dla wyjątków predefiniowanych).**

```
BEGIN
  ...
  RAISE ex_moj_wyjatek;
  ...
```


Wyjątki użytkownika – przykład

```
DECLARE
   v_id_zesp zespoly.id_zesp%TYPE := &zespol;
   v_liczba INTEGER;
   ex_pracownicy_w_zespole EXCEPTION;
BEGIN
   SELECT COUNT(*) INTO v_liczba
   FROM pracownicy WHERE id_zesp = v_id_zesp;

   IF (v_liczba > 0) THEN
      RAISE ex_pracownicy_w_zespole;
   END IF;

   DELETE FROM zespoly
   WHERE id_zesp = v_id_zesp;

   DBMS_OUTPUT.PUT_LINE('Zespół został usunięty!');
EXCEPTION
  WHEN ex_pracownicy_w_zespole THEN
   DBMS_OUTPUT.PUT_LINE('Do zespołu są przypisani pracownicy. Usunięcie anulowane!');
END;
```


Wyjątek użytk. związany z błędem systemowym

- **Istnieje możliwość związania wyjątku użytkownika z błędem systemowym (tj. błędem identyfikowanym przez numer, np. ORA-02292).**
- **Stosowane, gdy w programie chcemy obsłużyć błąd systemowy, dla którego nie istnieje wyjątek predefiniowany.**
- **Powiązanie realizowane przez użycie dyrektywy kompilatora EXCEPTION INIT:**

```
PRAGMA EXCEPTION_INIT(wyjątek_użytkownika, numer_błędu_sys);
```

  - **wyjątek użytkownika musi zostać wcześniej zadeklarowany.**

Przykład

- **Związanie błędu ORA-2292 z wyjątkiem użytkownika.**

```
DECLARE
   v_id_zesp zespoly.id_zesp%TYPE := &zespol;
   ex_pracownicy_w_zespole EXCEPTION;

   PRAGMA EXCEPTION_INIT(ex_pracownicy_w_zespole, -2292);
BEGIN
   DELETE FROM zespoly
   WHERE id_zesp = v_id_zesp;

   DBMS_OUTPUT.PUT_LINE('Zespół został usunięty!');
EXCEPTION
  WHEN ex_pracownicy_w_zespole THEN
   DBMS_OUTPUT.PUT_LINE('Do zespołu są przypisani pracownicy. Usunięcie anulowane!');
END;
```


Wiadomości uzupełniające

Wyświetlenie stosu błędów

- **Funkcja DBMS_UTILITY.FORMAT_ERROR_STACK**
- **Zwraca zawartość aktualnego stosu błędów**
- **Wykorzystywana najczęście w sytuacji śledzenia błędów wygenerowanych w kodzie (procedurze, funkcji, itd.), wywołanym z innego kodu**

```
DECLARE
...
BEGIN
...
EXCEPTION
  ...
  WHEN OTHERS THEN
     DBMS_OUTPUT.PUT_LINE(DBMS_UTILITY.FORMAT_ERROR_STACK);
END;
```