Unidad 4

Seguridad

Administración de sistemas gestores de bases de datos

Índice

Introducción

En todos los SGBD, debemos garantizar la confidencialidad de la información, así como la integración de ésta, evitando corrupción, pérdida o inconsistencias en los datos.

En los sistemas multiusuario, nos enfrentamos ante escenarios más complicados, en los que hemos de asegurar que cada usuario accede solamente a aquella información a la que se le ha autorizado, así como garantizar errores debidos a acceso simultáneos que provoquen inconsistencias en los datos.

Vamos a introducir algunos de estos conceptos relacionados con la seguridad de la información con los que trabajaremos a lo largo de la unidad.

- LOPD: Ley Orgánica de protección de datos.
- Datos de carácter personal: toda aquella información que permita identificar a una persona.
- Encriptación transparente (TDE): Mecanismo por el cual, el SGBD encripta la información al guardarla y la desencripta al recuperarla, sin que el usuario tenga que intervenir explícitamente.
- Restricciones: Conjunto de reglas que definen si ciertos valores son válidos o no para guardarlos en una base de datos. Su objetivo es garantizar la integridad de la información.
- Aislamiento: capacidad de mantener los datos de una transacción separados y aislados de los datos de otras transacciones. El aislamiento garantiza que las transacciones no se interrumpan ni afecten entre sí, y también impide que los datos sean leídos o modificados por otras transacciones hasta que se confirmen o se descarten.
- Niveles de aislamiento: Los niveles de aislamiento en una base de datos transaccional son los diferentes grados en los que se pueden aislar las transacciones unas de otras. El nivel de aislamiento más bajo es el nivel de aislamiento de lectura/escritura, donde las transacciones no pueden leer ni modificar los datos de otras transacciones. El nivel de aislamiento más alto es el nivel de aislamiento de serialización, donde las transacciones se ejecutan de forma completamente aislada una de otra.
- Concurrencia: permite la ejecución de múltiples transacciones a la vez.
- Interbloqueo: El interbloqueo es un problema que puede ocurrir en una base de datos transaccional cuando dos o más transacciones están bloqueando el mismo conjunto de datos. Esto puede ocurrir si las transacciones no están bien aisladas unas de otras y si no se está utilizando el nivel de aislamiento adecuado. El interbloqueo puede hacer que una base de datos se bloquee y deje de responder, lo que puede causar una pérdida de datos.
- Propiedades ACID: La base de datos debe cumplir con las propiedades ACID para garantizar que las transacciones se ejecuten de forma correcta y segura. Las propiedades ACID son:
  - Atomicidad. Las transacciones deben ser tratadas como unidades atómicas, o sea, deben ejecutarse completamente o no ejecutarse en absoluto.
  - Consistencia. Las transacciones deben dejar la base de datos en un estado consistente.
  - Aislamiento. Las transacciones deben estar aisladas unas de otras para evitar el interbloqueo.
  - Durabilidad. Una vez que se confirma una transacción, los cambios deben persistir en la base de datos, incluso si se produce un fallo del sistema.
- Tipos de políticas de seguridad en SGBD:
  - Activas. Las tareas son implementadas de forma proactiva, para prevenir fallos en el SGBD. Respecto a la confidencialidad, definen la gestión de usuarios y permisos, definición de esquemas externos y encriptación de la información. En referencia a la integridad, estas políticas gestionan la concurrencia, transacciones y restricciones.
  - Pasivas. Se interviene cuando ocurren fallos de las activas. Se trata de encontrar la causa de los fallos e incorporar un mecanismo de preventivo a las activas. Respecto a la confidencialidad, se incluyen las auditorías. En referencia a la integridad, se recurre a los mecanismos de respaldo y recuperación.

Al finalizar esta unidad

- Aprenderemos a establecer los mecanismos de seguridad del SGBD que nos van a permitir garantizar la integridad y confidencialidad de los datos.
- Conoceremos las diferentes herramientas de los SGBD que permiten garantizar la seguridad del sistema.
- Seremos conscientes de lo importante que es definir y mantener unas políticas de copias de seguridad y restauración.
- Comprenderemos la utilidad de las herramientas incluidas en el SGBD y que contribuyen a garantizar las medidas impuestas por la normativa de protección de datos.

4.1. Confidencialidad

Para garantizar la confidencialidad de la información, además de una adecuada gestión de usuarios y permisos junto con la definición de esquemas externos, debemos aplicar dos medidas adicionales que son la encriptación y la auditoría.

4.1.1. Encriptación

Como no podemos garantizar que usuarios no autorizados puedan acceder a determinados archivos, por ejemplo, un administrador de sistemas que no tenga permisos para acceder a ciertos datos sensibles de la base de datos podría acceder a los ficheros, pues tiene permisos de administrador sobre el equipo servidor.

En ocasiones es posible que, aunque un usuario tenga permisos para acceder a unas determinadas tablas, se debe proteger la información de ciertas columnas, recurriendo al cifrado, por medio de algoritmos como RSA o AES.

La información se puede cifrar de dos modos:

Mediante funciones

El SGBD proporciona las funciones para encriptar y desencriptar la información. En este tipo de cifrado es explícito, es decir, el programador tiene que encriptar o desencriptar la información.

Las funciones del SGBD son de dos tipos, unas que encriptan la información y no permiten desencriptarla y otras que usan la misma clave de cifrado (simétrico) que permite desencriptar la información. Dependiendo del uso, nos podremos encontrar con los siguientes tipos de funciones:

- **Encriptar/desencriptar:** Se utiliza para proteger cierta información que los usuarios no deben ver, aunque tengan acceso a la tabla, por ejemplo, salarios de empleados, datos personales protegidos por LOPD, etc. Los datos pueden ser desencriptados si se conoce la clave de cifrado, que es la misma.
- **Hash:** Se utiliza para guardar contraseñas. En ningún momento se desencripta, sino que se calcula el hash de la entrada con el hash guardado para validar la contraseña.
- **MAC:** Similar a la función hash, pero añade seguridad, pues se agrega una clave de encriptación necesaria para el cifrado. Protege ante ataques por fuerza bruta o diccionario. Sin conocer la clave, el atacante no podría conocer el hash.

Cifrado transparente

Permite almacenar la información encriptada en los discos sin necesidad de que el usuario tenga que encriptarla al almacenar ni desencriptarla al recuperarla. Los datos viajan cifrados por la red, estableciendo comunicaciones por medio de protocolos seguros como SSL.

El cifrado transparente, protege la información guardada físicamente, para que, en caso de robo, no pueda ser extraída. También protege a nivel de red contra ataques que capturen tráfico de la red. Sin embargo, las operaciones de consulta e inserción, dentro del SGBD y a nivel local, se realizan con los datos descifrados, por lo tanto, no es posible cifrar la información para unos usuarios del SGBD y no para otros, para ello se requiere cifrado mediante funciones.

