Chapter 4B:
Handling Runtime Errors in PL/SQL Programs

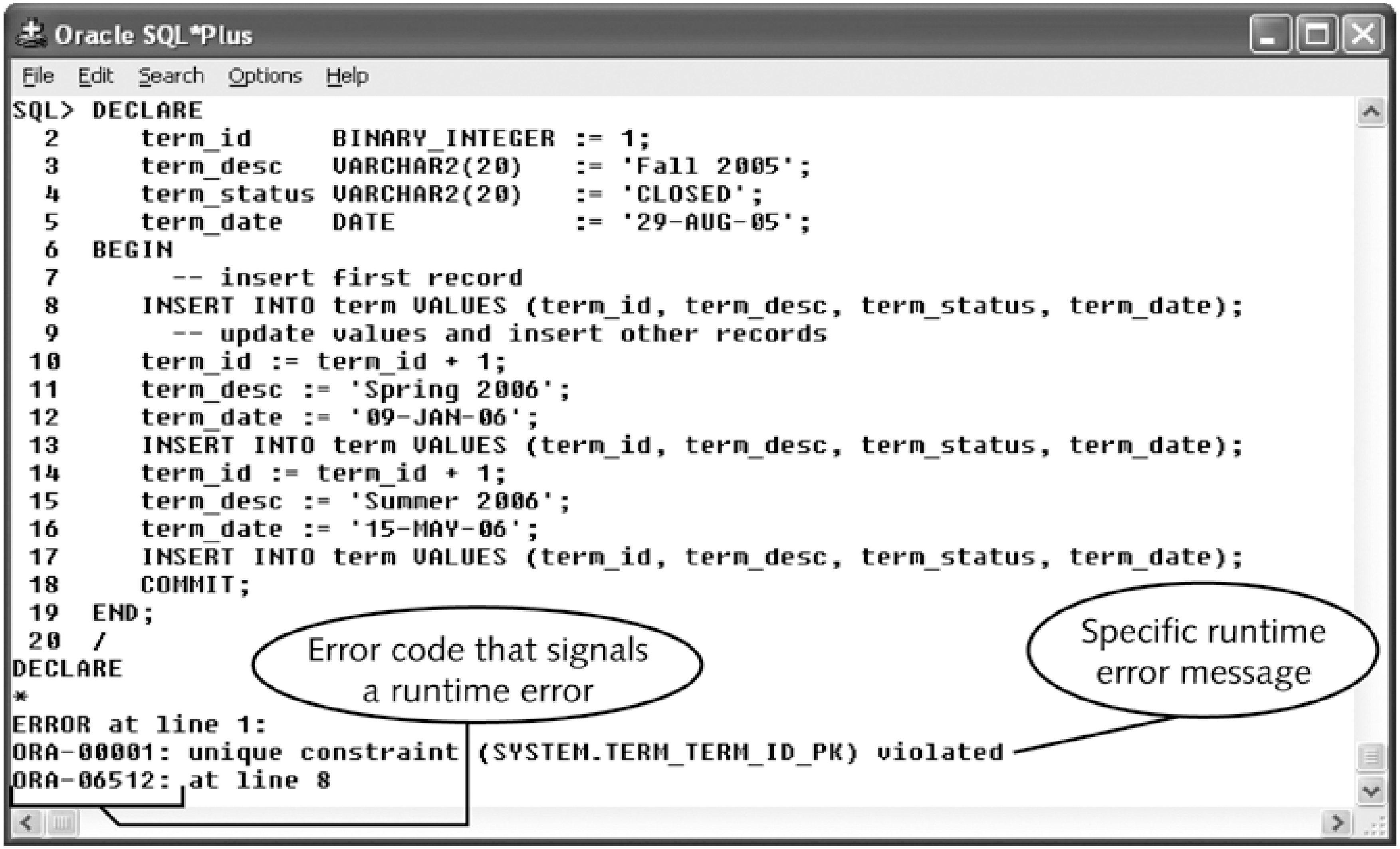
Joory Alselawy

CIS 318

Handling Runtime Errors in PL/SQL Programs

- Runtime errors
  - Occur when an **exception** (unwanted event) is **raised**
  - Cause program to fail during execution
- Possible causes (exceptions):
  - Division by zero
  - Constraint violation
  - inserting incompatible data
  - retrieving 0/several rows with implicit cursor
- Exception handling
  - Programmers place commands in EXCEPTION section
- Handle exception options
  - Correct error without notifying user of problem
  - Inform user of error without taking corrective action
- After exception handler executes
  - Program ends

```
DECLARE
    variable declarations
BEGIN
    program statements
EXCEPTION
    error-handling statements
END;
```


