Manejo de Errores

Manejo de Excepciones

- Una excepción es un error en PL/SQL que es redireccionado durante la ejecución del programa.
- Una excepción puede ser redireccionada:
  - Implícitamente por Oracle server
  - Explícitamente por el programa
- Una excepción puede ser manejada:
  - Atrapándola con un manejador
  - Propagando su manejo al entorno llamador

Manejo de Excepciones

- Cuando se genera una excepción el control se transfiere a la parte de manejo de excepciones.
- Sólo un manejador de excepción será ejecutado.
- El manejador de excepciones opcional OTHERS debe ser siempre el último en un bloque .

Manejador de Excepciones

```
EXCEPTION
  WHEN excepción1 [OR excepción2 …] THEN
        sentencia1;
        sentencia2;
        …
  [WHEN excepción3 [OR excepción4 …] THEN
        sentencia1;
        sentencia2;
        …]
  [WHEN OTHERS THEN
        sentencia1;
        sentencia2;
        …]
```

Tipos de Excepciones

Excepción	Descripción	Cómo se Manejan
Error predefinido del servidor ORACLE	Uno de los casi 20 errores más comunes en el código PL/SQL	No se declaran. El servidor ORACLE la soporta automáticamente
Error no predefinido del servidor ORACLE	Cualquier otro error estándar reconocido por el servidor ORACLE	Se declara en la sección de declaraciones. El servidor ORACLE la reporta automáticamente
Error definido por el usuario	Una condición que el desarrollador decide que no es normal	Se declara en la sección de declaraciones y se genera en forma explícita

Excepciones Definidas por el Usuario

- Declaración de Excepciones

```
DECLARE
  fecha_vencida EXCEPTION;
  ...
```

- Reglas de Alcance
  - No se puede declarar una excepción dos veces en el mismo bloque.
  - El ámbito de la excepción es igual que el de las variables.

Excepciones Definidas por el Usuario

- Ejemplo:

```
DECLARE
     fecha_vencida  EXCEPTION;
BEGIN
    ...
    DECLARE    ------------------  ---------- comienza el sub-bloque
        fecha_vencida    EXCEPTION;  -- esta declaración prevalece
    BEGIN
        ...
         IF ... THEN
             RAISE   fecha_vencida;  -- esta excepción no es manejada
         END IF;
    END;            --------------------------- finaliza el sub-bloque
EXCEPTION
    WHEN fecha_vencida THEN  -- no maneja la excepción generada
            ...
END;
```

Excepciones Predefinidas de ORACLE

Excepción	Error ORACLE	Descripción
CURSOR_ALREADY_OPEN	ORA-06511	Intento de abrir un cursor ya abierto
DUP_VAL_ON_INDEX	ORA-00001	Intento de inserción de un valor repetido en una columna UNIQUE
INVALID_CURSOR	ORA-01001	Operación de cursor ilegal
INVALID_NUMBER	ORA-01722	En una sentencia SQL, falló la conversión de caracteres a números
NO_DATA_FOUND	ORA-01403	Un SELECT de una fila no retorna datos
TOO_MANY_ROWS	ORA-01422	Un SELECT de una fila retorna varias filas
VALUE_ERROR	ORA-06502	Ocurrió un error aritmético, de conversión, de truncado o de restricción de tamaño
ZERO_DIVIDE	ORA-01476	Intento de dividir por cero

Excepciones Predefinidas de ORACLE

```
PROCEDURE consulta_inventario (
        v_prod_id IN productos.id%TYPE) IS
        v_id    productos.id%TYPE;
BEGIN
        SELECT id INTO v_id FROM productos
            WHERE id = v_prod_id;
EXCEPTION
        WHEN NO_DATA_FOUND THEN
            ...
        WHEN TOO_MANY_ROWS THEN
            ...
        WHEN OTHERS THEN
            ...
END;
```

Excepciones No Predefinidas de ORACLE

- Se utilizan para manejar excepciones internas sin nombre, en lugar del manejador OTHERS.
- Asocian una excepción nominada con un error Oracle determinado.
- Sintaxis:

```
PRAGMA EXCEPTION_INIT(nombre_excepción,
        número_error_ORACLE);
```

Excepciones No Predefinidas de ORACLE

- Ejemplo:

```
DECLARE
   abrazo_mortal_detectado EXCEPTION;
   PRAGMA EXCEPTION_INIT(abrazo_mortal_detectado, -60);
BEGIN
        ...
EXCEPTION
     WHEN abrazo_mortal_detectado THEN
        -- maneja el error
       ...
END;
```

Generación de Excepciones

- A través de la sentencia RAISE se puede generar excepciones en forma explícita.
- Las excenciones predefinidas se pueden generan implísitamente cuando ocurre el error Oracle asociado o en forma explícita.
- Las excepciones definidas por el usuario solo se generan en forma explícita.