PARA TENER EN CUENTA...

El cifrado transparente, se suele realizar mediante un algoritmo simétrico como AES. El SGBD, utiliza una clave para encriptar y desencriptar la información. Esto ha de tenerse en cuenta cuando se vaya a migrar la base de datos a otro servidor, pues debe copiarse la clave en el nuevo servidor.

4.1.2. Auditoría

Una auditoría es una medida de seguridad pasiva, pues se suele recurrir a ella cuando se han detectado ciertas anomalías, como un acceso u operación inadecuado y es necesario investigar las causas. Nos va a permitir detectar:

- **Accesos no autorizados al sistema:** por ejemplo, accesos o intentos en horas anómalas.
- **Desbordamiento de memoria:** el atacante pasa como entrada a un programa más datos de los que debería recibir. Esto permitiría al atacante escribir en una zona de memoria más allá de la reservada.
- **Inyección de Código SQL:** ataque consistente en introducir código SQL a través de las aplicaciones que acceden a la base de datos.

Una auditoría se puede realizar en dos niveles:

- **Sesión:** usuarios conectados a la base de datos, tablas accedidas o ubicación del acceso durante una sesión.
- **Objeto:** este nivel de auditoría es más preciso, pues permite conocer todos los detalles sobre los accesos y operaciones que se han realizado sobre la base de datos. De este modo, se permite controlar el acceso a los datos sensibles protegidos por LOPD.

PARA TENER EN CUENTA...

La información de la auditoría se puede guardar en ficheros o tablas (dentro del diccionario de datos). Cuando la información se almacena en el diccionario, facilita su consulta posterior y la realización de informes de auditoría. Esto tiene el inconveniente de consumir más recursos del SGBD, lo que podría comprometer la capacidad del sistema.

Debemos guardar la información resultante de la auditoría en un disco diferente del sistema operativo del servidor donde tengamos el SGBD. Es recomendable hacerlo en discos externos o sistemas de almacenamiento en red para que, en caso de caída, pueda consultarse la información de la auditoría y analizar las causas con el fin de implementar medidas correctivas.

4.2. Confidencialidad en Oracle

Vamos a concretar en la práctica, como Oracle implementa las medidas descritas anteriormente para garantizar la confidencialidad de los datos.

4.2.1. Encriptación en Oracle (Funciones)

Respecto a la encriptación de funciones, Oracle incorpora el paquete DBMS_CRYPTO, donde almacena todas las funciones necesarias para encriptar y desencriptar la información. Los paquetes del sistema se almacenan en el diccionario de datos y podremos acceder a ellos con el usuario SYS.

Estas funciones hacen uso de algoritmos de criptografía como DES, AES, SHA, en sus diferentes versiones, aunque algunas ya están obsoletas como DES.

Las funciones más comunes son:

- **Encrypt:** se utiliza para encriptar un texto CLOB (Cadena de caracteres larga), que podremos desencriptar posteriormente con decrypt.

```
FUNCTION  Encrypt (src IN            RAW,
                   typ IN            PLS_INTEGER,
                   key IN            RAW,
                   iv  IN            RAW          DEFAULT NULL)
  RETURN RAW;

PROCEDURE Encrypt (dst IN OUT NOCOPY BLOB,
                   src IN            BLOB,
                   typ IN            PLS_INTEGER,
                   key IN            RAW,
                   iv  IN            RAW          DEFAULT NULL);

PROCEDURE Encrypt (dst IN OUT NOCOPY BLOB,
                   src IN            CLOB         CHARACTER SET ANY_CS,
                   typ IN            PLS_INTEGER,
                   key IN            RAW,
                   iv  IN            RAW          DEFAULT NULL);
```

Imagen 1. Función Encrypt (Oracle)

- **Decrypt:** Función que recibiría el bloque cifrado como parámetro y lo devolvería descifrado. Tanto encrypt como decrypt, deben usar los mismos algoritmos y parámetros.

```
FUNCTION  Decrypt (src IN            RAW,
                   typ IN            PLS_INTEGER,
                   key IN            RAW,
                   iv  IN            RAW          DEFAULT NULL)
  RETURN RAW;

PROCEDURE Decrypt (dst IN OUT NOCOPY BLOB,
                   src IN            BLOB,
                   typ IN            PLS_INTEGER,
                   key IN            RAW,
                   iv  IN            RAW          DEFAULT NULL);

PROCEDURE Decrypt (dst IN OUT NOCOPY CLOB         CHARACTER SET ANY_CS,
                   src IN            BLOB,
                   typ IN            PLS_INTEGER,
                   key IN            RAW,
                   iv  IN            RAW          DEFAULT NULL);
```

Imagen 2. Función Decrypt (Oracle)

```
select DBMS_CRYPTO.encrypt(
UTL_RAW.CAST_TO_RAW('TextoPlano'),
4353, /* ENCRYPT_DES + CHAIN_CBC + PAD_PKCS5 */
UTL_RAW.CAST_TO_RAW ('FFFFFFFFFKKKKKKK')) /* Clave */
AS TextoCifrado
from dual; /* Devuelve FF35F180C1F431F3DA77AEAF5855A2E9 */

 select UTL_RAW.CAST_TO_varchar2(
DBMS_CRYPTO.decrypt(
'FF35F180C1F431F3DA77AEAF5855A2E9', /* Texto cifrado */
4353, /* ENCRYPT_DES + CHAIN_CBC + PAD_PKCS5 */
UTL_RAW.CAST_TO_RAW ('FFFFFFFFFKKKKKKK')))
as TextoPlano
from dual;
```

Imagen 3. Ejemplo básico de uso de encrypt/decrypt en Oracle

> **Hash:** Con esta función se calcula el hash, que no será desencriptado.

```
FUNCTION Hash (src IN RAW,
               typ IN PLS_INTEGER)
  RETURN RAW DETERMINISTIC;

FUNCTION Hash (src IN BLOB,
               typ IN PLS_INTEGER)
  RETURN RAW DETERMINISTIC;

FUNCTION Hash (src IN CLOB         CHARACTER SET ANY_CS,
               typ IN PLS_INTEGER)
  RETURN RAW DETERMINISTIC;
```

Imagen 4. Función Hash (Oracle)

- **MAC:** La misma utilidad que la función Hash pero con clave de encriptación.

```
FUNCTION Mac (src IN RAW,
              typ IN PLS_INTEGER,
              key IN RAW)
  RETURN RAW;

FUNCTION Mac (src IN BLOB,
              typ IN PLS_INTEGER,
              key IN RAW)
  RETURN RAW;

FUNCTION Mac (src IN CLOB         CHARACTER SET ANY_CS,
              typ IN PLS_INTEGER,
              key IN RAW)
  RETURN RAW;
```

Imagen 5. Función Hash (Oracle)

La definición de todas las funciones del paquete puede localizarse en el diccionario de datos. Se recomienda consultar la documentación de Oracle y la definición para comprender su funcionamiento.

