

Fair Use Notice:

The material used in this presentation i.e., pictures/graphs/text, etc. is solely intended for educational/teaching purpose, offered free of cost to the students for use under special circumstances of Online Education due to COVID-19 Lockdown situation and may include copyrighted material - the use of which may not have been specifically authorised by Copyright Owners. It's application constitutes Fair Use of any such copyrighted material as provided in globally accepted law of many countries. The contents of presentations are intended only for the attendees of the class being conducted by the presenter.

DATABASE SYSTEMS (SW215)

EXCEPTION HANDLING

By : HIRA NOMAN

EXCEPTIONS

- Sometimes the normal flow of a PL/SQL program is disrupted due to the occurrence of an abnormal condition while execution.
- Such abnormal condition are known as exceptions.
- Exception may occur due to invalid user input, system failure or logical errors.
- Every exception in PL/SQL has an error number and an error message and some exceptions have names as well.
- Exception handling in a PL/SQL code is carried out in the EXCEPTION block.
- There are three categories of PL/SQL exceptions:
 1. PRE-DEFINED SYSTEM EXCEPTIONS.
 2. USER-DEFINED EXCEPTIONS.
 3. UNDEFINED SYSTEM EXCEPTIONS.

PRE-DEFINED SYSTEM EXCEPTIONS

- Every pre-defined exception has a name and an error number, but these exceptions must be handled by their names.
- These exceptions are thrown automatically by the system.
- There is no need to raise pre-defined exceptions.
- There is no need to even declare the pre-defined exceptions anywhere in the PL/SQL code.

COMMON PRE-DEFINED EXCEPTIONS

Exception Name	Oracle Error	Description
DUP_VAL_ON_INDEX	ORA-00001	It is raised when duplicate values are attempted to be stored in a column with unique index.
INVALID_CURSOR	ORA-01001	It is raised when attempts are made to make a cursor operation that is not allowed, such as closing an unopened cursor.
LOGIN_DENIED	ORA-01017	It is raised when a program attempts to log on to the database with an invalid username or password.
NO_DATA_FOUND	ORA-01403	It is raised when a SELECT INTO statement returns no rows.
TOO_MANY_ROWS	ORA-01422	It is raised when a SELECT INTO statement returns more than one row.
ZERO_DIVIDE	ORA-01476	It is raised when an attempt is made to divide a number by zero.

```
DECLARE
    e_id emp.empno%type := 8 ;
    e_name emp.ename%type ;

BEGIN
    SELECT  ename INTO  e_name
    FROM emp
    WHERE empno = e_id;
    DBMS_OUTPUT.PUT_LINE ('Name: ' || e_name);

--EXCEPTION
    -- WHEN no_data_found THEN
        -- dbms_output.put_line('No such employee!');
    --WHEN others THEN
        -- dbms_output.put_line('Error!');

END ;
```

Error encountered



An error was encountered performing the requested operation:

ORA-01403: no data found
ORA-06512: at line 6
01403. 00000 - "no data found"
*Cause:
*Action:
Vendor code 1403Error at Line:1

OK

SYNTAX:

DECLARE

<declarations section>

BEGIN

<executable command(s)>

EXCEPTION

<exception handling goes here >

WHEN exception1 THEN

exception1-handling-statements

WHEN exception2 THEN

exception2-handling-statements

WHEN exception3 THEN

exception3-handling-statements

.....

WHEN OTHERS THEN

exception4-handling-statements

END;

EXAMPLE:

DECLARE

e_id emp.empno%type := 8 ;

e_name emp.ename%type ;

BEGIN

SELECT ename INTO e_name

FROM emp

WHERE empno= e_id;

DBMS_OUTPUT.PUT_LINE ('Name: ' || e_name);

EXCEPTION

WHEN no_data_found THEN

dbms_output.put_line('No such employee!');

WHEN others THEN

dbms_output.put_line('Error!');

END;

/

DECLARE

```
e_id emp.empno%type := 8 ;  
e_name emp.ename%type ;
```






BEGIN






```
SELECT ename INTO e_name  
FROM emp  
WHERE empno = e_id;  
DBMS_OUTPUT.PUT_LINE ('Name: ' || e_name);
```

EXCEPTION

```
WHEN no_data_found THEN  
    dbms_output.put_line('No such employee!');  
WHEN others THEN  
    dbms_output.put_line('Error!');
```

END ;

 Results |  Script Output |  Explain |  Autotrace |  DBMS Output

 |  |  |  | Buffer Size: |  Poll

No such employee!

USER-DEFINED EXCEPTIONS

- A user-defined exception must be declared and then raised explicitly, using either a RAISE statement or the procedure.
- These exceptions are declared with EXCEPTION datatype in the declarative section.
- User-defined exceptions can also be raised using the WHEN clause.