Generación de Excepciones (definidas por el usuario)

- Ejemplo:

```
DECLARE
        fuera_de_stock        EXCEPTION;
        cantidad_disponible   NUMBER(4);
BEGIN
        ...
        IF cantidad_disponible < 1 THEN
            RAISE fuera_de_stock;
        END IF;
        ...
EXCEPTION
        WHEN fuera_de_stock THEN
            -- trata el error
END;
```

Generación de Excepciones (predefinidas)

- Ejemplo:

```
DECLARE
      tipo_cuenta INTEGER;
      ...
BEGIN
      ...
       IF tipo_cuenta NOT IN (1, 2, 3) THEN
          RAISE INVALID_NUMBER;  -- predefinida
       END IF;
        ...
EXCEPTION
        WHEN INVALID_NUMBER THEN
              ROLLBACK;
END;cc
```

Procedimiento RAISE_APPLICATION_ ERROR

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

- Permite definir mensajes de error desde subprogramas almacenados.
- Permite reportar errores a las aplicaciones y prevenir excepciones no tratadas.
- TRUE/FALSE es un parámetro opcional Booleano. Si es TRUE, el error es ubicado sobre la pila de errores previos. Si es FALSE (default), el error reemplaza a todos los errores previos
- *error_number el rango permitido va desde -20000 a -20999*

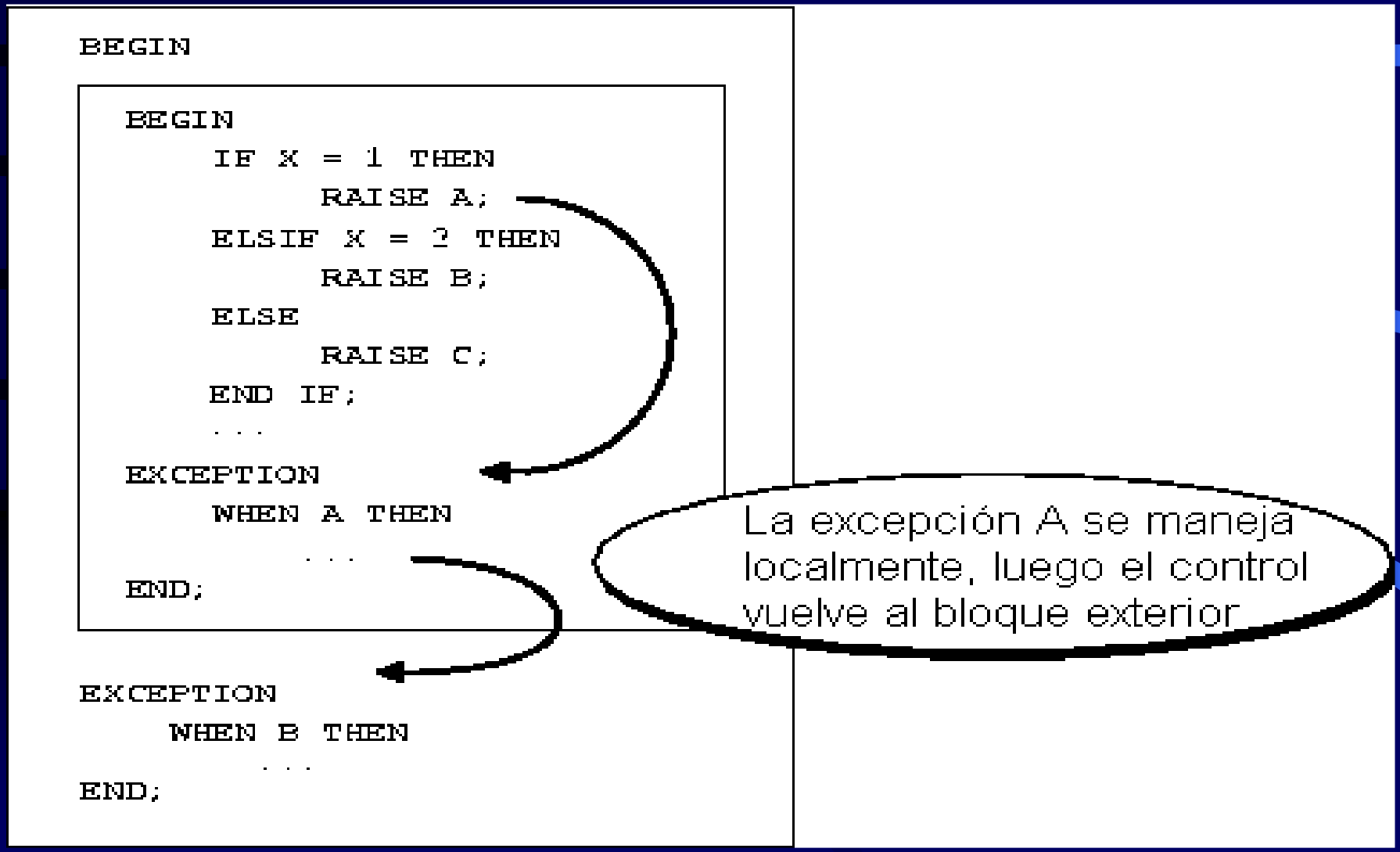
Procedimiento RAISE_APPLICATION_ ERROR

```
CREATE PROCEDURE account_status (
  due_date DATE,
  today    DATE
)
IS
BEGIN
  IF due_date < today THEN                    -- explicitly raise exception
    RAISE_APPLICATION_ERROR(-20000, 'Account past due.');
  END IF;
END;
/

DECLARE
  past_due  EXCEPTION;                        -- declare exception
  PRAGMA EXCEPTION_INIT (past_due, -20000);   -- assign error code to exception
BEGIN
  account_status (TO_DATE('01-JUL-2010', 'DD-MON-YYYY'),
                  TO_DATE('09-JUL-2010', 'DD-MON-YYYY'));    -- invoke procedure

EXCEPTION
  WHEN past_due THEN                          -- handle exception
    DBMS_OUTPUT.PUT_LINE(TO_CHAR(SQLERRM(-20000)));
END;
/
```