Hemos visto un ejemplo muy sencillo de cómo encriptar y desencriptar una cadena de texto. Requiere conversión de tipos a los tipos especificados en la definición de la función y, conocer el código "typ" que define el algoritmo y los parámetros de cifrado, que se calcula sumando el código del algoritmo de cifrado (ENCRYPT_DES = 1), el modo de cifrado de bloques (CHAIN_CBC = 256) y el modificador de relleno (PAD_PKCS5 = 4096) → 1+256+4096 = 4353. Navegando por el paquete, podremos encontrar las funciones y su documentación detallada donde describe los códigos.

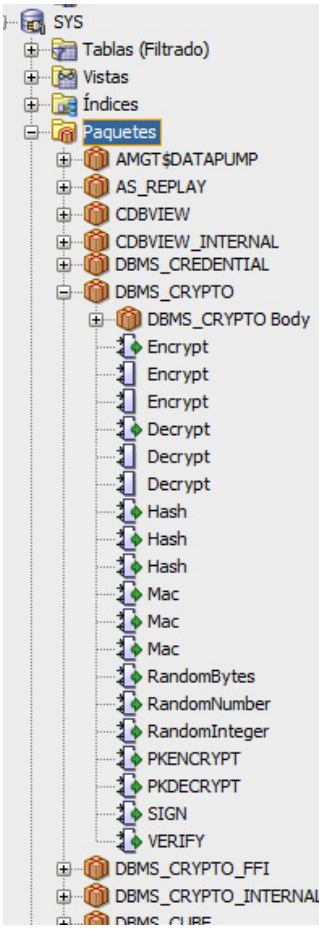

Imagen 6. Definición de funciones del paquete DBMS_CRYPTO (diccionario de datos)

4.2.2. Encriptación en Oracle (TDE)

Oracle permite cifrado transparente, es decir, que la información de determinadas columnas o tablespace se puede guardar siempre encriptada, sin que el usuario o administrador tenga que intervenir, pues el SGBD se encarga de ello, cifrando al guardar y descifrando al recuperar. Esto tiene un inconveniente y es que, aunque mejor a la seguridad, incrementa los tiempos de acceso. Deberíamos cifrar solamente la información sensible para no malgastar recursos innecesariamente.

Imagen 7. Encriptación transparente

Para implementar el cifrado transparente, Oracle utiliza una clave maestra de cifrado (Master Encryption Key).

La clave maestra está formada por un par de claves pública y privada (o certificado digital), que se guardan en el Wallet (versiones 11g y anteriores) o Keystore (versiones posteriores), que es un fichero para almacenar claves.

Podemos abrir o cerrar el keystore para permitir o denegar el acceso a los usuarios. Cuando está cerrado, no se puede descifrar la información.

Vamos a realizar los pasos necesarios para habilitar la encriptación transparente.

Creación del Keystore

Se necesita ser DBA. Los almacenes de claves se generan desde la instancia. Los hay dos tipos:

- **Basados en contraseña:** requiere utilizar la contraseña para operar con ellos. Son más fáciles de crear, pero requieren abrirlos manualmente con cada reinicio de la instancia. Vamos a realizar las operaciones necesarias para crear un keystore basado en contraseña:

```
ADMINISTER KEY MANAGEMENT
CREATE KEYSTORE 'C:\app\Administrador\product\21c\admin\XE\wallet\'
IDENTIFIED BY "m1pwk34st0r3";
```

Imagen 8. Creación de un keystore basado en contraseña

- **De autenticación automática:** A partir de uno basado en contraseña se crean y lo abren automáticamente cada vez que este se cierra.

Incluirlo en el sqlnet.ora

Debemos agregar la siguiente línea a nuestro fichero de configuración:

```
ENCRYPTION_WALLET_LOCATION=
    (SOURCE=
      (METHOD=FILE)
     (METHOD_DATA=(DIRECTORY= C:\app\Administrador\product\21c\...\wallet)))
C:\app\Administrador\product\21c\admin\XE\wallet
```

Abrir o cerrar el keystore

Para hacerlo en todas las bases de datos, debemos incluir la opción CONTAINER = ALL.

```
ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN
IDENTIFIED BY "mlpwk34st0r3" CONTAINER=ALL;
```

Imagen 9. Abrir el keystore

El keystore queda abierto para todas las PDBs, pero no tiene una clave maestra para encriptar y desencriptar la información. Si ejecutamos la siguiente consulta:

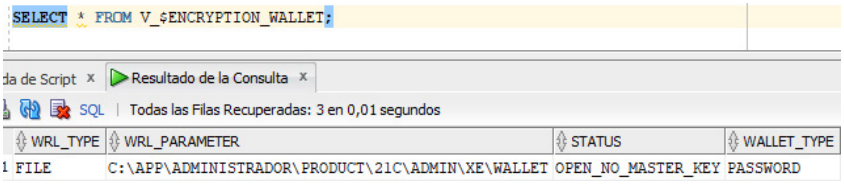

Imagen 10. Estado del keystore antes de crear la clave maestra

Podemos comprobar que en la columna STATUS, no hay una clave maestra definida.

Crear la clave maestra

Se debe crear sobre la instancia, también con usuario con rol DBA.

```
ADMINISTER KEY MANAGEMENT SET KEY IDENTIFIED BY "mlpwk34st0r3" WITH BACKUP CONTAINER=ALL;
```

Imagen 11. Creando la clave maestra

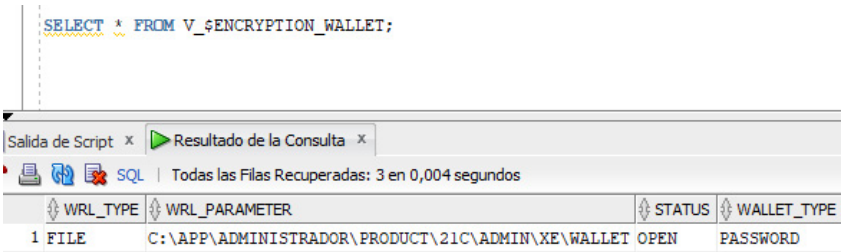

Imagen 12. Estado del keystore tras crear la clave maestra

Crear el keystore de autenticación automática

Ya tenemos el almacén creado basado en contraseña, el problema es que cada vez que se reinicie la instancia, tendremos que abrirlos explícitamente. Para evitarlo, vamos a crearlo de forma que no sea necesario abrirlo con cada reinicio:

```
ADMINISTER KEY MANAGEMENT
CREATE AUTO_LOGIN KEYSTORE
FROM KEYSTORE 'C:\app\Administrador\product\21c\admin\XE\wallet'
IDENTIFIED BY "mlpwk34st0r3";

SELECT * FROM V_$ENCRYPTION_WALLET;
```

```
la de Script × | Resultado de la Consulta ×
SQL | Todas las Filas Recuperadas: 3 en 0,093 segundos
```

WRL_TYPE	WRL_PARAMETER	STATUS	WALLET_TYPE
FILE	C:\APP\ADMINISTRADOR\PRODUCT\21C\ADMIN\XE\WALLET	OPEN	AUTOLOGIN

Imagen 13. Convertir el keystore de tipo contraseña en uno de autenticación automática.