Oracle SQL*Plus
File Edit Search Options Help
Error code that signals a runtime error
Specific runtime error message

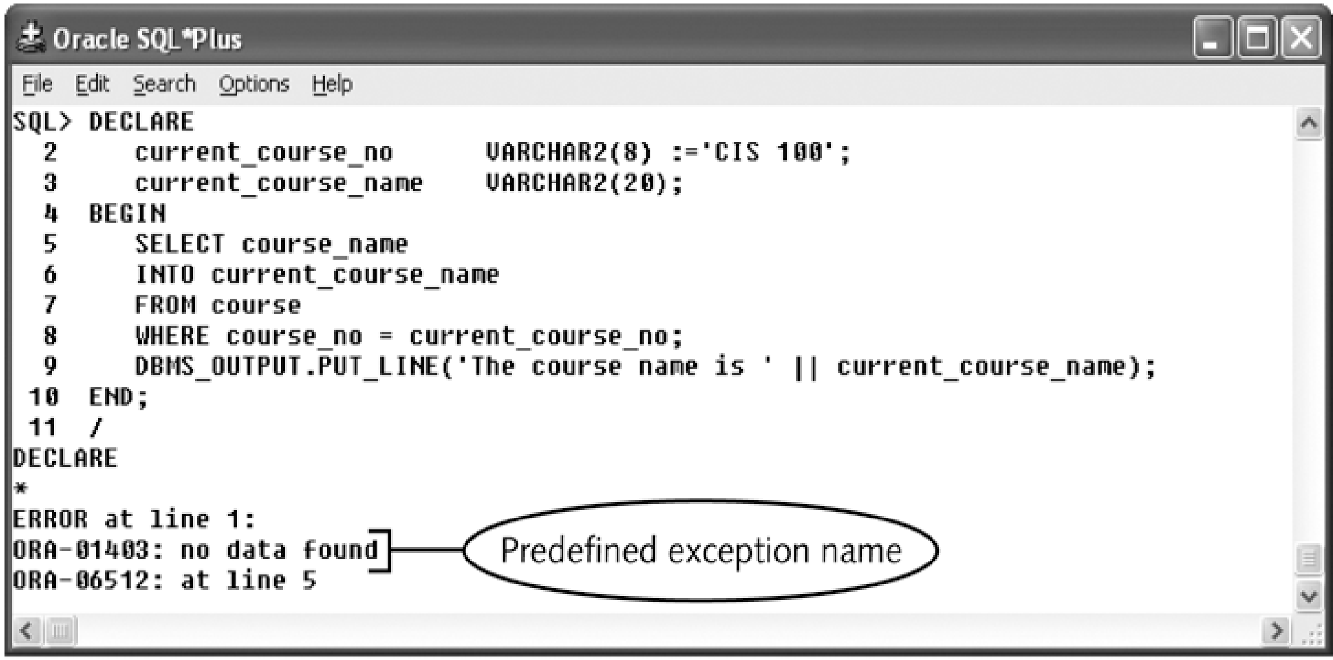
```
SQL> DECLARE
       term_id      BINARY_INTEGER := 1;
       term_desc    VARCHAR2(20)    := 'Fall 2005';
       term_status  VARCHAR2(20)    := 'CLOSED';
       term_date    DATE            := '29-AUG-05';
     BEGIN
         -- insert first record
       INSERT INTO term VALUES (term_id, term_desc, term_status, term_date);
         -- update values and insert other records
       term_id := term_id + 1;
       term_desc := 'Spring 2006';
       term_date := '09-JAN-06';
       INSERT INTO term VALUES (term_id, term_desc, term_status, term_date);
       term_id := term_id + 1;
       term_desc := 'Summer 2006';
       term_date := '15-MAY-06';
       INSERT INTO term VALUES (term_id, term_desc, term_status, term_date);
       COMMIT;
     END;
     /
DECLARE
*
ERROR at line 1:
ORA-00001: unique constraint (SYSTEM.TERM_TERM_ID_PK) violated
ORA-06512: at line 8
```

Figure 4-32 Example of a runtime error

- Handling error procedure depends the type of exception:
  - Predefined exception - undefined exception
  - User-defined exception

Predefined Exceptions

- Most common errors that occur in programs
- PL/SQL language:
  - Assigns exception name
  - Provides built-in exception handler for each predefined exception
- System automatically displays error message informing user of nature of problem