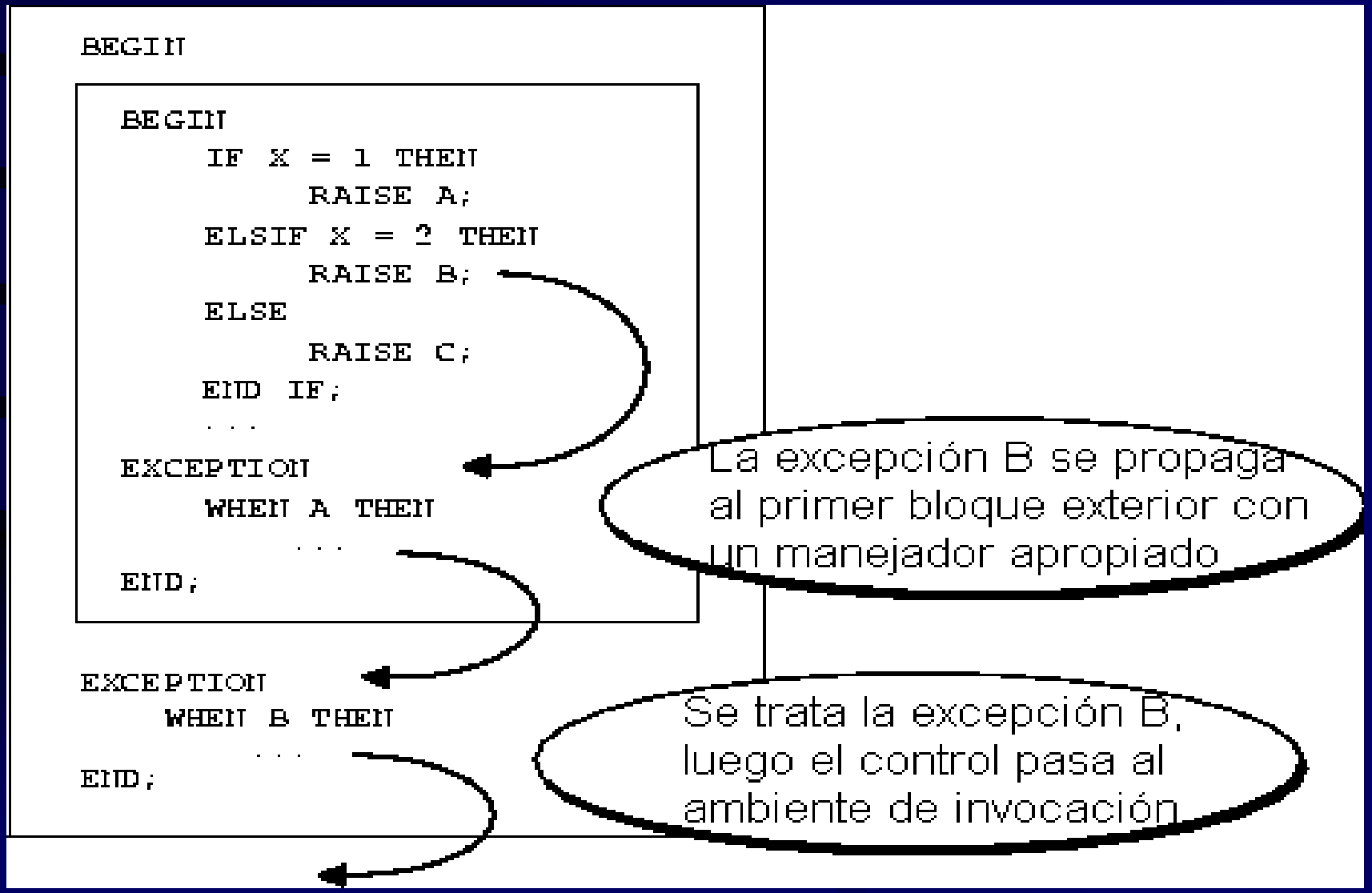
Funciones para la Captura de Errores

```
DECLARE
        nro_err NUMBER;
        msj_err VARCHAR2(100);
BEGIN
        ...
EXCEPTION
        ...
WHEN OTHERS THEN
        nro_err := SQLCODE;
        msj_err := SUBSTR(SQLERRM, 1, 100);
        INSERT INTO errores VALUES (nro_err, msj_err);
END;
```

Propagación de las Excepciones



Propagación de las Excepciones



Propagación de las Excepciones