Si todo ha funcionado correctamente tendremos al menos, los dos ficheros en nuestra carpeta wallet:

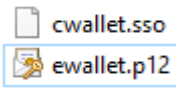

Imagen 14. Fichero sso (keystore autologin), .p12 (keystore password).

4.2.3. Auditoría en Oracle

Para verificar si la auditoría está activa, tenemos que poner el foco en el valor del parámetro "audit_trail". Para ello, consultaremos la vista que muestra la información referente a ese parámetro:

```
SELECT * FROM v$parameter WHERE name = 'audit_trail'
```

```
ía de Script × | Resultado de la Consulta ×
SQL | Todas las Filas Recuperadas: 1 en 0,011 segundos
```

NUM	NAME	TYPE	VALUE	DISPLAY_VALUE	DEFAULT
3990	audit_trail	2	DB	DB	none

Imagen 15. Valor del parámetro "audit_trail".

Si nos fijamos en la columna "value", vemos que actualmente muestra "db". Los posibles valores que puede tomar son:

- `db:` este valor activa la auditoría y los datos se irán guardando en el diccionario de datos. En concreto en la tabla SYS.AUD$.
- `none:` auditoría desactivada.
- `os:` Activa la auditoría, pero los registros se almacenan en el sistema operativo.
- `Xml:` La auditoría está activa y los eventos se guardan en ficheros xml en un directorio especificado como parámetro ("audit_file_dest").

Si queremos desactivar la auditoría, bastaría con poner el siguiente comando:

```
ALTER SYSTEM SET audit_trail = 'none' SCOPE=SPFILE;
```

Imagen 16. Comando para desactivar la auditoría

Oracle ofrece herramientas para la gestión de la auditoría en tres niveles diferentes: mediante comandos audit/noaudit, de grano fino (paquete dbms_fga) y auditorías a nivel de objeto.

Comandos audit y noaudit

Para poder ejecutar estos comandos, la auditoría debe estar activa. Es necesario consultar valores de audit_trail y establecer el valor distinto de 'none'. Ese no es el único requisito, pues si queremos auditar un objeto, debemos ser el propietario del objeto o tener privilegios de sistema (AUDIT ANY).

Mediante el comando audit, podremos auditar tres tipos de acciones:

- **Inicio de sesión:** se auditan los intentos de conexión con las bases de datos. Podemos filtrar por usuario o incluso auditar solamente intentos de conexión o inicios de sesión fallidos. El comando es el siguiente:

  ```
  AUDIT SESSION [BY <usuario>] [WHENEVER
  SUCCESSFUL | NOT SUCCESSFUL];
  ```

- **Acción:** se audita cualquier acción sobre un tipo de objeto de la base de datos. Las acciones a auditar son (CREATE, ALTER, DROP), las cuales se pueden agrupar para facilitar el mantenimiento. También es posible limitar las operaciones a registrar sobre el usuario. El comando es el siguiente:

  ```
  AUDIT [<CREATE | ALTER | DROP> <TABLE | SYNONYM
  | TABLESPACE, …>  [BY <usuario>];
  ```

- **Objeto:** se auditan las operaciones de manipulación de datos sobre algunos objetos (vistas, tablas o ejecución de funciones o procedimientos). En este nivel de auditoría, no se puede restringir el registro de datos a nivel de usuario.

  ```
  AUDIT {Insert | update | delete | select | all}
  ON <objeto> [WHENEVER [NOT] SUCCESSFUL];
  ```

Con la misma descripción anterior, podemos ejecutar el comando NOAUDIT, en lugar de AUDIT, para detener la auditoría solamente en el nivel que especifiquemos en el comando.

Auditoría de grano fino: paquete DBMS_FGA

Este tipo de auditoría está basada en reglas. FGA nos permite especificar bajo qué condiciones se tiene que generar un registro de auditoría.

Volvemos a hacer uso de los paquetes de Oracle, pero esta vez las funciones y procedimientos de auditoría de grano fino están definidas en el paquete DBMS_FGA, al que podremos acceder navegando por el diccionario de datos con el usuario SYS.

Cuando hacemos uso de funciones o procedimientos de un paquete cuyo uso todavía no nos es familiar, es conveniente buscar en el paquete cómo funciona. Veamos las funciones del paquete:

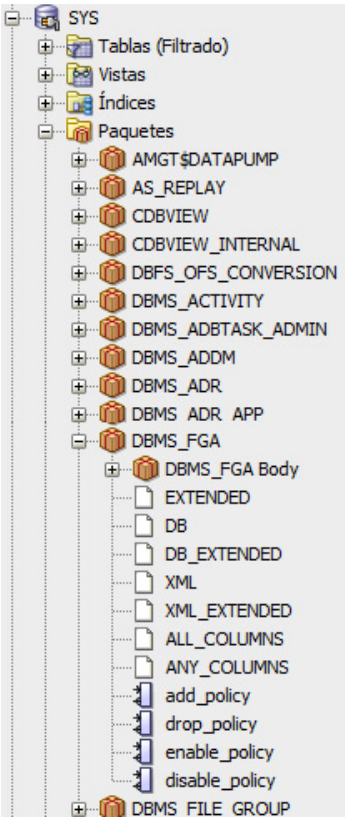

Imagen 17. Funciones del paquete DBMS_FGA

Vemos que el paquete tiene cuatro funciones para añadir, eliminar y activar o desactivar reglas. Para poder hacer uso de estas funciones, debemos tener permisos de ejecución sobre el paquete.

Cuando creemos una regla con add_policy, debemos especificar a qué objeto la vamos a asociar y bajo qué condiciones se generará el evento de auditoría que se va a registrar. Por ejemplo, podemos auditar si hay accesos a una tabla desde una geolocalización fuera de del país. Podemos especificar concretamente qué tipo o tipos de acceso ha habido sobre la tabla (operaciones select, insert, update o delete).