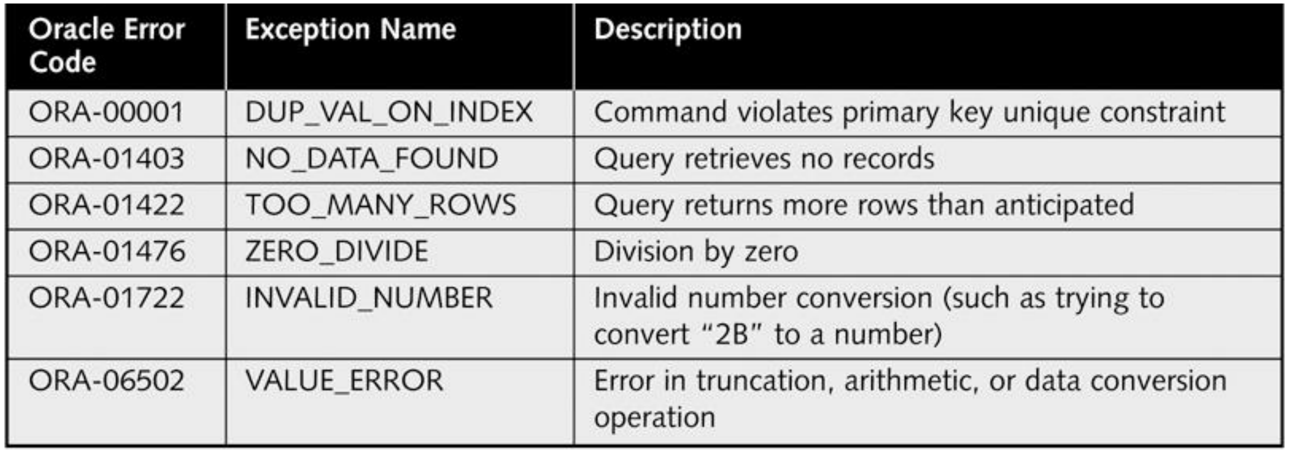


```
SQL> DECLARE
       current_course_no      VARCHAR2(8) :='CIS 100';
       current_course_name    VARCHAR2(20);
     BEGIN
       SELECT course_name
       INTO current_course_name
       FROM course
       WHERE course_no = current_course_no;
       DBMS_OUTPUT.PUT_LINE('The course name is ' || current_course_name);
     END;
     /
DECLARE
*
ERROR at line 1:
ORA-01403: no data found
ORA-06512: at line 5
```

