

La definición de datos. Niveles
de descripción. Lenguajes.
Diccionario de datos.

TEMA 35

ABACUS NT

Oposiciones 2021

Índice

- 1. Introducción**
- 2. Lenguajes para la definición y manipulación de datos**
 - 2.1. Lenguajes para la definición de datos**
 - 2.2. Lenguajes para la manipulación de datos**
 - 2.3. Características**
- 3. Lenguaje de definición de datos (DDL).**
- 4. Niveles de descripción de datos**
- 5. Arquitecturas de referencia**
 - 5.1. Niveles de abstracción**
 - 5.2. Nivel interno (estructura física).**
 - 5.3. Nivel conceptual (estructura lógica global)**
 - 5.4. Nivel externo (estructura lógica de usuario)**
 - 5.5. Interfaces**
- 6. Diccionario de recursos de información**
 - 6.1. Tipos de almacenes de datos: Diccionario de Datos y Directorio de Datos**
- 7. Conclusión**
 - 7.1. Relación del tema con el sistema educativo actual**
- 8. Bibliografía**

1. Introducción

La función de descripción o definición debe permitir al administrador de la base de datos especificar los elementos que la integran, su estructura y las relaciones que existen entre ellos, las reglas de integridad de semántica, los controles a efectuar antes de autorizar el acceso a la base, etc., así como las características de tipo físico y las vistas lógicas de los usuarios.

Esta función, realizada por el lenguaje de descripción o definición de datos (DDL) propio de cada SGBD, debe suministrar los medios para definir las tres estructuras de datos (**externa, lógica global e interna**), especificando las características de los datos en cada uno de estos tres niveles.

El DDL más conocido es SQL (Structured Query Language). Es un lenguaje **declarativo** que **incluye** tanto el DML, como el DDL. Hoy en día es un **estándar de facto**, y es utilizado por la mayoría de los SGBD relacionales comerciales, como Oracle, o MySQL, entre otros.

Pero SQL, también es un **estándar de iure**. Así, en 1986, ANSI (American National Standards Institute) e ISO (International Standards Organization) elaboran un estándar común para el lenguaje SQL, el SQL86 o SQL1. En 1992 se mejoró significativamente, obteniéndose SQL92 o SQL2, y en 1999, se publicó un nuevo estándar, SQL99 o SQL3 , que incluye, según los autores Abraham Silberschatz, Henry F. Korth, y S. Sudarshan, en su libro Fundamentos de bases de datos (2014), las siguientes características para el soporte a la orientación a objetos:

- Relaciones anidadas
- Tipos complejos
- Herencia
- Tipos de referencia
- Funciones y procedimientos

2. Lenguajes para la definición y manipulación de datos

Según el autor **C.J Date**, en su libro Introduction to Database Systems (2003), “una base de datos es un conjunto de datos persistentes que es utilizado por los sistemas de aplicación de alguna empresa dada” (entendiéndose como empresa una organización).

Para trabajar con dicho conjunto de datos, la arquitectura ANSI/X3/SPARC diferencia dos tipos de lenguajes:

- **Lenguaje de definición de datos:**
- **Lenguaje de manipulación de datos**

A continuación, vamos a ver en qué consiste cada uno de ellos

2.1. Lenguajes para la definición de datos

El lenguaje de descripción o definición de datos (DDL, Data Definition Language), propio de cada SGBD, permite al desarrollador de la base de datos especificar los elementos de datos que integran su estructura, y las relaciones que existen entre ellos, las reglas de integridad semántica, las características de tipo físico y las vistas lógicas de los usuarios.

Generalmente, los DDL de los diferentes SGBD son lenguajes muy simples basados en una gramática muy sencilla.

La representación de los datos compilación es almacenada en denominado **Diccionario de Datos**.

el DDL cuenta con un **sublenguaje** encargado del control y seguridad de los datos, el cual se denomina **Lenguaje de Control de Datos (DCL)** y permite el control del acceso de a la información almacenada en el diccionario de datos (definición de privilegios y tipos de acceso), así como el control de seguridad de los datos.