Veamos los parámetros del procedimiento dbms_fga.add_policy:

```
PROCEDURE add_policy(object_schema     IN VARCHAR2 := NULL,
                     object_name       IN VARCHAR2,
                     policy_name       IN VARCHAR2,
                     audit_condition   IN VARCHAR2 := NULL,
                     audit_column      IN VARCHAR2 := NULL,
                     handler_schema    IN VARCHAR2 := NULL,
                     handler_module    IN VARCHAR2 := NULL,
                     enable            IN BOOLEAN  := TRUE,
                     statement_types   IN VARCHAR2 := 'SELECT',
                     audit_trail       IN PLS_INTEGER  := 3,
                     audit_column_opts IN BINARY_INTEGER DEFAULT 0,
                     policy_owner      IN VARCHAR2 := NULL);
```

Veamos un ejemplo concreto:

```
execute DBMS_FGA.ADD_POLICY (
   object_schema      =>  'HR',
   object_name        =>  'EMPLOYEES',
   policy_name        =>  'mypolicy1',
   audit_condition    =>  'SALARY < 100',
   audit_column       =>  'SALARY',
   handler_schema     =>   NULL,
   handler_module     =>   NULL,
   enable             =>   TRUE,
   statement_types    =>  'INSERT, UPDATE',
   audit_column_opts  =>   DBMS_FGA.ANY_COLUMNS);
   END;
```

Imagen 18. Agregando una regla de auditoría

```
execute DBMS_FGA.DROP_POLICY(
   object_schema   => 'HR',
   object_name     => 'EMPLOYEES',
   policy_name     => 'mypolicy1);
```

Imagen 19. Eliminando una regla de auditoría

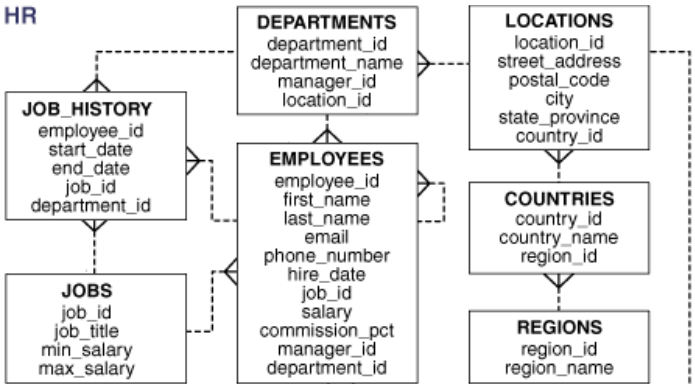

Imagen 20. Esquema de ejemplo incluido en Oracle

PARA TENER EN CUENTA...

Durante el proceso de instalación de Oracle XE, tenemos la opción de configurar un esquema con datos de ejemplo, con una tabla que simula un departamento de recursos humanos.

Auditoría unificada

Nos facilita la auditoría para un grupo de usuarios. Mientras que con audit se realiza manualmente, almacenando registros en varias tablas diferentes, la auditoría unificada agiliza esta tarea, permitiendo la creación de reglas similares a las de la auditoría fina pero asignándolas a un grupo de usuarios.

Para crear las reglas de auditoría unificadas, utilizaremos el comando

```
CREATE AUDIT POLICY.
```

Cuando se haya creado, se podrá deshabilitar o habilitar con

```
AUDIT POLICY <regla> [BY <usuario>] [WHENEVER [NOT]
SUCCESSFUL];
NOAUDIT POLICY <regla> [BY <usuario>] [WHENEVER [NOT]
SUCCESSFUL];
```

4.2.4. Auditoría en el diccionario de datos de Oracle

Cuando hemos activado el registro de auditoría a nivel de base de datos (DB), se podrá consultar la información en diferentes tablas/vistas:

- **Vista DBA_AUDIT_SESSION:** Contiene la información de los inicios de sesión.
- **Vista DBA_AUDIT_OBJECT:** Muestra la información de los objetos auditados.
- **Tabla FGA_LOGS$:** Guarda os registros de auditoría fina, aunque puede consultarse con la vista DBA_FGA_AUDIT_TRAIL.
- **Vista DBA_STMT_AUDIT:** Acciones e inicios de sesión que el sistema está auditando.
- **Vista DBA_OBJ_AUDIT_OPTS:** registro de operaciones que se realizan sobre los objetos auditados.
- **DBA_AUDIT_POLICIES:** reglas de auditoría que han sido definidas por medio de FGA.
- **AUDIT_UNIFIED_POLICIES:** Reglas de auditoría unificadas. Con AUDIT_UNIFIED_ENABLED_POLICIES, se consultan únicamente las habilitadas.
- **DBA_COMMON_AUDIT_TRAIL:** Muestra todas las reglas de auditoría que se han definido independientemente del método utilizado.

4.3. Integridad

Para poder garantizar la integridad de la información, un SGBD suele implementar cuatro funcionalidades que contribuyen a proporcionarla: restricciones, control de concurrencia, transacciones y copias de seguridad.

4.3.1. Restricciones

En los SGBD relacionales, las restricciones se van definen con el modelo de datos. Aunque fuera del modelo, pueden definirse otras restricciones sobre uno o varios campos de una tabla o conjunto de tablas. Las siguientes restricciones son las más comunes:

- **Nulidad:** se comprueba que no se insertan nulos.
- **Unicidad:** protege contra la inserción de datos duplicados.
- **Clave primaria:** asegura que el dato insertado no es nulo ni duplicado según la definición de la clave que puede ocupar uno o varios campos.
- **Clave foránea:** Limita el conjunto de valores insertados o modificados a los valores que toma la clave primaria de otra tabla.
- **Valor por defecto:** en caso de no especificar el valor a insertar, se asegura que hay un valor por defecto que será insertado.
- **Comprobación:** Se define una condición que se debe cumplir, de lo contrario no se permite la inserción/modificación de datos.
- **Disparadores:** Se programan acciones que se ejecutarán antes o después de realizar modificaciones en los datos de una tabla.

Por razones de mantenimiento o control, es posible que se den ocasiones en las que sea necesario desactivar esas restricciones. Hay algunos SGBD que obligan a deshabilitar todo tipo de restricciones y otros permiten hacerlo de forma parcial, por ejemplo, deshabilitar disparadores únicamente.

La sintaxis para deshabilitar las claves foráneas sería:

```
ALTER TABLE <tabla_nombre> DISABLE CONSTRAINT
<fk_nombre>
```

Para volverla a habilitar sería igual, pero cambiando DISABLE por ENABLE.

PARA TENER EN CUENTA...

Cuando se vuelven a habilitar las restricciones, se realiza una comprobación de que se cumplen todas, sino lanzaría un error de restricción la instrucción ENABLE. Dependiendo de la cantidad de restricciones que se hayan podido incumplir durante las operaciones de mantenimiento, esta tarea puede ser tediosa, especialmente en aquellos SGBD que solamente pueden deshabilitar las restricciones a nivel global.

4.3.2. Control de concurrencia

Las transacciones son unidades "lógicas" de movimiento de datos, es decir, son un conjunto de operaciones que se ejecutan de forma atómica. Al comienzo de la transacción, se guarda un estado de la base de datos y en caso de que alguna de las operaciones del conjunto no se haya podido realizar, se vuelve al estado guardado.