Figure 4-34 PL/SQL program that generates a predefined exception

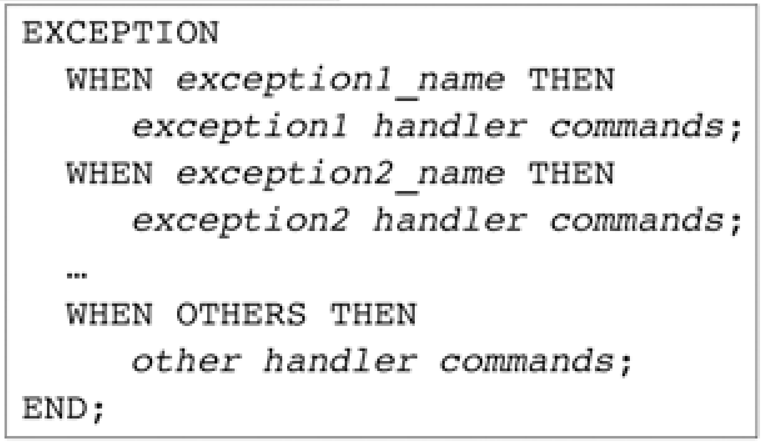
Exception Handler Syntax

- Can create exception handlers to display alternate error messages



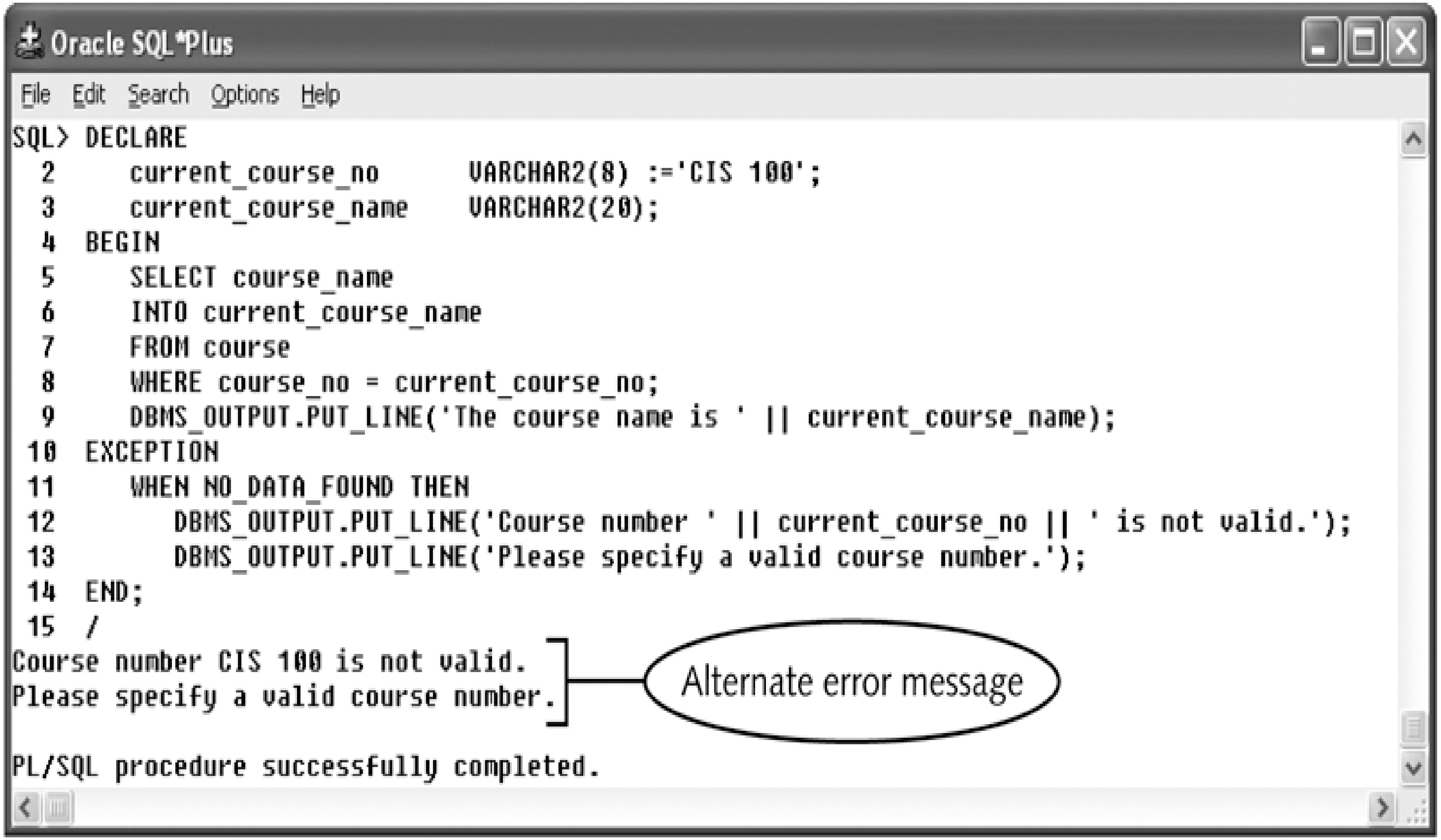
Oracle Error Code	Exception Name	Description
ORA-00001	DUP_VAL_ON_INDEX	Command violates primary key unique constraint
ORA-01403	NO_DATA_FOUND	Query retrieves no records
ORA-01422	TOO_MANY_ROWS	Query returns more rows than anticipated
ORA-01476	ZERO_DIVIDE	Division by zero
ORA-01722	INVALID_NUMBER	Invalid number conversion (such as trying to convert "2B" to a number)
ORA-06502	VALUE_ERROR	Error in truncation, arithmetic, or data conversion operation

Table 4-10 Common PL/SQL predefined exceptions



```
EXCEPTION
  WHEN exception1_name THEN
     exception1 handler commands;
  WHEN exception2_name THEN
     exception2 handler commands;
  …
  WHEN OTHERS THEN
     other handler commands;
END;
```

