8 Обработка исключений

Обработка исключительных ситуаций

В PL/SQL ошибки всех видов интерпретируются как исключения — ситуации, которые не должны возникать при выполнении программы. При возникновении ORA-XXXX инициируется исключение. Выполнение программы прекращается и управление передается разделу обработки исключения, если он есть.

Способы инициации исключения :

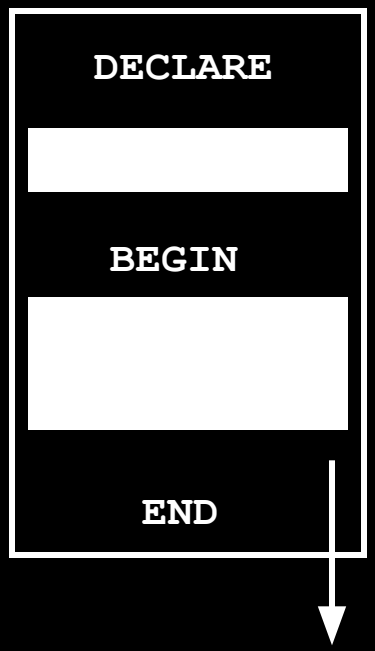
- ошибки, генерируемые системой (нехватка памяти, места и тп);
- ошибки, вызванные действиями пользователя (констрейнты, типы данных);
- предупреждения, выдаваемые приложением пользователю. Явный вызов исключения разработчиком в коде программы. (нарушение логики работы)

Способы обработки исключения:

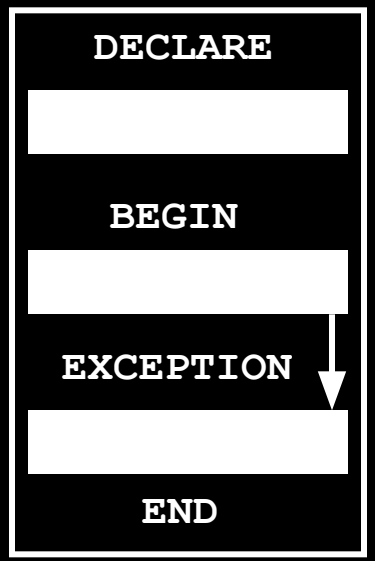
- Обработать в том же блоке.
- Передать ошибку в вызываемую среду.

Обработка исключительных ситуаций

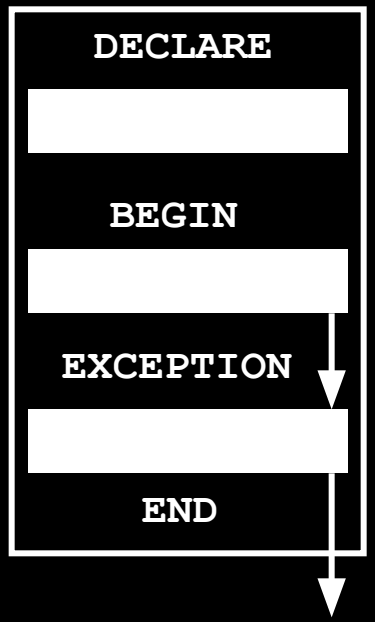
Исключение не обработано, передается внешней среде



Обработка исключения в том же блоке. Исключение обработано



Передача исключения во внешнюю среду. Исключение не обработано и передается внешней среде



Возникло исключение или инициация исключения

Обработка исключений

- С ключевого слова EXCEPTION начинается блок обработки особых ситуаций.
- Допустимо использование нескольких обработчиков событий
- Ключевое слово WHEN OTHERS должно быть указано в конце блока обработки.