Las instrucciones de transacción, que vienen definidas en el lenguaje TCL, son solamente dos: commit y rollback. La primera confirma que la transacción se completó y la segunda es la que se ejecuta cuando hay algún tipo de error. Es común utilizar rollback en las excepciones del código fuente de un programa.

Durante el proceso de una transacción, es posible que se necesite acceder de forma concurrente a los datos, lo cual puede llevar a tres tipos de problemas de concurrencia:

Lectura sucia (dirty read)

Ocurre cuando hay una transacción sin finalizar y otra transacción lee los datos modificados por la primera.

Por ejemplo:

Un cliente tiene 2 cuentas en un banco C1 y C2. C1 tiene 1500 € y C2 500 €.

La transacción T1 lee el saldo de C1, realiza una operación UPDATE para modificarlo a 2000 € y otra operación UPDATE de C2 a 0, porque ha traspasado 500 € de C2 a C1,. Antes de que T1 haga el segundo UPDATE y el commit, la transacción T2 calcula la suma sus cuentas, dando como resultado 2500.

Lectura no repetible (nonrepeteable read)

Ocurre cuando durante el proceso de una transacción, se realiza una misma consulta sobre un dato dos veces y se obtienen valores distintos debido a que otra transacción lo ha modificado.

Lectura fantasma (phantom read)

Si una transacción lee unos datos que no existían al inicio de la transacción.

Para dar solución a estos problemas de concurrencia, se suele emplear el mecanismo de bloqueo. Cuando una transacción está realizando una lectura o escritura sobre una tabla, se impide el acceso a la misma al resto de transacciones hasta que no finalice con un commit o rollback.

El bloqueo se puede realizar en varios niveles, tabla completa, datafile, partición, determinadas filas, etc. Aunque no todos los SGBD soportan todos los niveles. Por ejemplo, MySQL los realiza a nivel de tabla, lo cual impide que se puedan realizar operaciones de modificación de datos en toda la tabla durante la transacción.

Problemas de los bloqueos (deadlocks)

El uso de mecanismos de bloqueos puede dar lugar a casos de interbloqueos (deadlock). Estos se producen cuando una transacción bloquea un elemento que otra transacción lo necesita, quedando ambas atrapadas.

Aislamiento y gestión de interbloqueos

Para evitar este problema, se debe prevenir y detectar. El SGBD puede obligar a que las transacciones reserven los recursos que van a utilizar por adelantado para que, una transacción que vaya a usar los mismos recursos que otra ya comenzada, no pueda hacerlo hasta que no se liberen. El SGBD también dispone de mecanismos de control que detectan periódicamente esos interbloqueos.

El SGBD permite definir diferentes niveles de aislamiento para que los bloqueos sean más o menos restrictivos en cuanto a operaciones de lectura y escritura que puedan generar bloqueos. Cuando hablamos de aislamiento, nos referimos a cómo se ven los cambios realizados por una transacción antes de que finalice. Los niveles de aislamiento definidos por el estándar SQL son:

- **Lectura no confirmada (read uncommited):** No genera bloqueos, y por lo tanto los cambios que va realizando una transacción pueden ser vistos por el resto. No hay restricciones, por lo que puede generar los tres tipos de problemas de concurrencia.
- **Lectura confirmada (read commited):** Bloquea las operaciones de escritura, pero no las de lectura, esto puede dar lugar a los errores de lectura fantasma o lectura no repetible. Solamente protege contra lectura sucia.
- **Lectura repetible (repeteable read):** Se bloquean las operaciones de lectura y escritura sobre los datos seleccionados hasta que finalice la transacción. No se gestionan los bloqueos de rango, por lo que pueden ocurrir lecturas fantasma.
- **Serializable:** Este es el nivel más restrictivo, las transacciones bloquean los recursos que van a emplear antes de comenzar y el resto de transacciones que hagan uso de los mismos recursos bloquedaos, esperan en cola y se van ejecutando en serie según van liberándose.

Niveles de aislamiento y problemas				
Nivel de aislamiento	Lectura sucia	Lectura no repetible	Lectura fantasma	Interbloqueo
Lectura no confirmada	V	V	V	-
Lectura confirmada	-	V	V	V
Lectura repetible	-	-	V	V
Serializable	-	-	-	-

Bloqueos según nivel de aislamiento			
Nivel de aislamiento	Bloqueo de escritura	Bloqueo de lectura	Bloqueo de rango
Lectura no confirmada	-	-	-
Lectura confirmada	V	-	-
Lectura repetible	V	V	-
Serializable	V	V	V

4.3.3. Recuperación

Para garantizar la disponibilidad de la información y recuperación ante desastres, los SGBD incorporan mecanismos de copia y restauración. Los mecanismos principales que podemos encontrar en un SGBD son:

- **Copias de seguridad:** Permiten importar y exportar la información a un soporte externo.
- **Cuaderno de bitácora:** Guarda el registro de operaciones transaccionales. Si se recibe un rollback, se eliminan las operaciones del diario. Regularmente, el SGBD registra puntos de restauración, en la copia de seguridad, lo que permite ejecutar todas las operaciones confirmadas en el diario de transacciones.

Las tareas del DBA referentes a la gestión de copias de seguridad son tres:

- **Backup:** Guardar regularmente la copia de los ficheros de la base de datos en un fichero externo, en base a las políticas de seguridad definidas.
- **Restore:** Recuperar los datos de los ficheros copiados.
- **Recovery:** Volver a un estado previo a un fallo, se ejecutarán todas las operaciones registradas en el cuaderno de bitácora posteriores al punto de restauración.

En relación con estas tareas, el administrador de base de datos tiene la responsabilidad de definir las políticas de copia de seguridad (completas, parciales y configuración de los archivos del cuaderno de bitácora), el procedimiento de restauración ante fallos, que debe ser lo más rápido posible y debe preparar también simulacros de recuperación.

4.3.4. Tipos de copias de seguridad

Podemos clasificar los tipos de copias de seguridad en diferentes aspectos:

Copias Totales o parciales

Cuando realizamos una copia total, en ella incluimos todos los objetos de la base de datos, archivos de datos, control, logs, cuaderno de bitácora, ficheros de configuración, etc., mientras que, en una copia parcial, especificaremos los objetos o partes de la base de datos de los que queremos realizar la copia. Cuando realizamos una migración a otro SGBD distinto, se consideraría copia parcial, pues muchos de los archivos de configuración e incluso objetos no son compatibles y se tienen que adaptar.

Copias Lógicas o físicas

Las copias lógicas permiten guardar objetos de la base de datos (tablas, índices, etc.), mientras que las copias físicas almacenan archivos o bloques. Con los dos tipos de copias se puede guardar la base de datos completa, solo que se guardan en diferente formato.

Copias Completas o incrementales

Las copias incrementales permiten guardar una copia de las variaciones ocurridas desde la última copia completa. Tienen la ventaja de que son más rápidas y ocupan menos espacio.