Figure 4-33 Exception handler syntax



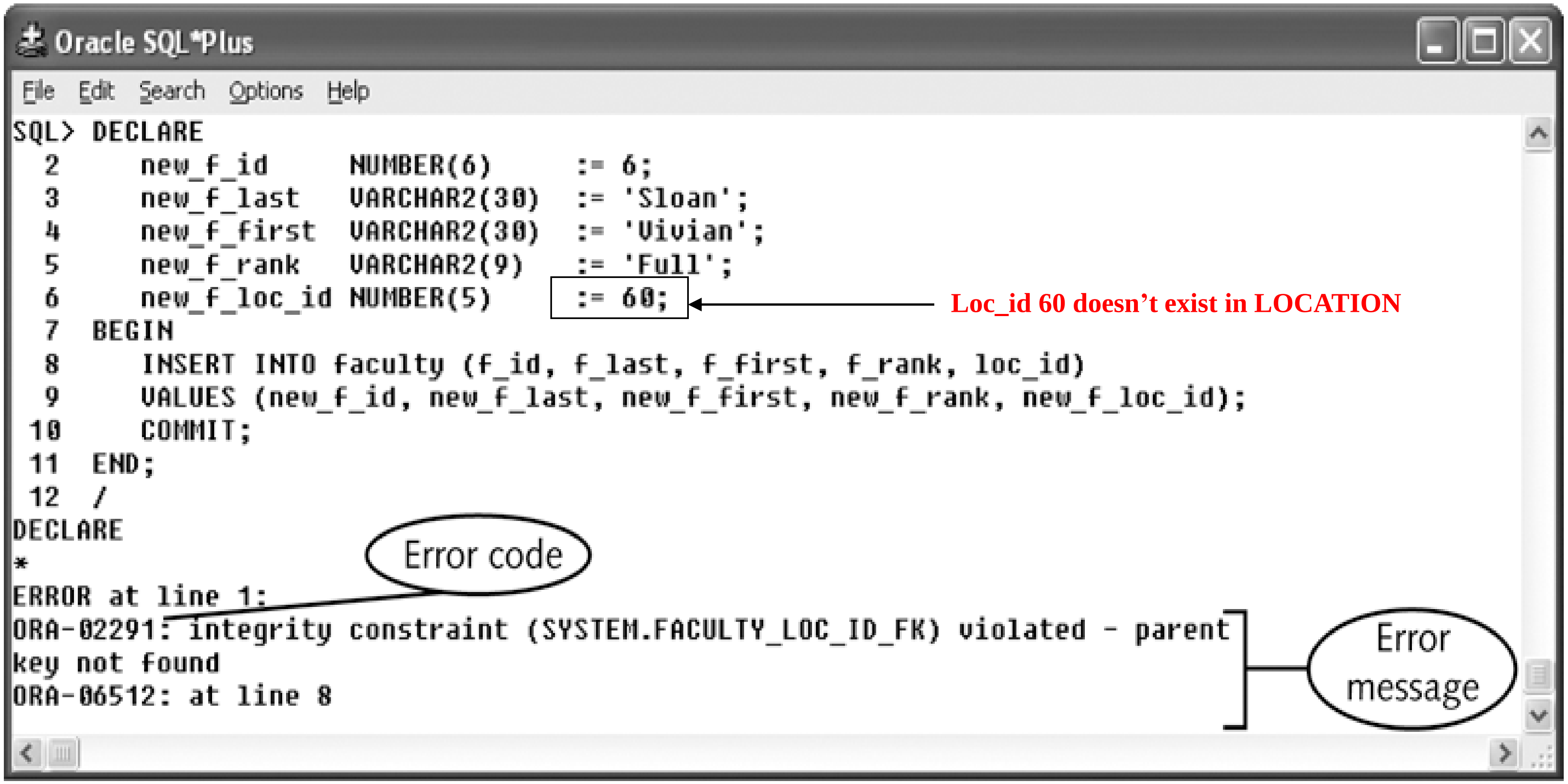
```
SQL> DECLARE
       current_course_no      VARCHAR2(8) :='CIS 100';
       current_course_name    VARCHAR2(20);
     BEGIN
       SELECT course_name
       INTO current_course_name
       FROM course
       WHERE course_no = current_course_no;
       DBMS_OUTPUT.PUT_LINE('The course name is ' || current_course_name);
     EXCEPTION
       WHEN NO_DATA_FOUND THEN
          DBMS_OUTPUT.PUT_LINE('Course number ' || current_course_no || ' is not valid.');
          DBMS_OUTPUT.PUT_LINE('Please specify a valid course number.');
     END;
     /
Course number CIS 100 is not valid.
Please specify a valid course number.

PL/SQL procedure successfully completed.
```

Figure 4-35 Creating an exception handler that displays an alternate message for a predefined exception

Undefined Exceptions

- Less common errors
- Do not have predefined names
- Must explicitly declare exception in program's declaration section
- Associate new exception with specific Oracle error code
- Create exception handler in exception section
  - Using same syntax as for predefined exceptions

