BASES DE DATOS UD-7

MODELO OBJETO-RELACIONAL

ÍNDICE

[INTRODUCCIÓN 3](#_Toc135125374)

[1) SGBD-OR 3](#_Toc135125375)

[2) Características 3](#_Toc135125376)

[3) Modelo relacional anidado 4](#_Toc135125377)

[4) Vistas útiles 4](#_Toc135125378)

[5) Permisos necesarios 4](#_Toc135125379)

[6) Tipos de datos 5](#_Toc135125380)

[TIPOS DE DATOS MULTIVALUADOS 5](#_Toc135125381)

[A. ARRAY (COLECCIÓN) 5](#_Toc135125382)

[1) Definición 5](#_Toc135125383)

[2) Creación y borrado 5](#_Toc135125384)

[3) Manipulación y consulta 6](#_Toc135125385)

[B. TABLA ANIDADA (*NESTED TABLE)* 7](#_Toc135125386)

[1) Definición 7](#_Toc135125387)

[2) Creación y borrado 8](#_Toc135125388)

[3) Manipulación y consulta 9](#_Toc135125389)

[ORIENTACIÓN A OBJETOS 9](#_Toc135125390)

[A. TIPO OBJETO 9](#_Toc135125391)

[1) Definición 9](#_Toc135125392)

[2) Creación y borrado 10](#_Toc135125393)

[3) Uso del tipo objeto en un campo de una tabla que sólo tenga ese campo 10](#_Toc135125394)

# INTRODUCCIÓN

1. SGBD-OR

En los años noventa irrumpieron los lenguajes de programación orientados a objetos frente a la programación estructurada. Esto hizo que aparecieran las bases de datos orientadas a objetos, que organizaron los datos de forma adecuada para poder ser manipulada con facilidad por estos nuevos lenguajes de programación.

Los Sistemas Gestores de Bases de Datos Orientados a Objetos puros no son demasiados. Sin embargo, los desarrolladores de los SGBD Relacionales consideraron adecuada la incorporación de ciertos conceptos de orientación a objetos en sus sistemas dando lugar así a los SGBD Objeto-Relacionales (SGBDOR).

Algunos SGBDOR:

* Oracle.
* Microsoft SQL Server.
* IMB DB2.
* PostgreSQL.
* SAP.
* IBM Informix.

A pesar de que los SGBDOR han ampliado su repertorio de objetos, el estándar SQL:1999 ya incluía sus elementos básicos, en los que nos vamos a centrar.

La principal diferencia entre las BBDD-R y las BBDD-OR radica en la existencia de tipos de datos. Estos tipos se pueden construir mediante sentencias DDL y serán la base del desarrollo de BBDD-OR.

1. Características
2. Facilita el manejo de datos desde los lenguajes de programación orientados a objetos.

Es posible recuperar datos de la base de datos para instanciar objetos de programación, y recorrerlos mediante los atributos, métodos y funciones apropiados.

1. Incorpora campos multivaluados.

Se pueden definir tipos de datos complejos (colecciones, tablas anidadas y objetos), con severas consecuencias para el modelo relacional subyacente puesto que estos tipos de datos son considerados multivaluados y por tanto contradicen frontalmente la primera forma normal (1FN), al no garantizar la atomicidad de los atributos.

Tabla

Descripción generada automáticamente

Los tipos de datos pueden representar una estructura, las tablas dejan de ser bidimensionales y pueden incluir nuevas dimensiones de profundidad de la información.

Tabla

Descripción generada automáticamente

1. Modelo relacional anidado

Dado que el modelo objeto-relacional permite que los valores de un campo sean registros de otra tabla (que, a su vez, podría contar con registros de una tercera tabla como valores de sus propios campos), esta forma de almacenar la información se conoce como modelo relacional anidado. Gracias a esta característica se pueden eliminar tablas relacionadas del modelo relacional clásico y simplificar las sentencias SELECT de consulta, mejorando la vinculación con un lenguaje de programación orientado a objetos.

1. Vistas útiles

En Oracle, se puede consultar la vista user\_types para examinar los tipos que se han creado en el SGBD.



1. Permisos necesarios

Para definir un tipo de dato en Oracle es necesario tener permiso de creación de tipos (create any type), además de todos los permisos necesarios para manipularlo:

Diagrama

Descripción generada automáticamente

1. Tipos de datos

Interfaz de usuario gráfica, Diagrama, Texto

Descripción generada automáticamente

# TIPOS DE DATOS MULTIVALUADOS

## ARRAY (COLECCIÓN)

1. Definición

Un array o colección es una lista de elementos a los que se puede acceder mediante su posición. Esta lista es de tamaño fijo (tiene un número determinado de elementos). Por tanto, si un campo es de tipo array permitirá asignar varios valores: es multivaluado.

Diagrama

Descripción generada automáticamente

1. Creación y borrado

La sintaxis para la creación de un tipo array indicándole el número de elementos que contendrá:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

La sintaxis para la creación de una tabla con un campo tipo array es:

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Nota: a un campo de tipo array no se le pueden aplicar restricciones como ser clave primaria.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Para el borrado de un tipo array:



1. Manipulación y consulta

Para consultar, insertar, borrar o actualizar información en tablas con campos de tipo colección es necesario utilizar el constructor del tipo:

Logotipo

Descripción generada automáticamente con confianza baja

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Es importante destacar que desde una consulta SQL, los valores de un array (varray) solamente pueden ser accedidos y recuperados en un bloque. Es decir, no se puede acceder a los elementos de un array individualmente.