2.2. Lenguajes para la manipulación de datos

Una vez descrita la estructura de la base de datos, los usuarios utilizarán un lenguaje de manipulación de datos (DML, Data Manipulation Language) para realizar las siguientes operaciones sobre dicha estructura:

- **Recuperar información:** Consultar la base de datos.
- **Actualizar la información:** La actualización supondrá tres tipos de operaciones
 - Inserción
 - Borrado
 - Modificación de los datos.

Al igual que el DDL, el DML está basado en un modelo de datos y, por tanto, los SGBD basados en distintos modelos de datos tienen diferente DML. Se trata también de un lenguaje basado en una gramática completa, sencilla y, generalmente, fácil de entender por usuarios no expertos.

Los SGBD también son capaces de procesar peticiones de DML que se han formulado desde programas escritos en otros lenguajes de programación

2.3. Características

Como hemos visto anteriormente, para trabajar con las bases de datos vamos a tener dos tipos de lenguajes: los DML y los DDL, éstos, a su vez, se pueden clasificar atendiendo a las siguientes características:

- **Embebido/autocontenido:** Un lenguaje autocontenido no necesita de otro, mientras que un lenguaje embebido se inserta en el seno de un programa escrito en otro lenguaje, denominado anfitrión.

- **Más o menos procedimental:** Un lenguaje de manipulación de datos es más procedimental, cuanto con más detalles es preciso especificar el procedimiento necesario para acceder a la base de datos.
- **Diferido/Conversacional:** Algunos DML se utilizan en modo diferido, esto es, en tratamiento por lotes, pero en la actualidad la mayoría permiten su uso en modo conversacional, es decir, interactivo, desde un terminal, o interfaz gráfica.
- **Navegacional/Especificación:** Existen DML llamados navegacionales que recuperan o actualizan los datos registro a registro, debiéndose indicar el camino que se ha de recorrer, hasta llegar al registro buscado. Otros, actúan sobre el conjunto de registros, como por ejemplo SQL.
- **Compilación/Interpretación:** Los lenguajes de compilación necesitan de un compilador, el cual se encarga de generar código máquina a partir de un código fuente. Por el contrario, un lenguaje interpretado es el que es ejecutado paso a paso, sin ser necesaria una traducción previa a la ejecución.

3. Lenguaje de definición de datos (DDL).

Cualquier lenguaje de bases de datos está formado por un DDL y por un DML. Existe un lenguaje considerado como estándar para manipular bases de datos relacionales: SQL (Structured Query Language). Estudiaremos el DDL y el DML que utiliza el SQL. Las características principales de SQL son su sencillez, su carácter estándar y ser un lenguaje declarativo o no procedural (para que SQL realice una acción concreta no se le dice cómo tiene que hacerla, sino lo que queremos obtener).

El lenguaje de definición de datos (DDL) proporciona órdenes para definir, eliminar y modificar tablas, así como para crear índices y vistas. Las órdenes SQL más importantes del DDL son:

Creación de una tabla

CREATE sirve para crear objetos de la base de datos, entre estos objetos tenemos tablas, vistas etc.

```
CREATE TABLE nombre_tabla
(nombrecoll tipocoll
[CONSTRAINT nombre_restricción]
[not NULL]
[PRIMARY KEY]
[UNIQUE]
[DEFAULT valor]
[check <condici>]
[REFERENCES Nombre_tabla_ref (colref) [ON DELETE CASCADE]] ,...
[Restricciones de la tabla]
)
[tablespace nombre-tablespace];
```

Donde:

- **Nombre tabla:** Es el nombre de la nueva tabla. Debe ser único dentro de la BD y además debe identificar su contenido, el nombre de la tabla puede ser una cadena de 1 a 30 caracteres alfanuméricos (0-9, a-z, subrayado, \$, #) empezando siempre por un carácter alfabético.
- **nombreCol:** Es el nombre de una columna de la tabla. Los nombres de columna deben cumplir las reglas de los identificadores y deben ser únicos en la tabla.
- **tipoCol:** Especifica el tipo de datos de la columna. Se permiten los tipos de datos del sistema o definidos por el usuario. Especifica el tipo de datos de la columna. Se permiten los tipos de datos del sistema o definidos por el usuario.

