# 8. HANDLING EXCEPTION

## Objektif

Setelah mengikuti materi ini, diharapkan untuk dapat :

1. Mendefinisikan PL/SQL exception
2. Menangani exception yang tidak bisa ditangani
3. Menyebutkan dan menggunakan tipe penanganan exception PL/SQL yang berbeda
4. Mengatasi kesalahan yang tidak diantisipasi
5. Menjelaskan efek pertambahan exception dalam nested block
6. Mengatur pesan exception PL/SQL

## 8.1. Menangani Exception Dengan PL/SQL

Exception adalah sebuah identifier dalam PL/SQL, yang muncul pada eksekusi sebuah blok yang terhenti. Eksekusi blok selalu berhenti saat PL/SQL memunculkan exception, namun anda harus menspesifikasikan sebuah penanganan exception / exception handler untuk menyelesaikannya.

## 8.2. Metode Untuk Memunculkan Exception

- Terjadi kesalahan dan exception yang bersangkutan otomatis dimunculkan. Sebagai contoh, jika terjadi kesalahan ORA-01403 saat tidak terdapat baris yang ditampilkan saat statement SELECT dijalankan, maka PL/SQL memunculkan exception NO_DATA_FOUND.
- Memunculkan exception secara eksplisit dengan mengeluarkan statement RAISE dalam sebuah blok.

## 8.3. Menangani Exception

Jika exception muncul di executable section sebuah block, proses menuju ke penanganan exception yang bersangkutan. Jika PL/SQL berhasil menangani exception, maka exception tidak berpindah ke enclosing block atau environment.

## 8.4. Memindahkan Exception

Jika exception yang muncul pada executable section sebuah block tidak memiliki penanganan exception, blok PL/SQL berhenti dengan suatu kesalahan dan exception dipindahkan ke calling environment.

## 8.5. Tipe-Tipe Exception

Program exception dibuat untuk menghindari gangguan selama runtime. Terdapat tiga tipe exception.

| Exception | Deskripsi | Petunjuk Penanganan |
|---|---|---|
| Predefined Oracle Server error | Satu dari sekitar 20 error yang peling sering terjadi pada PL/SQL code | Tidak perlu dideklarasikan dan biarkan Oracle Server menanganinya secara implisit. |
| Non-predefined Oracle Server error | Oracle Server error standar lainnya | Deklarasikan pada declarative section dan biarkan Oracle Server menanganinya secara implisit. |
| User-defined error | Sebuah kondisi yang didefinisikan tidak normal oleh developer / programer | Deklarasikan pada declarative section dan dimunculkan secara eksplisit |

## 8.6. Exception

Penanganan kesalahan dengan menyertakan routine pada exception handling section di blok PL/SQL. Setiap penanganan exception berisi klausa WHEN, yang menspesifikasikan sebuah exception diikuti oleh urutan statement yang akan dijalankan saat exception muncul.

***Sintaks:***

```
EXCEPTION
     WHEN exception1 [OR exception2 ...] THEN
          statement1;
     statement2;
...
     [WHEN exception1 [OR exception2 ...] THEN
          statement1;
     statement2;
          ...]
     [WHEN OTHERS THEN
```

```
      statement1;
 statement2;
      ...]
```

***dengan:***

| | |
|---|---|
| exception | nama standar predefined exception atau nama dari user-defined yang dideklarasikan di declarative section |
| statement | statement PL/SQL |
| OTHERS | klausa penanganan exception opsional yang akan menangani exception yang belum dispesifikasikan |

```
SET SERVEROUTPUT ON

DECLARE
   stock_price NUMBER := 9.73;
   net_earnings NUMBER := 0;
   pe_ratio NUMBER;

BEGIN
-- Perhitungan yang memungkinkan terjadinya error karena pembagian dengan nilai 0
   pe_ratio := stock_price / net_earnings;
   DBMS_OUTPUT.PUT_LINE('Price/earnings ratio = ' || pe_ratio);

EXCEPTION  -- exception handlers begin
-- Hanya satu blok WHEN yang akan dijalankan
   WHEN ZERO_DIVIDE THEN  -- blok untuk menangani error karena pembagian dengan nilai 0
      DBMS_OUTPUT.PUT_LINE('Company must have had zero earnings.');
      pe_ratio := NULL;
   WHEN OTHERS THEN  -- blok untuk menangani error lainnya
      DBMS_OUTPUT.PUT_LINE('Some other kind of error occurred.');
      pe_ratio := NULL;
END;  -- exception handlers dan blok habis disini
/
```

*Lihat Video*

## 8.7. Penanganan Exception Dengan When Others

Exception–handling section hanya menangani exception yang dispesifikasikan, oleh karena itu exception lain tidak akan ditangani kecuali jika anda menyertakan exception handler OTHERS. Exception handler OTHERS inilah yang akan menyelesaikan exception yang belum ditangani. Oleh sebab itu, OTHERS diletakkan paling akhir dari semua exception handler yang didefinisikan.