Sin embargo, desde un programa PL/SQL sí que es posible definir un bucle que itere sobre los elementos de un array. Para ello:

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

## TABLA ANIDADA (*NESTED TABLE)*

1. Definición

Una tabla anidada es un tipo especial de tabla que permite introducir una lista de elementos a los que se puede acceder mediante su posición. Esta lista es de tamaño variable (se puede insertar un número indefinido de registros en la tabla anidada). El tamaño de cada elemento sí es fijo.

Por tanto, si un campo es de tipo tabla anidada asignar varios valores: es multivaluado.

Diagrama

Descripción generada automáticamente

1. Creación y borrado

La sintaxis para la creación de un tipo de tabla:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Cuando se utilice este tipo de datos en un campo de una tabla, dado que a priori no se conoce el espacio que ocuparán (es una lista ilimitada de elementos), será necesario definir una “tabla almacén” en la que se almacenarán los datos de ese campo. Esta “tabla almacén” es una tabla especial denominada tabla anidada utilizada para este fin, no es necesario crearla como una tabla usual con CREATE TABLE.

La sintaxis para la creación de una tabla con un campo de tipo tabla es:

Texto, Escala de tiempo

Descripción generada automáticamente

Nota: vamos a trabajar con un solo campo de tipo tabla.

Texto

Descripción generada automáticamente

Para el borrado de un tipo tabla anidada:



1. Manipulación y consulta

Para consultar, insertar, borrar o actualizar información en tablas con campos de tipo tabla anidada, al igual que con los arrays, es necesario utilizar el constructor del tipo:

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Es importante destacar que desde una consulta SQL, los valores de una tabla anidada solamente pueden ser accedidos y recuperados en un bloque. Es decir, no se puede acceder a los elementos de un array individualmente.

Sin embargo, desde un programa PL/SQL sí que es posible definir un bucle que itere sobre los elementos de una tabla anidada. Para ello se puede seguir el mismo procedimiento que en el caso de los arrays.

# ORIENTACIÓN A OBJETOS

## TIPO OBJETO

1. Definición

Recordemos que en programación orientada a objetos (POO), una clase es como el molde o un patrón para la fabricación de los objetos. Por tanto, un objeto es una instancia de una clase.

En Oracle, se pueden crear clases a través de los tipos, más concretamente de los tipos de objetos, usando la sentencia create type as object. Gracias a este tipo objeto se puede agregar información a la base de datos anidando la información en una única fila en tantos niveles como se desee.

Diagrama

Descripción generada automáticamente

1. Creación y borrado

Un tipo de objeto puede contener atributos, procedimientos y funciones. La sintaxis básica para la creación de un tipo objeto es:

Interfaz de usuario gráfica, Aplicación, Escala de tiempo

Descripción generada automáticamente

Para instanciar un objeto, se emplea el constructor:



Para el borrado de un tipo objeto:



Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

1. Uso del tipo objeto en un campo de una tabla que sólo tenga ese campo

Es posible definir una tabla que solo contenga un campo de tipo objeto. Este es un caso simplificado en el que se manipulan los datos de la tabla como si los atributos del tipo de objeto fueran directamente atributos de la tabla.

Imagen que contiene Texto

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Texto

Descripción generada automáticamente con confianza media

1. Uso del tipo objeto en un campo de una tabla

Es posible definir un campo de una tabla como un objeto. En este caso cada registro contendrá un solo objeto dentro de ese campo.

Diagrama

Descripción generada automáticamente

Para manipular los datos de tipo objeto de esta tabla es necesario emplear un alias de tabla y acceder al dato:

Texto

Descripción generada automáticamente

1. Uso del tipo objeto en una tabla anidada

Es posible definir un campo de una tabla como una tabla anidada en la que cada elemento de esa tabla anidada sea un objeto.

Diagrama

Descripción generada automáticamente

Pasos:

1. Creación del tipo Objeto.
2. Creación de tabla anidada donde guardaremos el tipo objeto (tipo tabla).
3. Creación de la tabla final.

Texto

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

## MÉTODOS

Hasta ahora solo hemos utilizado los tipo objeto para definir tipos de datos compuestos pero, al igual que en la POO, cada clase define el comportamiento de sus objetos a través de métodos, en las BBDD-OR también es posible crear métodos para los tipo objeto.

Se pueden definir funciones o procedimientos miembros, cuyas acciones modelan el comportamiento de un tipo de objeto. Estas funciones o procedimientos miembros que pertenecen a un tipo de objeto se denominan métodos.

Los tipos de objeto que contienen métodos, deben declarar un BODY que contendrá el código de todos los métodos.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Cuando un tipo de objeto no tiene métodos, no es necesario declarar su BODY.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

## CONSTRUCTOR

Un tipo especial de métodos serán los constructores que serán funciones miembros del tipo de objeto y se crearán de forma automática. Para referirse a los atributos miembros se utilizará la palabra SELF (equivalente a this en JAVA).

Estos constructores tienen el mismo nombre que el tipo objeto, y devolverán el objeto creado.

Hay que tener en cuenta que cuando se insertan datos (INSERT INTO), realmente se está llamando al constructor por defecto.

Texto

Descripción generada automáticamente con confianza media

Texto

Descripción generada automáticamente

## HERENCIA

Una de las grandes ventajas de la POO es la herencia, gracias a la cual se pueden crear superclases abstractas para después, crear subclases más específicas que hereden los atributos y métodos de las superclases.

En BBDDO-R es posible hacer algo parecido con los tipos de objetos: se pueden crear subtipos de objetos a partir de otros super tipos de objetos creados previamente.

* **Supertipo de objeto**: se declaran con la cláusula NOT FINAL.

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

* **Subtipo de objeto**: hay que indicar de que supertipo desciende con la cláusula UNDER.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Texto, Correo electrónico

Descripción generada automáticamente