Cuando almacenamos datos las tablas se ajustan a una serie de restricciones predefinidas, por ejemplo, que una columna no pueda tomar valores negativos o que una columna tenga que almacenarse en mayúsculas.

La integridad hace referencia al hecho de que los datos de la BD han de ajustarse a restricciones antes de almacenarse en la BD y una restricción de integridad será una regla que restringe el rango de valores de una o más columnas de una tabla.

Existe otro tipo de integridad que es la integridad referencial, que garantiza que los valores de una o varias columnas de una tabla dependan de los valores de otro o otras columnas de otra tabla.

Las restricciones se definen dentro de la orden CREATE TABLE y para ello se utiliza la cláusula CONSTRAINT. Una vez creadas las restricciones se pueden añadir, modificar o borrar a través de la orden ALTER TABLE.

Una cláusula CONSTRAINT puede restringir una sola columna, se habla en este caso de restricción de columna o puede restringir un grupo de columnas de una tabla, en este caso se llama restricción de tabla.

- **CONSTRAINT.** Es una palabra clave opcional que indica el principio de la definición de una restricción para la columna o tabla: PRIMARY KEY, NOT NULL, UNIQUE, FOREIGN KEY o CHECK. Las restricciones son propiedades especiales que exigen la integridad de los datos y pueden crear índices para la tabla y sus columnas.
- **nombre_restricción.** Los nombres de restricción deben ser únicos en una base de datos.
- **NULL | NOT NULL.** Son palabras clave que determinan si se permiten o no valores NULL en la columna. Si la restricción es NOT NULL significa que dicha columna no puede tener valores nulos, es decir, ha de tener un valor obligatoriamente; en caso contrario causa una excepción.
- **PRIMARY KEY.** Es una restricción que indica qué columna o columnas formarán la clave primaria de la tabla. Sólo se puede crear una restricción PRIMARY KEY por cada tabla.
- **UNIQUE.** Es una restricción que proporciona la integridad de entidad para la columna o columnas indicada a través de un índice único. Una tabla puede tener varias restricciones UNIQUE.

- **DEFAULT.** Especifica el valor suministrado para la columna cuando no se ha especificado explícitamente un valor durante la inserción.
- **REFERENCES.** Es una restricción que proporciona integridad referencial para los datos de una columna o columnas. Las restricciones REFERENCES requieren que cada valor de la columna o columnas existan en la columna o columnas de referencia correspondiente de la tabla a la que se hace referencia. Las restricciones REFERENCES pueden hacer referencia sólo a columnas que sean restricciones PRIMARY KEY en la tabla de referencia.
- **Nombre_tabla_ref.** Es el nombre de la tabla a la que hace referencia la restricción REFERENCES.
- **(colref [...n]).** Es una columna o lista de columnas de la tabla a la que hace referencia la restricción REFERENCES.
- **ON DELETE CASCADE.** Especifica qué acción tiene lugar en una fila de la tabla creada, si esa fila tiene una relación referencial y la fila a la que hace referencia se elimina en la tabla primaria. En nuestro caso si se elimina una fila de la tabla primaria, también se eliminan las filas de la tabla desde donde se hace referencia. Cuando la restricción de Integridad Referencial se realiza sobre la definición de un campo en la sentencia CREATE TABLE solo se utiliza la cláusula REFERENCES, no se utiliza la cláusula FOREIGN KEY; esta última se utiliza cuando la restricción se crea a nivel de tabla.
- **CHECK.** Es una restricción que exige la integridad del dominio al limitar los valores posibles que se pueden escribir en una o varias columnas.

Borrado de una tabla (estructura y datos)

```
Drop table nombre_tabla [CASCADE CONSTRAINT] ;
```

Al borrar una tabla, se borra tanto su estructura como sus datos, sus índices asociados y los privilegios concedidos sobre estas también se borran, las vistas creadas directa o indirectamente sobre esta tabla son desactivadas de forma automática por ORACLE, pero no borradas.

