Paquetes

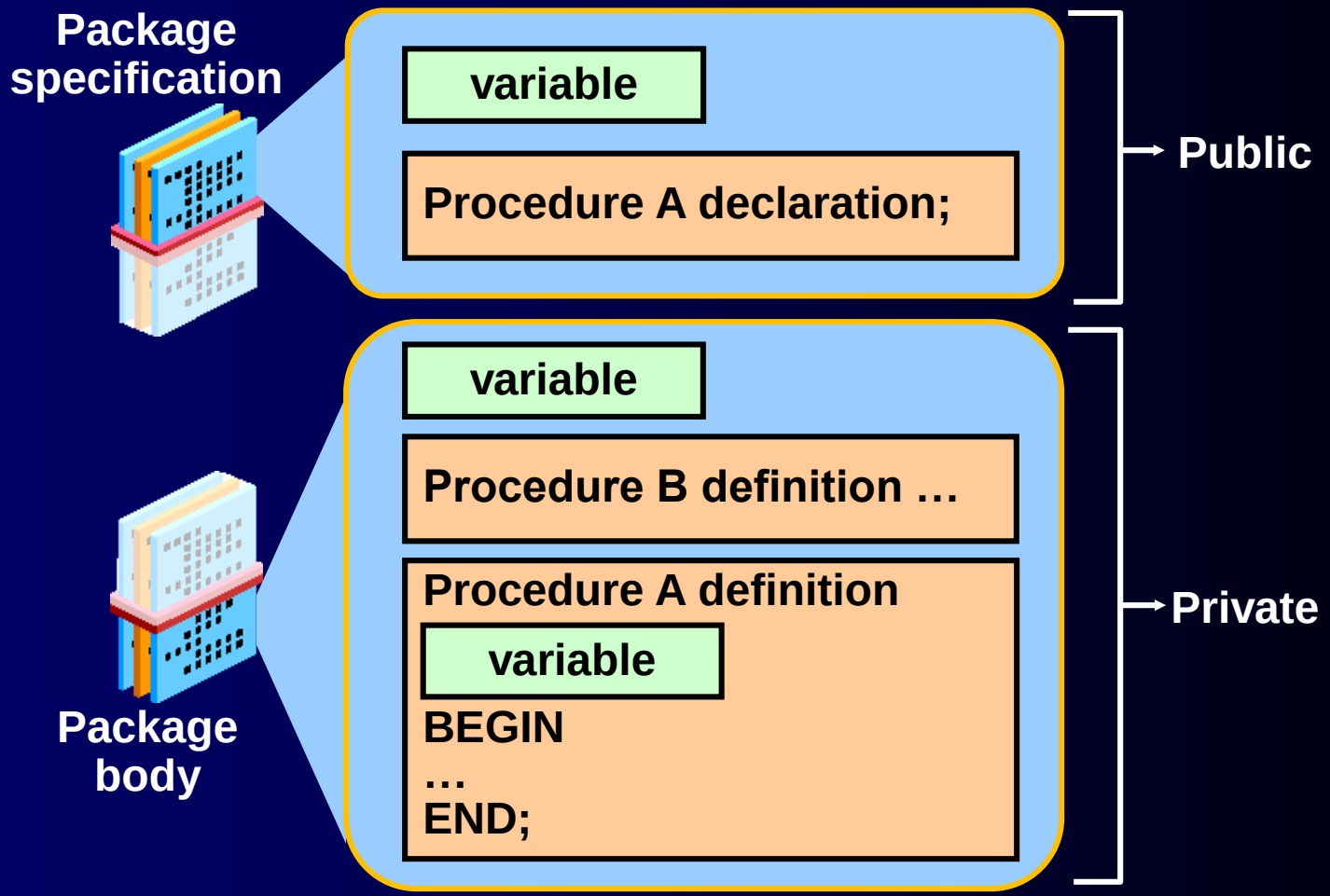
¿Qué es un Paquete?