## 8.8. Petunjuk Penanganan Exception

- Awali exception-handling section pada blok dengan keyword EXECPTION.
- Definisikan beberapa exception-handling, masing-masing dengan sekumpulan aksi yang terkait.
- Ketika sebuah exception terjadi, PL/SQL hanya memproses satu handler sebelum keluar dari blok tersebut.
- Letakkan klausa OTHERS setelah semua klausa exception-handling.
- Klausa OTHERS paling banyak satu.
- Exception tidak dapat muncul dalam assignment statement atau statement SQL.

## 8.9 Menangani Predefined Oracle Server Error

Penanganan Predefined Oracle Server error adalah dengan merujuk pada nama standarnya di exception-handling routine.

***Catatan :*** PL/SQL mendeklarasikan predefined exception pada paket STANDARD. Disarankan untuk selalu menyertakan exception NO_DATA_FOUND dan TOO_MANY_ROWS, yang merupakan hal yang paling sering terjadi.

**PREDEFINED EXCEPTIONS**

| Nama Exception | Oracle Server Error Number | Deskripsi |
|---|---|---|
| ACCES_INTO_NULL | ORA-06530 | Berusaha untuk memberikan nilai pada atribut dari sebuah objek yang belum dipersiapkan. |

| | | |
|---|---|---|
| COLLECTION_IS_NULL | ORA-06531 | Berusaha untuk mengaplikasikan metode kumpulan, selain EXIST terhadap nested table atau varray yang belum dipersiapkan. |
| CURSOR ALREADY OPEN | ORA-06511 | Membuka sebuah kursor yang sudah terbuka. |
| DUP_VAL_ON_INDEX | ORA-00001 | Menyisipkan sebuah nilai duplikat. |

| | | |
|---|---|---|
| INVALID_CURSOR | ORA-01001 | Terjadi operasi kursor yang ilegal. |
| INVALID NUMBER | ORA-01722 | Konversi string karakter ke numerik tidak berhasil. |
| LOGIN DENIED | ORA-01017 | Log-in ke Oracle dengan username atau password yang salah. |
| NO_DATA_FOUND | ORA-01403 | Single row SELECT namun tidak ada data. |
| NOT LOGGED ON | ORA-01012 | Program PL/SQL mencoba koneksi ke database tanpa terhubung dangan Oracle. |
| PROGRAM_ERROR | ORA-06501 | PL/SQL mempunyai masalah internal. |
| ROWTYPE_MISMATCH | ORA-06504 | Variabel host kursor dan variabel kursor PL/SQL memiliki tipe yang berbeda. |

| | | |
|---|---|---|
| STORAGE_ERROR | ORA-06500 | PL/SQL kekurangan memori atau memori rusak. |
| SUBSCRIPT_BEYOND_COUNT | ORA-06533 | Merujuk ke sebuah nested table atau anggota varray menggunakan nomor indeks yang lebih besar dari jumlah anggota yang ada. |
| SUBSCRIPT_OUTSIDE_LIMIT | ORA-06532 | Merujuk ke sebuah nested table atau anggota varray menggunakan nomor indeks di luar dari batas yang legal (misalnya –1). |
| TIMEOUT_ON_RESOURCE | ORA-00051 | Terjadi time-out ketika Oracle menunggu ketersediaan sumber |
| | | daya. |
| TOO MANY ROWS | ORA-01422 | Single row SELECT menghasilkan lebih dari satu baris. |
| VALUE ERROR | ORA-06502 | Terjadi kesalahan aritmatik, konversi, pemotongan (truncate), atau size-constraint. |
| ZERO DEVIDE | ORA-01476 | Berusaha untuk membagi dengan nol. |

## 8.10. Menangani Exception Predefined Oracle Server

Hanya satu exception yang muncul dan dapat ditangani pada satu waktu.