Cada usuario puede borrar sus propias tablas, pero no puede borrar las de otro usuario al menos que tenga concedido un permiso adecuado. Si se hace referencia a la clave primaria de esta tabla mediante restricciones FOREIGN KEY o REFERENCES, la cláusula CASCADE CONSTRAINT permite suprimir estas restricciones de integridad referencial en las tablas 'descendientes'.

Borrado de los registros de una tabla

Con la orden TRUNCATE se eliminan todas las filas de una tabla y se puede liberar espacio utilizado por esta tabla. Es una orden del lenguaje DDL y por tanto no se puede anular. Tampoco activa disparadores DELETE por lo que es más rápido que una orden DELETE. Su sintaxis es:

```
Truncate table nombre_table [{DROP | REUSE} STORAGE] ;
```

Con DROP STORAGE se desasigna todo el espacio. Con DROP REUSE mantendrá reservado el espacio para nuevas filas.

Modificar la estructura de una tabla

Para modificar la estructura de una tabla se utiliza el comando ALTER TABLE.

```
ALTER TABLE Tabla
{ [ADD      ( Columna Tipodato [restricción de columna] [...] ) ]
[MODIFY ( Columna Tipodato [restricción de columna] [...] ) ]
[ADD CONSTRAINTS restricción]
[DROP CONSTRAINTS restricción] };
```

Añadir o modificar columnas (nombre, tipo, valor por defecto, restricción NOT NULL)

```
alter table nombre_table {ADD | MODIFY} ( columna tipo [restricción,...])
```

Eliminación de columnas

```
alter table nombre_table DROP COLUMN nombre_columna
```

Añadir restricción de tabla

```
alter table nombre_tabla ADD CONSTRAINT nombre_restricción restricción
```

Eliminar una restricción.

```
alter table nombre_table DROP CONSTRAINT nombre_restricción
```

Activación y desactivación de una restricción.

```
alter table nombre_table [ENABLE VALIDATE|ENABLE NOVALIDATE|DISABLE] nombre_restricción
```

Donde:

- ENABLE VALIDATE activa la restricción si el conjunto de filas ya presentes en la tabla cumplen dicha restricción.
- ENABLE NOVALIDATE activa la restricción para las siguientes instrucciones de manipulación de datos.

- DISABLE desactiva la restricción.

Hay que tener en cuenta que, si la tabla está vacía, al añadir una columna con la restricción NOT NULL no habrá ningún error, pero si tiene filas no permitirá añadir una columna con esta opción.

VISTAS

Las vistas son tablas virtuales ‘que contienen’ el resultado de una consulta SELECT, tienen la misma estructura que una tabla cualquiera, es decir, están organizadas por filas y columnas. Una de las principales ventajas de utilizar vistas procede del hecho de que la vista no almacena los datos, sino que hace referencia a una o varias tablas de origen mediante una consulta SELECT, consulta que se ejecuta cada vez que se hace referencia a la vista. De esta forma, cualquier modificación que se realice sobre los datos de las tablas de origen es inmediatamente visible en la vista, cuando ésta vuelve a ejecutarse. Su sintaxis es:

```
CREATE [OR REPLACE] VIEW Nombre_vista
[ (Lista de columnas) ]
AS SELECT [...]
```

La opción REPLACE, lo que hace es, reemplazar la vista en el caso de que esta ya exista. Las vistas se utilizan de forma análoga a las tablas, permitiendo realizar consultas sobre las vistas, también se pueden realizar sentencias DML sobre las vistas, sin embargo, las modificaciones, borrados e inserciones están restringidas a vistas que estén definidas sobre una única tabla.

Borrado de una vista.

La orden para borrar una vista es DROP VIEW. Su sintaxis es:

```
DROP VIEW Nombre_vista
```

Creación de un índice

Los índices sirven para mejorar el rendimiento de las consultas. El optimizador de Oracle los utiliza implícitamente y se actualizan de forma automática al actualizar las filas.