- Un paquete es un objeto del esquema que agrupa tipos, ítems y subprogramas PL/SQL lógicamente relacionados.
- Permite a Oracle server leer múltiples objetos en memoria simultáneamente (cuando una construcción del paquete es referenciada por primera vez, todo el paquete es ubicado en memoria. Los siguientes accesos a constructores en el mismo paquete no requieren I/O de disco)
- Los paquetes generalmente tienen dos partes: una especificación y un cuerpo.

Componentes de un Paquete PL/SQL

- La **especificación** es la interface a la aplicación; ésta declara tipos, variables, constantes, excepciones, cursores y subprogramas disponibles para usar.
- El **cuerpo** define completamente subprogramas, y así implementa la especificación.

Componentes de un Paquete PL/SQL



Componentes de un Paquete PL/SQL

- **Sintaxis:**

```
CREATE PACKAGE nombre AS  -- especificación (visible)
    -- declaración de tipos e ítems públicos
    -- especificación de subprogramas
END [nombre];

CREATE PACKAGE BODY nombre AS  -- cuerpo (oculta)
    -- declaración de tipos e ítems privados
    -- cuerpos de subprogramas
[BEGIN
    -- sentencias de inicialización]
END [nombre];
```

Ventajas de los Paquetes

- Los paquetes ofrecen varias ventajas:
  - modularidad,
  - diseño de aplicaciones más fácil,
  - ocultamiento de información,
  - funcionalidad agregada, y
  - mejor performance.

Ventajas de los Paquetes

- Modularidad:

Los paquetes permiten encapsular tipos, ítems y subprogramas lógicamente relacionados en un módulo PL/SQL nombrado.

Cada paquete es fácil de entender, y las interfaces entre los paquetes son simples, claras y bien definidas.

Ventajas de los Paquetes

- Diseño de Aplicaciones Más Fácil:

Cuando se diseña una aplicación, todo lo que se necesita inicialmente es información de la interfaz en las especificaciones del paquete.

Es posible codificar y compilar una especificación sin su cuerpo.

Ventajas de los Paquetes

- Ocultamiento de Información:

Con los paquetes, es posible especificar qué tipos, ítems y subprogramas son públicos (visibles y accesibles) o privados (ocultos e inaccesibles).

El paquete oculta la definición del subprograma privado de tal manera que sólo el paquete (no la aplicación) se ve afectado si cambia la definición.

Ventajas de los Paquetes

- Funcionalidad Agregada:

Las variables y cursores públicos empaquetados persisten por la duración de una sesión.

De esta manera, todos los subprogramas que ejecutan en el entorno pueden compartirlos.

Ventajas de los Paquetes

- Mejor Performance:

Cuando se llama a un subprograma empaquetado por primera vez, el paquete completo se carga en memoria. De esta manera, llamadas posteriores a subprogramas en el paquete mencionado no requieren entrada / salida de disco.

La Especificación de Paquete

- La especificación del paquete contiene declaraciones públicas.
- El alcance de estas declaraciones es local al esquema de la base de datos y global al paquete.
- La especificación lista los recursos del paquete disponible para las aplicaciones.

La Especificación de Paquete

- Por ejemplo, la siguiente declaración muestra que la función llamada factorial toma un argumento de tipo INTEGER y devuelve un valor de tipo INTEGER:

```
FUNCTION factorial (n INTEGER) RETURN INTEGER;
   -- devuelve n!
```

- No es necesario considerar la implementación subyacente de factorial (por ejemplo, si es iterativa o recursiva).