Example of undefined exception


Oracle SQL*Plus
File Edit Search Options Help
Loc_id 60 doesn't exist in LOCATION
Error code
Error message

```
SQL> DECLARE
  2      new_f_id     NUMBER(6)     := 6;
  3      new_f_last   VARCHAR2(30)  := 'Sloan';
  4      new_f_first  VARCHAR2(30)  := 'Vivian';
  5      new_f_rank   VARCHAR2(9)   := 'Full';
  6      new_f_loc_id NUMBER(5)     := 60;
  7   BEGIN
  8      INSERT INTO faculty (f_id, f_last, f_first, f_rank, loc_id)
  9      VALUES (new_f_id, new_f_last, new_f_first, new_f_rank, new_f_loc_id);
 10      COMMIT;
 11   END;
 12   /
DECLARE
*
ERROR at line 1:
ORA-02291: integrity constraint (SYSTEM.FACULTY_LOC_ID_FK) violated - parent
key not found
ORA-06512: at line 8
```

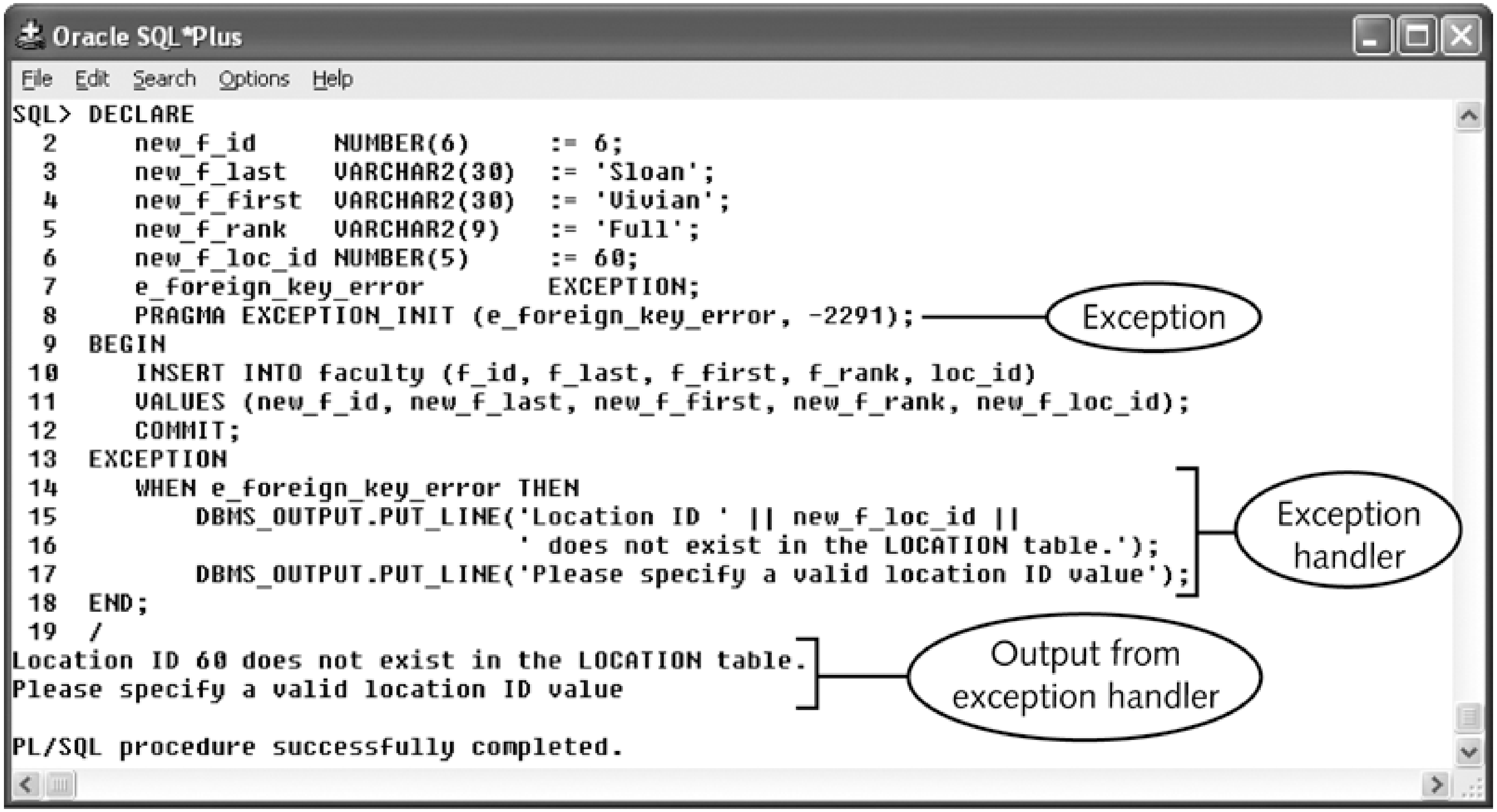
Figure 4-37 Example of an undefined exception