En general, los índices se crean sobre todas las claves externas y sobre los criterios de búsqueda actuales.

```
CREATE [unique] INDEX nombre_indice
ON nombre_tabla (columnas [{asc | desc}] [,.....])
[TABLESPACE Nombre_Tablespace]
```

Borrado de un índice

Cuando se borra una tabla, automáticamente se borran los índices asociados a ella. Los índices ocupan espacio dentro de la BD como si de una tabla se tratara y por esa razón se aconseja tener solo como índices aquellas columnas por las cuales se realizan consultas de forma periódica. Para borrar un índice se utiliza la orden:

```
drop index nombre_index;
```

4. Niveles de descripción de datos

4.1. Niveles de abstracción

Un **SGBD** es un conjunto de procedimientos, ayudas de documentación, lenguajes y programas de software que administran los ficheros de la base de datos.

Uno de los *objetivos* de un SGBD es proporcionar a los usuarios una visión abstracta de la información, es decir, ocultar ciertos detalles referentes a la forma en que los datos se almacenan y mantienen, pero siempre permitiendo una recuperación eficaz de la información.

La independencia de datos es la capacidad para modificar el esquema de un nivel del sistema sin tener que modificar el esquema del nivel inmediato superior. Esto significa:

Independencia física de los datos. Aunque el nivel físico cambie, el nivel conceptual no debe verse afectado. En la práctica esto significa que, aunque se añadan o cambien discos u otro hardware, o se modifique el sistema operativo u otros cambios relacionados con la física de la base de datos, el nivel conceptual permanece invariable.

Independencia lógica de los datos. Significa que, aunque se modifique el esquema conceptual, los esquemas externos y los programas de aplicación no serán afectados.

En 1975 el comité ANSI/SPARC propuso una arquitectura cuyo objetivo es el de separar los programas de aplicación de la base de datos física, que en su división X3 establece: “la arquitectura de una base de datos debe poseer tres niveles: **Interno, Conceptual y Externo**”. Cada uno de ellos pertenece a un tipo de vista diferente: **almacenamiento físico, del usuario, y del programador**.

4.2. Nivel interno (estructura física).

Es el nivel más bajo de abstracción de la información. Es la representación inferior de una base de datos, por ello, es el más cercano al almacenamiento físico. Permite describir los datos tal como están almacenados en el ordenador. Por ejemplo:

- Los ficheros que contienen (nombre, abstracción, ubicación)
- Los registros de estos ficheros (longitud, campos, ...)

- Las rutas de acceso a esos registros (índices, encadenamientos, ...)

El nivel interno es descrito por el SGBD por medio del **esquema interno** o **vista interna**.

Hay que recordar que para un programador un registro interno almacenado difiere de un registro lógico. Un registro físico puede estar formado por uno o más registros lógicos junto a elementos descriptivos del sistema: longitud y tipo de registro, banderas y punteros, y está almacenado en un dispositivo electrónico codificado como una combinación de ceros y unos.

La operación de transformar registros lógicos en físicos se denomina *transformación de datos o mapeo*.

4.3. Nivel conceptual (estructura lógica global)

Es el siguiente nivel más alto de abstracción, el administrador de la base de datos del SGBD define el nivel conceptual por medio de un **esquema** o **vista conceptual**, al decidir qué información se guarda en la base de datos.

Este nivel corresponde a la estructura organizacional de los datos de la base, obtenida al reunir los requerimientos de todos los usuarios de una empresa, sin preocuparse de su organización física ni de las vías de acceso.

El esquema conceptual podría contener:

- Los datos elementales que definen los campos, **atributos** de los elementos de una empresa (NIF, nombre, concepto, ...).
- Los datos compuestos que permiten reagrupar los **campos** para describir los registros, las entidades del mundo real (persona, artículo, ...).
- Los datos compuestos que permiten reagrupar los campos para describir las **asociaciones** del mundo real (compras de artículos, propietarios de viviendas, ...).

4.4. Nivel externo (estructura lógica de usuario)

Es el nivel más alto de abstracción, y, por ello, el más cercano a los usuarios. El nivel externo representa la percepción individual de cada usuario o programador de la base de datos.

El SGBD es el encargado de extraer los datos requeridos por los registros lógicos externos de uno o más registros físicos de la base de datos, cada vez que se ejecuta una operación de entrada/salida en un programa específico.