Referencias a los Contenidos de Paquetes

- Para referenciar los tipos, ítems, y subprogramas declarados en la especificación de un paquete, se utiliza la notación de punto, como sigue:
  - nombre_paquete.nombre_tipo
  - nombre_paquete.nombre_item
  - nombre_paquete.nombre_subprograma

El Cuerpo de Paquete

- El cuerpo del paquete implementa la especificación del paquete.
- El cuerpo del paquete contiene la definición de cada cursor y subprograma declarado en la especificación del paquete.
- Los subprogramas definidos en el cuerpo de un paquete sólo son accesibles desde fuera del mismo si sus especificaciones también aparecen en la especificación del paquete.

El Cuerpo de Paquete

- El cuerpo del paquete también puede contener declaraciones privadas, que definen tipos e ítems necesarios para el funcionamiento interno del paquete.
- El alcance de estas declaraciones es local al cuerpo del paquete.
- Los tipos e ítems declarados son inaccesibles excepto desde dentro del cuerpo del paquete.

El Cuerpo de Paquete

- A continuación de la parte declarativa del cuerpo de un paquete está la parte de inicialización, opcional, que generalmente contiene sentencias que inicializan algunas de las variables previamente declaradas en el paquete.
- La parte de inicialización de un paquete se corre una sola vez, la primera vez que se referencia al paquete dentro de una sesion.

El Cuerpo de Paquete

- Si la especificación de un paquete declara sólo tipos, constantes, variables y excepciones, el cuerpo del paquete es innecesario.
- Sin embargo, el cuerpo aún se puede usar para inicializar ítems declarados en la especificación.

Sobrecarga (Overloading)

- PL/SQL permite que dos o más subprogramas empaquetados tengan el mismo nombre, siempre que sus parámetros formales difieran en número, orden o tipos de datos.
- Esta opción es útil cuando se desea que un subprograma acepte parámetros que tienen diferentes tipos de datos.

Sobrecarga (Overloading)

- Dada la siguiente declaración:

```
DECLARE
    TYPE FechaTabTipo IS TABLE OF DATE INDEX BY BINARY_INTEGER;
    TYPE RealTabTipo IS TABLE OF REAL INDEX BY BINARY_INTEGER;
    fechacontrato_tab       FechaTabTipo;
    sal_tab   RealTabTipo;
```

Sobrecarga (Overloading)

- Se podría escribir el siguiente procedimiento:

```
PROCEDURE inicializar (tab OUT FechaTabTipo, n INTEGER)
   IS
BEGIN
   FOR i IN 1..n LOOP
     tab(i) := SYSDATE;
   END LOOP;
END inicializar;
```

Sobrecarga (Overloading)

- Y otro procedimiento:

```
PROCEDURE inicializar (tab OUT RealTabTipo, n
    INTEGER) IS
BEGIN
    FOR i IN 1..n LOOP
      tab(i) := 0.0;
    END LOOP;
END inicializar;
```

Sobrecarga (Overloading)

- Podría utilizarse de la siguiente forma:

```
DECLARE
  TYPE FechaTabTipo IS TABLE OF DATE ...;
  TYPE RealTabTipo IS TABLE OF REAL ...;
  fechacontrato_tab    FechaTabTipo;
  comision_tab         RealTabTipo;
  indice          BINARY_INTEGER; ...
BEGIN
  indice := 50;
  inicializar(fechacontrato_tab, indice);  -- primer versión
  inicializar(comision_tab, indice);  -- segunda versión
END;
```

Sobrecarga (Overloading)

- Restricciones:
  - Sólo los subprogramas locales o empaquetados se pueden sobrecargar.
  - No es posible sobrecargar subprogramas stand-alone.
  - Tampoco se pueden sobrecargar dos subprogramas si sus parámetros formales difieren sólo en el nombre o el modo del parámetro.

Sobrecarga (Overloading)

- Restricciones:
  - Además, no se pueden sobrecargar dos programas si sus parámetros formales difieren sólo en el tipo de datos y los diferentes tipos de datos son de la misma familia.
  - No se pueden sobrecargar dos funciones que difieren sólo en el tipo devuelto (el tipo de dato del valor resultado) aún si los tipos son de diferentes familias.