Copias Online / Offline

La copia Offline, se hace con la base de datos cerrada, lo cual permite una copia consistente de los datos, aunque tiene la desventaja de que se requiere dejar de trabajar con ella, y esto no siempre es posible. La copia online en cambio se realiza cuando se está trabajando con la base de datos. Esto supone que se van generando cambios mientras dura el proceso de copia. La copia online es menos consistente y se tiene que apoyar en el cuaderno de bitácora.

4.4. Integridad en Oracle

Oracle permite habilitar o deshabilitar las restricciones de forma individual:

```
ALTER TABLE <tabla> {ENABLE | DISABLE} <restricción>;
```

La vista DBA_CONSTRAINTS permite ver el listado de restricciones.

4.4.1. Restricciones en Oracle

Restricciones en Oracle		
Tipo de restricción	**Sintaxis**	**Descripción**
Nulidad	`NOT NULL`	La columna no permite valores nulos.
Valor por defecto	`DEFAULT [Valor]`	Es el valor que adquiere la columna cuando no se especifica ninguno.
Clave primaria	`PRIMARY KEY`	Se indica que la columna es clave primaria
Unicidad	`UNIQUE`	Se impone la restricción en un conjunto de columnas cuyos valores no se pueden repetir.
Comprobación	`CHECK`	Se especifica la condición que debe cumplir el valor de una columna. Si no se cumplen, no se permite la inserción o modificación de los datos.

4.4.2. Control de concurrencia en Oracle

Para reducir tiempos de espera en la ejecución de transacciones en serie, Oracle bloquea primero lo mínimo posible (por ejemplo, una fila), y se va extendiendo el bloqueo a otros niveles si es necesario. De esta forma, se optimiza el trabajo concurrente.

Oracle implementa tres tipos de aislamiento:

- **Lectura confirmada (read commited):** nivel de aislamiento por defecto.
- **Serializable**
- **Read only:** las transacciones solo ven los datos confirmados antes de empezar la transacción y no podrán realizar modificaciones sobre los datos.

Oracle emplea grafos de espera para encontrar problemas de interbloqueos. Cuando detecta alguno, realiza un rollback de la transacción que entró más tarde.

Los niveles de aislamiento en Oracle se pueden modificar a nivel de sesión o por cada transacción particular. La sintaxis para modificar los niveles de aislamiento es:

```
ALTER SESSION SET TRANSACTION ISOLATION LEVEL
SERIALIZABLE;
SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;
```

4.4.3. Tipos de copias de seguridad en Oracle

Podemos hacer uso de tres herramientas para realizar las copias de seguridad en Oracle: Export/import, datadump y RMAN.

Export/Import

Los comandos exp/imp permiten hacer copias lógicas que se guardarán en el equipo local donde se realice la copia. Las copias pueden ser de cualquier tipo (totales/parciales, completas, online/offline). Cuando se hace de forma online, se produce un bloqueo de la tabla que se está exportando.

El fichero de backup se crea con el comando "exp", y para restaurar la base de datos se utiliza "imp".

La aplicación de este tipo de backup suele ser en los siguientes casos:

- Copias locales cuando un usuario no tiene acceso al servidor.
- Copias de bases de datos pequeñas.
- Corregir problemas de fragmentación.
- Migración de una base de datos a otro servidor.

```
c:\>exp -help
```

Imagen 20. Consultar la ayuda del comando exp

```
Palabra clave    Descripci¾n (por defecto)      Palabra clave      Descripci¾n (por defecto)
------------------------------------------------------------------------------
USERID     usuario/contraseła          FULL         exportar archivo completo (N)
BUFFER      tamało buffer de datos        OWNER        lista de usuarios propietarios
FILE       archivos de salida (EXPDAT.DMP)  TABLES       lista de nombres de tabla
COMPRESS   importar en una extensi¾n (Y) RECORDLENGTH longitud de registro E/S
GRANTS     exportar permisos (Y)          INCTYPE      exportaci¾n incremental
INDEXES    exportar Ýndices (Y)         RECORD       exportaci¾n incremental de seguimiento (Y)
DIRECT     ruta de acceso directa (N)           TRIGGERS     exportar disparadores (Y)
LOG        archivo log de salida de pantalla  STATISTICS   analizar objetos (ESTIMATE)
ROWS       exportar filas de datos (Y)       PARFILE      archivo de parßmetros
CONSISTENT consistencia de referencias cruzadas (N)    CONSTRAINTS  exportar restricciones (Y)

OBJECT_CONSISTENT    transacci¾n definida en s¾lo lectura durante la exportaci¾n del objeto (N)
FEEDBACK             mostrar progreso cada x filas (0)
FILESIZE             tamało mßximo de cada archivo de volcado
FLASHBACK_SCN        SCN utilizado para volver a definir instantßneas de sesi¾n en
FLASHBACK_TIME       tiempo utilizado para obtener el SCN mßs pr¾ximo al tiempo especificado
QUERY                seleccionar clßusula utilizada para exportar un subjuego de una tabla
RESUMABLE            suspender cuando se produce un error relacionado con el espacio (N)
RESUMABLE_NAME       cadena de texto utilizada para identificar la sentencia de reanudaci¾n
RESUMABLE_TIMEOUT    tiempo de espera para RESUMABLE
TTS_FULL_CHECK       ejecutar comprobaci¾n de dependencia total o parcial para TTS
TABLESPACES          lista de los tablespaces que se van a exportar
TRANSPORT_TABLESPACE exportar metadatos de tablespaces transportables (N)
TEMPLATE             nombre de la plantilla que llama a la exportaci¾n del modo iAS
```

Imagen 21. Opciones disponibles para exp

Vamos a ver un ejemplo de cómo exportar la tabla employees del equipo HR. Para ello, debemos seguir los siguientes pasos:

1. Escribir el comando exp en línea de comandos:
2. Indicar usuario y contraseña (HR)
3. Seleccionar el tamaño del buffer por defecto y el nombre del archivo (4096, export.DMP)
4. Seleccionar la opción esquema o tabla e indicar si el archivo debe comprimirse.

Para realizar el proceso inverso, sigue el mismo proceso, con el compando imp y especificando el fichero que previamente ha sido exportado.

Data Pump

Esta herramienta, viene con la instalación de Oracle desde la versión 10g. Nos va a permitir realizar migraciones entres bases de datos de Oracle.

Funciona con los comandos expdp e impdp También funcionan mediante la interfaz gráfica de EM.

Con esta herramienta, también podremos crear copias totales o parciales, completas, lógicas y en modo online/offline.

Esta herramienta es más rápida que imp/exp, pues permite la importación en paralelo, mientras que expdp e impdp, mejora el rendimiento si la base de datos tiene los ficheros o discos.

Los procesos de Data Pump, pueden ser reiniciados sin pérdida de datos, así como seleccionar todo tipo de objeto (Excepto un job Data Pump), filtrando incluso si es necesario los metadatos que van a ser incluidos en la copia.