SYNTAX (USER-DEFINED EXCEPTION DECLARATION):

DECLARE

my-exception EXCEPTION ;

RAISING EXCEPTIONS

SYNTAX:

DECLARE

exception_name EXCEPTION;

BEGIN

IF condition THEN

RAISE exception_name ;

END IF ;

EXCEPTION

WHEN exception_name THEN

statement ;

END;

DECLARE

```
c_id customers.id %type := &cc_id;
c_name customerS.Name %type;
c_addr customers.address %type;
-- user defined exception
ex_invalid_id EXCEPTION;
```

BEGIN

```
IF c_id <= 0 THEN
    RAISE ex_invalid_id ;
ELSE
    SELECT name, address INTO c_name, c_addr
    FROM customers
    WHERE id = c_id;
    DBMS_OUTPUT.PUT_LINE ('Name: ' || c_name);
    DBMS_OUTPUT.PUT_LINE ('Address: ' || c_addr);
END IF;
```

EXCEPTION

-- user defined exception

```
WHEN ex_invalid_id THEN
    dbms_output.put_line('ID must be greater
than zero!');
```

-- pre-defined exception

```
WHEN no_data_found THEN
    dbms_output.put_line('No such customer!');
WHEN OTHERS THEN
    dbms_output.put_line('Error!');
END;
```

DECLARE

Hire_date EXCEPTION ;

Emp_no NUMBER ;

H_date DATE ;

BEGIN

Emp_no := &emp_no ;

SELECT hiredate INTO H_date FROM emp

WHERE empno = emp_no ;

IF h_date != SYSDATE THEN

RAISE hire_date ;

END IF;

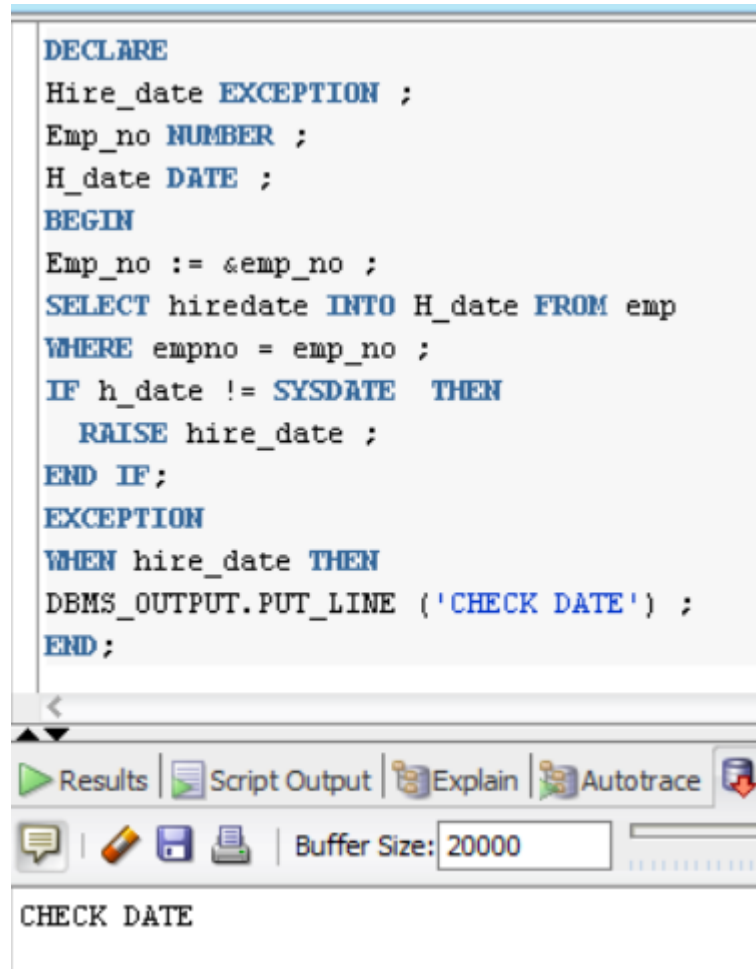
EXCEPTION

WHEN hire_date THEN

DBMS_OUTPUT.PUT_LINE ('CHECK DATE') ;

END;

```
DECLARE
Hire_date EXCEPTION ;
Emp_no NUMBER ;
H_date DATE ;
BEGIN
Emp_no := &emp_no ;
SELECT hiredate INTO H_date FROM emp
WHERE empno = emp_no ;
IF h_date != SYSDATE THEN
    RAISE hire_date ;
END IF;
EXCEPTION
WHEN hire_date THEN
DBMS_OUTPUT.PUT_LINE ('CHECK DATE') ;
END;
```



The screenshot shows a SQL IDE window with the following components:

- Code Editor:** Contains the PL/SQL code from the previous blocks, with keywords in blue and identifiers in black.
- Toolbar:** Includes icons for Results, Script Output, Explain, and Autotrace.
- Buffer Size:** A text box showing '20000'.
- Output Window:** Displays the text 'CHECK DATE'.

UN-DEFINED EXCEPTIONS

- The exceptions that are raised due to an error in PL/SQL or RDBMS processing and are not defined by PL/SQL are known as Un-defined exceptions.
- In PL/SQL only the most common exceptions have names and rest have only error numbers. These exception can be assigned the names using the special procedure **PRAGMA EXCEPTION_INIT**.

TASK A

1. Write a PL/SQL code that throws the DUP_VAL_ON_INDEX and then handles it with the help of an appropriate message.
2. Write a PL/SQL code that throws the TOO_MANY_ROWS and then handles it with the help of an appropriate message.