Por cada programa es necesario especificar un **esquema externo**, **subesquema** o **vista externa** para poder acceder de forma selectiva a un subconjunto de datos de la base integrada. Habrá usuarios que puedan acceder a más de un esquema externo, y un esquema externo puede ser compartido por varios usuarios; se protege, de esta manera, el acceso no autorizado a datos y el de usuarios mal intencionados.

4.5. Interfaces

Entre estos niveles existen unos interfaces que realizan funciones de traducción:

Función de traducción externo/conceptual. Define la correspondencia entre cada una de las vistas externas y la única vista conceptual (diferentes tipos de datos, diferentes nombres de campos, múltiples registros conceptuales fundidos en un único registro externo).

Función de traducción conceptual/interno. Establece cómo se almacena a nivel interno los registros y campos conceptuales. Si se modifica el almacenamiento de los datos, sólo es necesario modificar la aplicación de correspondencia, y no la vista conceptual.

5. Diccionario de datos

Debe existir una verdadera arquitectura de datos que facilite la integración. El núcleo de dicha arquitectura será el **diccionario de recursos de información (iRD)**, que contendrá las descripciones de todos los datos que constituyen el sistema de información.

Las finalidades principales de los DRI son precisamente ayudar a la gestión de la información como recurso y conseguir la integración de la semántica de las aplicaciones de forma centralizada, con lo que se aumentan los beneficios estratégicos, operacionales y de control de las empresas.

Para Codd (1990), uno de los requisitos para que un SGBD pueda considerarse como verdaderamente relacional es que soporte un catálogo dinámico en línea en el que la descripción de la base de datos se represente como cualquier otro dato, permitiendo a los usuarios autorizados aplicar el mismo lenguaje relacional tanto a la descripción de la base de datos como a los datos regulares.

Los recursos informáticos de las instituciones se gestionan almacenando, administrando y controlando los denominados **metadatos**, esto es, los datos que definen y describen los datos de la institución. El término más difundido para estos metadatos es el de **Diccionario de Datos (DD)** o catálogo del sistema.

El directorio de datos ha sido desde siempre un componente del SGBD, encargado de describir dónde y cómo se almacenan los datos de la base, el modo de acceso y otras características físicas de los mismos, atendiendo de este modo las peticiones de los programas. Contiene, en definitiva, las especificaciones necesarias para pasar de la representación externa de los datos a la representación interna de los mismos. Su objetivo principal es transmitir al SGBD la información necesaria para poder acceder a los datos contenidos en la base.

Cuando los SGBD eran más limitados, los usuarios empleaban una lista "manual" donde anotaban la descripción narrativa y técnica (tamaño, tipo...) de los datos, así como los permisos de acceso a cada uno de ellos, el uso que las aplicaciones hacían de ellos, etc. Esta lista de denominaba diccionario de datos. Consistía en información sobre los datos, o, en otras palabras, datos sobre los datos, también conocidos como metadatos.

Por tanto, el diccionario de datos se considera como un conjunto de metadatos (datos sobre los datos) que contiene las características lógicas de los datos que se van a utilizar en el sistema que se programa, incluyendo nombre, descripción, alias, contenido y organización.

Si el diccionario de datos lo usan sólo los administradores, usuarios y diseñadores, y no el software del SGBD.

Por último, a partir de los 80, el diccionario de datos se incorporó a los SGBD como un componente más e integrándose junto al directorio. Al conjunto se le suene denominar catálogo o repositorio. El catálogo es el responsable de establecer la correspondencia entre los tres niveles de abstracción. Los módulos del SGBD usan y leen el catálogo con mucha frecuencia; por ello es importante implementar su acceso de la forma más eficiente posible.

Con la amplia difusión del modelo relacional el catálogo se almacena junto al resto de los datos de la base, en forma de tablas. La información almacenada en el catálogo de un SGBD relacional incluye, entre otros aspectos:

- Descripciones de los nombres de las relaciones (tablas).
- Nombres de los atributos.
- Tipos de datos de los atributos (dominios).
- Claves primarias.
- Claves ajenas.
- Obligación de establecer los datos como no nulos en algunas columnas.
- Vistas.
- Índices.
- Estructuras de almacenamiento.
- Permisos.
- Información sobre el propietario de cada tabla.