Обработка исключений

```
DECALRE
BEGIN
……
EXCEPTION
    WHEN exception1 [OR exception2 . . .] THEN
        statement1;
        statement2;
     ...
   [WHEN exception3 [OR exception4 . . .] THEN
        statement1;
        statement2;
     ...]
   [WHEN OTHERS THEN
        statement1;
        statement2;
     ...]
END;          * Блоки begin exception end м.б. вложенными
```

Предопределенные ошибки Oracle

Не требуют объявления, доступны всегда. Часто возникающие ошибки:

- TOO_MANY_ROWS
- NO_DATA_FOUND
- INVALID_CURSOR
- CURSOR_ALREADY_OPEN
- NOT_LOGGED_ON
- PROGRAM_ERROR

```
EXCEPTION
   WHEN NO_DATA_FOUND THEN
      statement1;
   WHEN TOO_MANY_ROWS THEN
      statement1;
   WHEN OTHERS THEN
      statement1;
```

Исключения предопределенные ORACLE

Для ссылки на предопределенные исключения используются стандартные, зарезервированные описатели исключений.

Предопределенные исключения:

Имя исключения/Ошибка Oracle/ SQLCODE	Описание
CURSOR_ALREADY_OPEN ORA-6511 SQLCODE = –6511	Попытка открытия курсора, который был открыт ранее. Перед повторным открытием курсор необходимо сначала закрыть
DUP_VAL_ON_INDEX ORA-00001 SQLCODE = –1	Команда INSERT или UPDATE пытается сохранить повторяющиеся значения в столбцах, объявленных с ограничением UNIQUE
INVALID_CURSOR ORA-01001 SQLCODE = –1001	Ссылка на несуществующий курсор. Обычно ошибка встречается при попытке выборки данных из неоткрытого курсора или закрытия курсора до его открытия
INVALID_NUMBER ORA-01722 SQLCODE = –1722	Выполняемая SQL-команда не может преобразовать символьную строку в число. Это исключение отличается от VALUE_ ERROR тем, что оно инициируется только из SQL-команд
LOGIN_DENIED ORA-01017 SQLCODE = –1017	Попытка программы подключиться к СУБД Oracle с неверным именем пользователя или паролем. Исключение обычно встречается при внедрении кода PL/SQL в язык 3GL
NO_DATA_FOUND ORA-01403 SQLCODE = +100	Исключение инициируется в трех случаях: (1) при выполнении инструкции SELECT INTO (неявный курсор), которая не возвращает ни одной записи; (2) при ссылке на неинициализированную запись локальной таблицы PL/SQL; (3) при попытке выполнить операцию чтения после достижения конца файла при использовании пакета UTL_FILE
NOT_LOGGED ON ORA-01012 SQLCODE = –1012	Программа пытается обратиться к базе данных (обычно из инструкции DML) до подключения к СУБД Oracle
PROGRAM_ERROR ORA-06501 SQLCODE = –6501	Внутренняя программная ошибка PL/SQL. В сообщении об ошибке обычно предлагается обратиться в службу поддержки Oracle

Исключения предопределенные ORACLE

STORAGE_ERROR ORA-06500 SQLCODE = −6500	Программе PL/SQL не хватает памяти или память по какой-то причине повреждена
TIMEOUT_ON_RESOURCE ORA-00051 SQLCODE = −51	Тайм-аут СУБД при ожидании ресурса
TOO_MANY_ROWS ORA-01422 SQLCODE = −1422	Команда SELECT INTO возвращает несколько записей, хотя должна возвращать лишь одну (в таких случаях инструкция SELECT включается в явное определение курсора, а записи выбираются по одной)
TRANSACTION_BACKED_OUT ORA-00061 SQLCODE = −61	Удаленная часть транзакции отменена либо при помощи явной инструкции ROLLBACK, либо в результате какого-то другого действия (например, неудачного выполнения команды SQL или DML в удаленной базе данных)
VALUE_ERROR ORA-06502 SQLCODE = −6502	Ошибка связана с преобразованием, усечением или проверкой ограничений числовых или символьных данных. Это общее и очень распространенное исключение. Если подобная ошибка содержится в инструкции SQL или DML, то в блоке PL/SQL инициируется исключение INVALID_NUMBER
ZERO_DIVIDE ORA-01476 SQLCODE = −1476	Попытка деления на ноль

Пример

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

Не предопределенные ошибки Oracle

Не предопределенные ошибки обрабатываются в блоке исключения как и предопределенные, но в секции OTHERS Для получения информации о коде исключения используем функцию SQLCODE, информация о сообщении функция SQLERRM

```
WHEN OTHERS THEN
  ROLLBACK;
  v_error_code := SQLCODE ;
  v_error_message := SQLERRM ;
  INSERT INTO errors
  VALUES(v_error_code, v_error_message);
```

Можно предопределить любимые ошибки (связать номер ORA-XXXX и имя) в DECLARE и обращаться по имени как с предопределенными.