***Sintaks:***

```
BEGIN
EXCEPTION
        WHEN  NO_DATA_FOUND THEN
        statement1;
   statement2;
        WHEN TOO_MANY_ROWS THEN
             statement1;
        WHEN OTHERS THEN
        statement1;
   statement2;          statement3;
END;
```

## 8.11. Menangani Non-Predefined Oracle Server Errors

Menangani sebuah kesalahan non-predefined Oracle Server adalah dengan mendeklarasikannya terlebih dahulu, atau menggunakan OTHERS. Exception yang telah dideklarasikan muncul secara implisit. Pada PL/SQL, pragma (pseudoinstruction) EXCEPTION_INIT memerintahkan compiler untuk mengkaitkan nama sebuah exception dengan nomor Oracle error. Dengan demikian dapat merujuk ke internal exception dengan menyebutkan namanya dan menuliskan handler untuk menanganinya.

***Catatan :*** Pragma (biasa disebut *pseudoinstuctions*) merupakan keyword yang menandakan bahwa statement tersebut merupakan compiler directive, tidak diproses saat blok PL/SQL dieksekusi. Pragma inilah yang mengarahkan compiler PL/SQL untuk menerjemahkan semua nama exception dalam blok dengan nomor Oracle Server error.

## 8.12. Menangani Exception Non-Predefined Oracle Server

1. Deklarasikan nama exception pada declarative section.

   ***Sintaks:***

   ```
   exception EXCEPTION ;
   ```

   ***dengan:***

   *exception* nama exception.

2. Kaitkan exception yang telah dideklarasikan dengan nomor Oracle Server error standar menggunakan statement PRAGMA EXCEPTION_INIT.

   ***Sintaks:***

   ```
   PRAGMA EXCEPTION_INIT (exception,
   error_number);
   ```

   ***dengan:***

   *exception* nama exception.

3. Referensikan exception yang telah dideklarasikan dengan exception-handling routine yang sesuai.

## 8.13. Fungsi-Fungsi Penanganan Error

Ketika sebuah exception terjadi, error code atau error message dapat diidentifikasikan yang berkaitan dengan exception tersebut menggunakan dua buah fungsi. Berdasarkan nilai dari kode atau pesan, dapat diputuskan aksi apa yang akan dilanjutkan selanjutnya.

SQLCODE memberikan nomor Oracle error untuk exception internal. Anda dapat memberikan nomor kesalahan ke SQLERRM yang akan menghasilkan pesan yang sesuai dengan nomor error.

| Fungsi | Deskripsi |
|---|---|
| SQLCODE | Menghasilkan nilai numerik untuk error code (Anda bisa memakai variabel NUMBER). |

| | |
|---|---|
| SQLERRM | Menghasilkan data karakter yang berisi pesan yang sesuai dengan nomor error. |

Contoh Nilai SQLCODE

| **Nilai SQLCODE** | **Deskripsi** |
|---|---|
| 0 | Tidak ditemukan exception |
| 1 | User-defined exception |
| +100 | Exception NO_DATA_FOUND. |
| *negative number* | Nomor Oracle Server error yang lain. |

Ketika sebuah exception terperangkap dalam exception handlers WHEN OTHERS, dapat menggunakan satu set fungsi umum untuk dapat mengidentifikasikan kesalahan tersebut.

***Contoh:***
```
DECLARE   v_error_code        NUMBER;
    v_error_message   VARCHAR(255);
    BEGIN
    ...
    EXCEPTION
    ...
     WHEN OTHERS THEN
            ROLLBACK
       v_error_code := SQLCODE;
       v_error_message := SQLERRM;
            INSERT INTO errors
            VALUES(v_error_code, v_error_message);
       END;
```

Pada contoh diatas, nilai dari SQLCODE dan SQLERRM dapat ditugaskan menjadi sebuah variabel dan kemudian variabel tersebut dapat digunakan dalam statement SQL. Potong nilai dari SQLERRM sampai panjang tertentu sebelum menempatkannya pada sebuah variabel.

## 8.14. Menangani User-Defined Exception

PL/SQL memperbolehkan untuk mendefinisikan sendiri exception anda. User-defined PL/SQL harus:

- Dideklarasikan pada bagian blok deklarasi PL/SQL
- Dimunculkan secara eksplisit dengan statement RAISE.

Anda menangani user-defined exception dengan mendeklarasikan dan memunculkannya secara eksplisit.

1) Deklarasikan nama untuk user-defined exception pada declarative section.

   ***Sintaks:***

   ```
   exception    EXCEPTION;
   ```

2) Gunakan statement RAISE untuk memunculkan exception secara eksplisit pada bagian executable section.