Dado que los datos del catálogo se almacenan en forma de tablas, se puede acceder a los mismos mediante instrucciones de SQL.

Directorio de datos: Es un subsistema del SGBD, encargado de describir dónde y cómo se almacenan los datos de la base, el modo de acceso y otras características físicas de los datos, atendiendo de este modo las peticiones de los programas y procesos. Un directorio de datos contiene, en definitiva, las especificaciones necesarias para pasar de la representación externa de los datos a su representación interna. Ha de estar siempre en un formato legible para la máquina. Su objetivo principal es transmitir al sistema la información necesaria para poder acceder a los datos contenidos en la base. El directorio, a diferencia del diccionario, es un instrumento del SGBD, y está orientado a facilitar a éste la información que necesita para su funcionamiento.

Un desarrollo importante en los SGBD relacionales es la práctica común de almacenar el directorio como un conjunto de relaciones. Esto permite el uso de los lenguajes de manipulación de datos de los SGBD para consultar, actualizar y mantener el diccionario de datos.

Recientemente, ha aparecido un nuevo concepto, el **diccionario de recursos de información (dri)**, que pretende ser el eslabón final en la evolución de los almacenes de datos. El DRI constituye el depósito integrado de todos los datos sobre la organización, automatizados o no, que son utilizados para efectuar las labores de planificación, control y operación que permitan a la empresa cumplir sus objetivos. Éstos engloban de algún modo las capacidades y funciones de todos los almacenes de datos anteriores.

6. Conclusión

La función de descripción o definición de datos debe permitir al administrador de la base de datos especificar los elementos de datos que integran la BD, su estructura y las relaciones que existen entre ellos.

Dicha función se realiza mediante un lenguaje de definición de datos (DDL) propio de cada SGBD.

También hemos visto como la arquitectura ANSI/SPARC hace una división en tres niveles según la perspectiva con que se mire la información: nivel externo, conceptual e interno. El DDL por tanto, es el encargado de suministrar los medios para definir las tres estructuras de datos asociadas a dichos niveles (esquema externo, conceptual e interno).

Por último, la estructura de datos que facilita la integración es el Diccionario de Datos.

6.1. Relación del tema con el sistema educativo actual

Este tema es aplicado en el aula en los módulos profesionales siguientes, con las atribuciones docentes indicadas (PES/SAI):

Formación profesional básica

- Operaciones auxiliares para la configuración y la explotación(TPB en Informática de Oficina/ TPB en informática y Comunicaciones) (PES/SAI)
- Ofimática y archivo de documentos (TPB en Informática de Oficina) (PES/SAI)

Grado Medio

- Aplicaciones ofimáticas (GM de SMR) (PES/SAI)

Grado Superior

- Gestión de bases de datos (ASIR) (PES)
- Bases de Datos (DAW/DAM) (PES)

Bachillerato:

- 4º ESO – Tecnología de la Información y la comunicación (PES)
- Bachillerato – Tecnologías de la Información y la Comunicación (PES)

7. Bibliografía

- C.J. Date: **Introducción a los sistemas de bases de datos** Pearson, 2001.
- Elmasri, R.A. y Navathe S.B: "**Fundamentos de Sistemas de Bases de Datos**". Addison-Wesley, 3^a Edic, 2002.
- Olga Pons, J M Medina, M.A. Vila. **Introducción a los Sistemas de Bases de Datos** Edt Paraninfo (2005)
- Korth, H.F. y Silberschatz: "**Fundamentos de Bases de Datos**". McGraw -Hill, 4^a Edic., 2002.
- Garcia-Molina, H.; Ullman, J.D.; Widom, J. **Database systems: the complete book** - Pearson Education Limited, 2013.
- Abraham Silberschatz, Henry F. Korth, y S. Sudarshan, **Fundamentos de bases de datos** Edt. Mc Graw-Hill (2014)
- <https://elbauldelprogramador.com/> (2020)
- www.Unir.net (2020)