Функции SQLCODE SQLERRM

- SQLCODE: Возвращает числовое значение для кода последней возникшей ошибки (кроме NO_DATA_FOUND для нее =100).
- SQLERRM: Возвращает текст, связанный с номером ошибки.

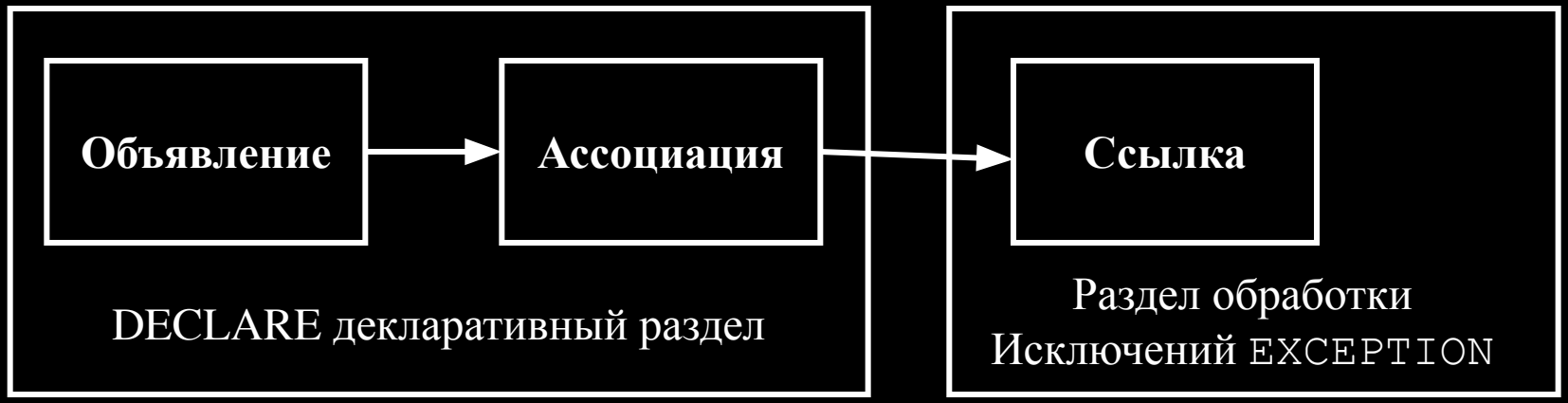
ORA-00001: unique constraint (string.string) violated
Cause: An UPDATE or INSERT statement attempted to insert a duplicate key.
Action: Either remove the unique restriction or do not insert the key.

SQLCODE =-1
SQLERRM: ORA-00001: нарушено ограничение уникальности

* Список ошибок Oracle - Database Error Messages
https://docs.oracle.com/database/121/ERRMG/toc.htm

Не предопределенные исключения Oracle

Требуют объявления (или анализа кода ошибки)



Имя исключения

Код PRAGMA EXCEPTION_INIT

Обработка исключения

Пример

Обработка исключения ORA–2292 нарушение ограничения целостности:
ORA-02292: violated integrity constraint (owner.constraintname)- child record found

```
DECLARE
   ERROR_DELETE_MY EXCEPTION;                    (1)
   PRAGMA EXCEPTION_INIT(ERROR_DELETE_MY,-2292); (2)
   v_deptno DEPARTMENTS.DEPARTMENT_ID%TYPE :=30;
BEGIN
   DELETE FROM DEPARTMENTS  WHERE DEPARTMENT_ID =
v_deptno;
EXCEPTION
   WHEN ERROR_DELETE_MY THEN                     (3)
   DBMS_OUTPUT.PUT_LINE('Cannot remove dept' ||
TO_CHAR(v_deptno) || '. Employees exist.');
END;
```