   ***Sintaks:***

   ```
   RAISE      exception;
   ```

3) Referensikan exception yang telah dideklarasikan dengan exception handling routine yang sesuai.

   ***Contoh:***

   ```
   DECLARE e_invalid_product
   EXCEPTION
   BEGIN
     UPDATE    product
   SET         descrip = '&product_description'
   WHERE prodid = &product_number;
   IF SQL%NOTFOUND THEN
     RAISE e_invalid_product;
   END IF;
   COMMIT;
   EXCEPTION
   WHEN e_invalid_product THEN
   ```

```
DBMS_OUTPUT.PUT_LINE('Invalid product number.');
END;
```

Contoh blok di atas memodifikasi deksripsi sebuah produk. User memberikan nomor produk dan deskripsi yang baru. Jika user memasukkan nomor yang tidak tercatat, maka tidak ada baris yang akan dimodifikasi dalam table product. Munculkan sebuah exception dan cetak pesan untuk user bahwa nomor yang diberikan adalah invalid.

***Catatan:***

Gunakan statement RAISE dalam exception handler untuk memunculkan exception yang sama ke calling environment.

## 8.15. Memindahkan Exception

Daripada menangani exception dengan blok PL/SQL, pindahkan exception ke calling environment yang akan menanganinya. Masing-masing calling environment memiliki metode sendiri untuk menampilkan dan mengakses kesalahan.

CALLING ENVIRONMENT

| **SQL *Plus** | **Menampilkan nomor kesalahan dan pesan dilayar** |
|---|---|
| Procedure Builder | Menampilkan nomor kesalahan dan pesan dilayar |
| Oracle Developer Forms | Mengakses nomor kesalahan dan pesan dalam sebuah trigger dgn fungsi paket ERROR_CODE dan ERROR_TEXT |
| Precompiler application | Mengakses nomor exception melalui data struktur SQLCA |
| An Enclosing PL/SQL blok | menangani exception dalam exception handling routine dari enclosing block |

## 8.16. Memindahkan Exception dalam Sebuah Blok

Saat sebuah subblok menangani exception, subblok tersebut diakhiri secara normal dan kontrol berada di enclosing blok setelah statement End. Tetapi, bila terdapat sebuah exception pada PL/SQL dan blok tersebut tidak memiliki sebuah handler untuk exception tersebut, maka exception dipindahkan ke enclosing blok berikutnya sampai ditemukan handler yang akan menanganinya. Jika tidak ada satupun blok yang dapat menangani exception tersebut, menghasilkan sebuah exception yang tidak tertangani di dalam lingkungan host.

Saat sebuah exception dipindahkan ke enclosing blok, aksi yang seharusnya dapat dijalankan dilewatkan. Salah satu keuntungannya adalah anda dapat menutup statementstatement yang memerlukan error handling eksklusif tersendiri dalam blok mereka sendiri, dan meninggalkan lebih banyak exception handling yang umum untuk enclosing blok.

## 8.17. Prosedur RAISE_APPLICATION_ERROR

Prosedur RAISE_APPLICATION_ERROR digunakan untuk mengkomunikasikan sebuah predefined exception secara interaktif dengan mengembalikan sebuah kode kesalahan non standar dan pesan kesalahan. Dengan RAISE_APPLICATION_ERROR, dapat melaporkan kesalahan pada aplikasi anda dan menghindari pengembalian exception yang tidak tertangani.

***Sintaks:***

```
raise_application_error ( error_number,
    message[ , { TRUE | FALSE }] );
```

***dengan:***

| | |
|---|---|
| Error number | Spesifikasi user untuk nomor exception antara – 20000 sampai – 20999 |
| Message | Pesan user-specified untuk exception, berupa karakter string sampai 2048 byte. |
| TRUE\|FALSE | Parameter boolean yang bersifat opsional (jika TRUE, error ditempatkan pada stack dari error sebelumnya. Jika FALSE, defaultnya, error menggantikan seluruh error sebelumnya). |

***Contoh:***

```
DECLARE
   num_tables NUMBER;
BEGIN
   SELECT COUNT(*) INTO num_tables FROM USER_TABLES;
   IF num_tables < 1000 THEN
      /* Membuat kode error sendiri (ORA-20101) dengan pesan error yang dibuat sendiri juga. */
      raise_application_error(-20101, 'Expecting at least 1000 tables');
   ELSE
      NULL; -- Do the rest of the processing (for the non-error case).
   END IF;
END;
/
```

*Lihat video*