- The ORA-02291 exception is not predefined.
- Need to explicitly declare the exception and write a handler

Creating an exception handler

Syntax

```
DECLARE
    e_exception_name EXCEPTION;
    PRAGMA EXCEPTION_INIT(e_exception_name, -Oracle_error_code);
```

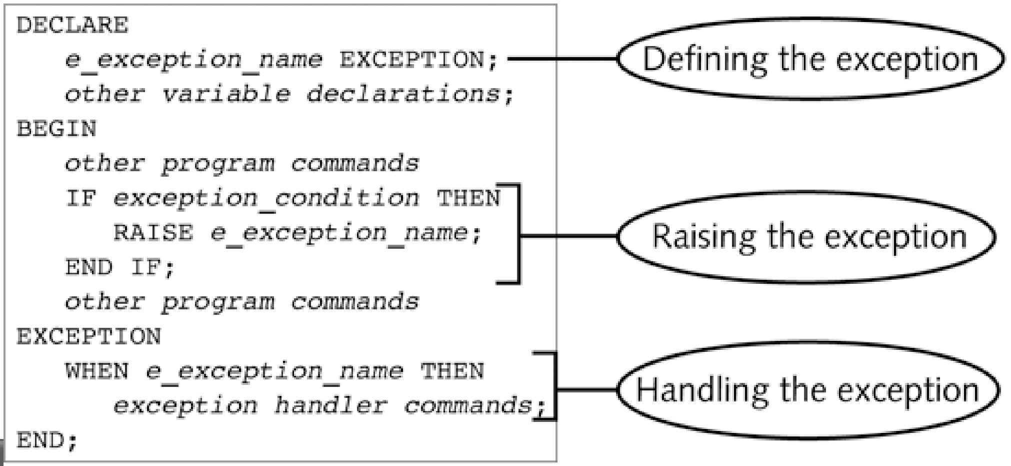


```
SQL> DECLARE
      new_f_id      NUMBER(6)      := 6;
      new_f_last    VARCHAR2(30)   := 'Sloan';
      new_f_first   VARCHAR2(30)   := 'Vivian';
      new_f_rank    VARCHAR2(9)    := 'Full';
      new_f_loc_id  NUMBER(5)      := 60;
      e_foreign_key_error          EXCEPTION;
      PRAGMA EXCEPTION_INIT (e_foreign_key_error, -2291);
   BEGIN
      INSERT INTO faculty (f_id, f_last, f_first, f_rank, loc_id)
      VALUES (new_f_id, new_f_last, new_f_first, new_f_rank, new_f_loc_id);
      COMMIT;
   EXCEPTION
      WHEN e_foreign_key_error THEN
          DBMS_OUTPUT.PUT_LINE('Location ID ' || new_f_loc_id ||
                                     ' does not exist in the LOCATION table.');
          DBMS_OUTPUT.PUT_LINE('Please specify a valid location ID value');
   END;
   /
Location ID 60 does not exist in the LOCATION table.
Please specify a valid location ID value

PL/SQL procedure successfully completed.
```