Data Pump no permite descargar una copia en local. Los ficheros deben ser exportados e importados desde el propio servidor. Además de los comandos y del EM, también puede ser invocado desde el paquete DBMS_DATAPUMP, donde, además, se pueden consultar los Jobs creados.

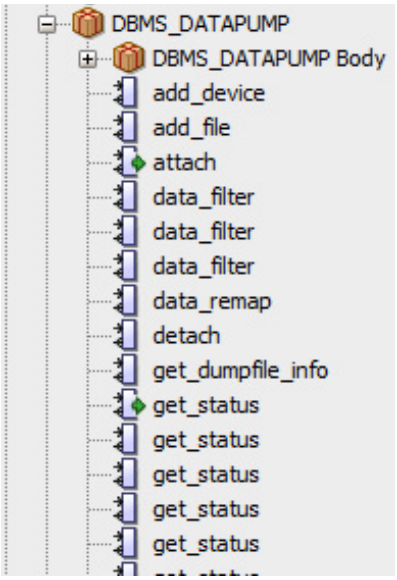

Imagen 22. Funciones y procedimientos del paquete DBMS_DATAPUMP

RMAN (Recovery Manager)

Las copias de seguridad que se realizan con esta herramienta son a nivel físico, permitiendo hacerlos de forma incremental, es decir copiar solamente los bloques de datos que hayan sido modificados desde la última copia.

Las copias se pueden realizar a nivel de base de datos completa, tablespace o datafile.

La ventaja de esta herramienta es que, al ser a nivel del bloque de datos, no se requiere respaldar bloques vacíos, ocupando menos espacio y maximizando la compresión de los datos.

Este tipo de copias puede hacerse de forma online, por lo que se puede tener un servidor de copias de seguridad que se actualice a partir del principal, y en caso de fallo, usar el de respaldo para restablecer el servicio rápidamente.

RMAN funciona leyendo las operaciones almacenadas en el cuaderno de bitácora y enviando esas operaciones al servidor de respaldo para ejecutarlos allí.

Para poder restaurar con RMAN, se necesita una copia incremental.

Con los paquetes DBMS_RCVMAN y DBMS_BACKUP_RESTORE, podremos realizar copias directamente desde comandos SQL.

Las opciones de copia y restauración en RMAN difieren bastante de los comandos de los otros dos métodos.

4.5. Normativa de protección de datos

El administrador de bases de datos, que es el responsable de gestionar los datos almacenados en la organización, debe garantizar que los accesos a dichos datos cumplen con la normativa vigente:

- UE 2016/679 27 de Abril de 2016 del Parlamento Europeo relativo al tratamiento y libre circulación de datos de carácter personal.
- 3/2018, 5 de Diciembre. LOPD (Ley Orgánica de Protección de Datos personales y garantía de los derechos digitales

4.5.1. LOPD

La LOPD es una ley española que establece las normas a seguir para el tratamiento de datos personales. Se aplica a todas las empresas y organizaciones que recopilan, almacenan o utilizan datos personales de cualquier tipo, ya sean físicos, electrónicos o de cualquier otro tipo. La ley establece un conjunto de reglas a seguir para garantizar que se protejan adecuadamente los datos personales y se respeten los derechos de privacidad de las personas.

Determina qué datos están sometidos a cuatro derechos de sus propietarios:

- **Acceso:** El propietario puede conocer y obtener gratuitamente información sobre sus datos personales sometidos a tratamiento.
- **Rectificación:** El propietario puede corregir, modificar o completar datos con el fin de garantizar la certeza de la información sometida a tratamiento.
- **Cancelación:** debe permitirse al propietario que se supriman los datos que resulten ser inadecuados o excesivos (fuera del contexto o propósito para el que fueron recopilados) sin perjuicio del deber de bloqueo según recoge la normativa.
- **Oposición:** Derecho del propietario a que no se traten sus datos de carácter personal.

Previo a la recolección de ningún tipo de información, se debe catalogar el tipo de datos que se van a tratar y clasificarlos en tres niveles de seguridad:

- **Básico:** Los ficheros con datos de carácter personal tienen que adoptar las medidas de nivel básico. Estas medidas son la gestión de accesos por parte de los propios usuarios, copias de seguridad y custodia de los soportes de almacenamiento.
- **Medio:** Estos datos se aplican a datos referentes a infracciones, tributos, seguridad social o determinadas características relacionadas con la personalidad de los propietarios. Las medidas a aplicar, además de las de nivel básico son: identificación de los responsables de seguridad, registro físico de acceso, entrada y salida de soportes, auditorías periódicas y asegurar la destrucción de la información cuando sea desechada.
- **Alto:** Estos datos se aplican a datos con información sobre ideología, religión, salud o vida sexual y datos policiales obtenidos sin consentimiento de los afectados. Deben almacenarse en soportes cifrados y protegidos con llave. La transferencia de estos datos debe ser cifrada. El incumplimiento de estas normas para este nivel puede llevar multas de entre 300 y 600 000 €.

PARA TENER EN CUENTA...

Las organizaciones que traten con datos de carácter personal deben disponer de un documento de seguridad, actualizado y cuyo contenido mínimo debe ser el siguiente:

1. Ámbito de aplicación: especificación de los recursos protegidos.
2. Medidas, normas y procedimientos de obligación del personal de la organización.
3. Estructura y descripción de los ficheros y sistemas de información
4. Protocolo de gestión ante incidentes.
5. Procedimiento de copias de respaldo y recuperación.
6. Protocolo de transporte, destrucción y reutilización de soportes de datos.

Cada organización debe notificar a la Agencia Española de Protección de Datos la información referente a los ficheros de carácter personal para su inscripción en el Registro General de Protección de Datos (RGPD).

4.5.2. Herramientas del SGBD para cumplir con la LOPD

El administrador de base de datos deberá hacer uso de las herramientas que proporciona el SGBD para garantizar el cumplimiento de la normativa. Esas herramientas son:

- **Usuarios y Permisos:** debe realizarse una apropiada gestión para garantizar que los usuarios solamente pueden acceder a la información a la que se le haya dado permiso y que además solo podrán realizar las operaciones permitidas sobre esos datos.
- **Mecanismos de copias de seguridad y restauración:** son fundamentales para mantener la información a salvo y garantizar la custodia de los datos.
- **Integridad:** utilizando apropiadamente las restricciones, y transacciones, garantizaremos que los datos son correctos.
- **Auditoría:** Con unas reglas bien definidas, permiten registrar todos los accesos realizados a los datos, así como las operaciones realizadas sobre ellos. Es común definir políticas de auditoría que registre cualquier tipo de incumplimiento sobre la LOPD.
- **Encriptación:** La encriptación tanto en los datos como en las comunicaciones es fundamental para garantizar que no hay accesos a la información de usuarios no autorizados para el tratamiento de datos.

www.universae.com