* Демонстрация 3-режимов см сноску:
- Без EXCEPTION
- OTHERS
- Предопределение

Пример логирования ошибок

```
DECLARE
   v_error_code NUMBER;
   v_error_message VARCHAR2(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;
```

Обработка пользовательских исключений

RAISE *имя_исключения;*
RAISE_APPLICATION_ERROR *(номер, текст, флаг);*

Обработка пользовательских исключений

Чтобы программист имел возможность самостоятельно инициировать исключения, в Oracle поддерживается команда RAISE. С ее помощью можно инициировать как собственные, так и системные исключения.

```
DECLARE
   e_invalid_department EXCEPTION;          1
BEGIN
   UPDATE dept SET dname ='Testers'
   WHERE deptno =11111;
  IF SQL%NOTFOUND THEN
      RAISE e_invalid_department;           2
   END IF;
   COMMIT;
EXCEPTION
   WHEN e_invalid_department THEN
   DBMS_OUTPUT.PUT_LINE('No such department id.');  3
   END;
```

RAISE_APPLICATION_ERROR

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

- Предназначена для генерации ошибок, установленных пользователем.
- Преимущество перед командой RAISE (которая тоже может инициировать явно объявленные исключения) заключается в том, что она позволяет связать с исключением сообщение об ошибке.
- Диапазон ошибок, доступный пользователю: -20000...-20999

RAISE_APPLICATION_ERROR

- Используется в двух различных местах:
  - Выполнимый раздел. Пример 1 след слайд
  - Раздел Исключения . Пример 1 след слайд
- Возвращает ошибку пользователю аналогичным способом, принятым в сервере ORACLE. Номер и текст.

Примеры

Пример 1

```
...
DELETE FROM emp
WHERE mgr = v_mgr;
   IF SQL%NOTFOUND THEN
    RAISE_APPLICATION_ERROR (-20202,'This is not a
                  valid manager');
   END IF;
```

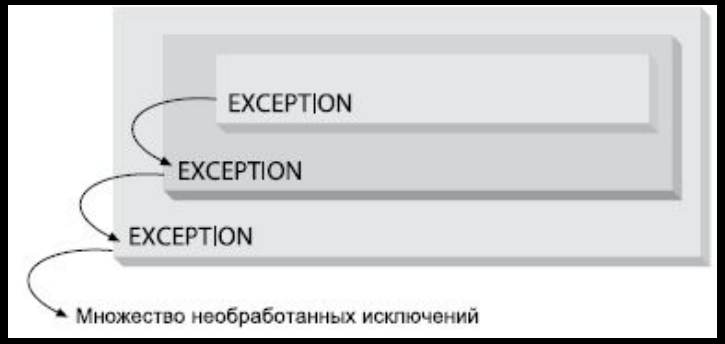
* Демонстрация RAISE vs RAISE_APPLICATION_ERROR

Пример 2

```
...
EXCEPTION
   WHEN NO_DATA_FOUND THEN
    RAISE_APPLICATION_ERROR (-20201,'This is not a
   valid employee');
   END;
```

Передача исключений во внешнюю среду

Сначала PL/SQL ищет обработчик исключения в текущем блоке (анонимном блоке, процедуре или функции). Если такового нет, PL/SQL пытается обработать исключение, инициировав его еще раз в родительском блоке. Так происходит в каждом внешнем по отношению к другому блоке до тех пор, пока все они не будут исчерпаны. После этого PL/SQL возвращает необработанное исключение в среду приложения, выполнившего ≪самый внешний≫ блокPL/SQL.




```
BEGIN
  ...                                                    Вложенный блок 1
  BEGIN
    BEGIN                                                Вложенный блок 2
      SELECT SUM (faults) INTO num_faults FROM profile   ... ;
      IF num_faults > 100
      THEN
        RAISE too_many_faults;
      END IF;
    END;
  END;
EXCEPTION
  WHEN too_many_faults THEN  ... ;
END list_my_faults;
```

Итоги

- Исключения
  - Предопределенные Oracle
  - Не предопределенные исключения
  - Пользовательские
- Инициализация и обработка исключений

Практика №8!

30 минут