Figure 4-38 Creating an exception handler for an undefined exception

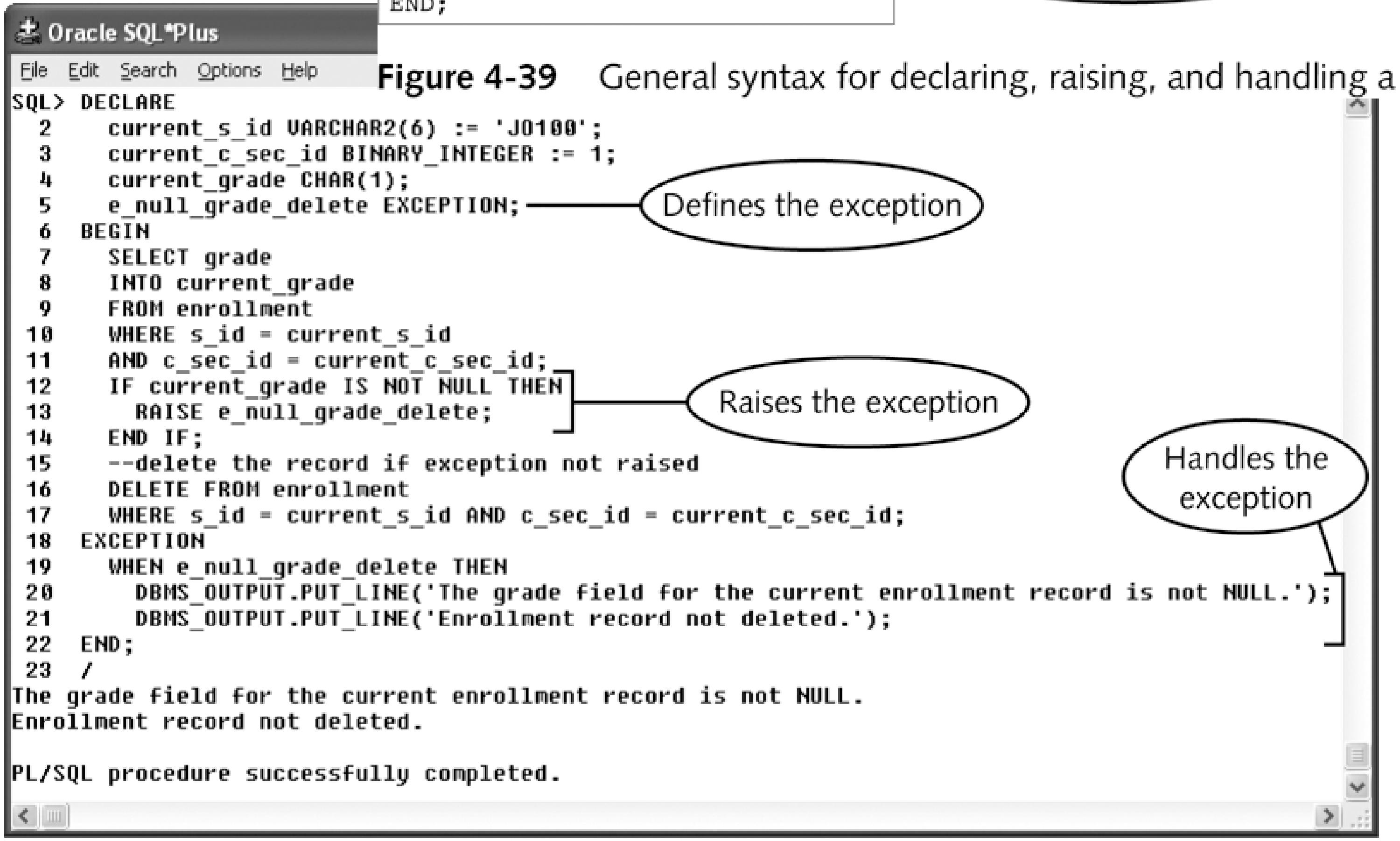
User-defined Exceptions

- Used to handle an exception that
  - Does not raise Oracle runtime error
  - But requires exception handling to
    - Enforce business rules or
    - Ensure integrity of database
- Example:
  - Internal Northwoods' rule is "Users can delete row from the ENROLLMENT table only if s_grade is NULL"
  - Trying to delete a delete an ENROLLMENT row where the s_grade is not NULL will raise an exception that needs to be handled



```
DECLARE
   e_exception_name EXCEPTION;
   other variable declarations;
BEGIN
   other program commands
   IF exception_condition THEN
      RAISE e_exception_name;
   END IF;
   other program commands
EXCEPTION
   WHEN e_exception_name THEN
      exception handler commands;
END;
```

Figure 4-39 General syntax for declaring, raising, and handling a user-defined exception



```
SQL> DECLARE
       current_s_id VARCHAR2(6) := 'JO100';
       current_c_sec_id BINARY_INTEGER := 1;
       current_grade CHAR(1);
       e_null_grade_delete EXCEPTION;
     BEGIN
       SELECT grade
       INTO current_grade
       FROM enrollment
       WHERE s_id = current_s_id
       AND c_sec_id = current_c_sec_id;
       IF current_grade IS NOT NULL THEN
         RAISE e_null_grade_delete;
       END IF;
       --delete the record if exception not raised
       DELETE FROM enrollment
       WHERE s_id = current_s_id AND c_sec_id = current_c_sec_id;
     EXCEPTION
       WHEN e_null_grade_delete THEN
         DBMS_OUTPUT.PUT_LINE('The grade field for the current enrollment record is not NULL.');
         DBMS_OUTPUT.PUT_LINE('Enrollment record not deleted.');
     END;
     /
The grade field for the current enrollment record is not NULL.
Enrollment record not deleted.

PL/SQL procedure successfully completed.
```

Figure 4-40 PL/SQL program to define, raise, and handle a user-defined exception

Thank You