ACCESO A DATOS

UNIDAD 01 – INTRODUCCIÓN A LOS GESTIÓN EMPRESARIAL

INDICE

1.1 – PROGRAMAS Y DATOS

- Los ordenadores ejecutan programas que gestionan información.
- La información se representa en forma de datos estructurados.
- Da igual el tipo de ordenador y el tipo de dato que manejen, el ordenador cuenta con dos tipos de medios de almacenamiento:
  - Almacenamiento primario o memoria principal.
  - Almacenamiento secundario.

1.1 – PROGRAMAS Y DATOS

- **Almacenamiento primario** o memoria principal.
  - Datos con los que se está trabajando.
  - Su contenido se borra al apagar el ordenador.
  - Capacidad baja.
  - Tiempos de acceso rápidos.

1.1 – PROGRAMAS Y DATOS

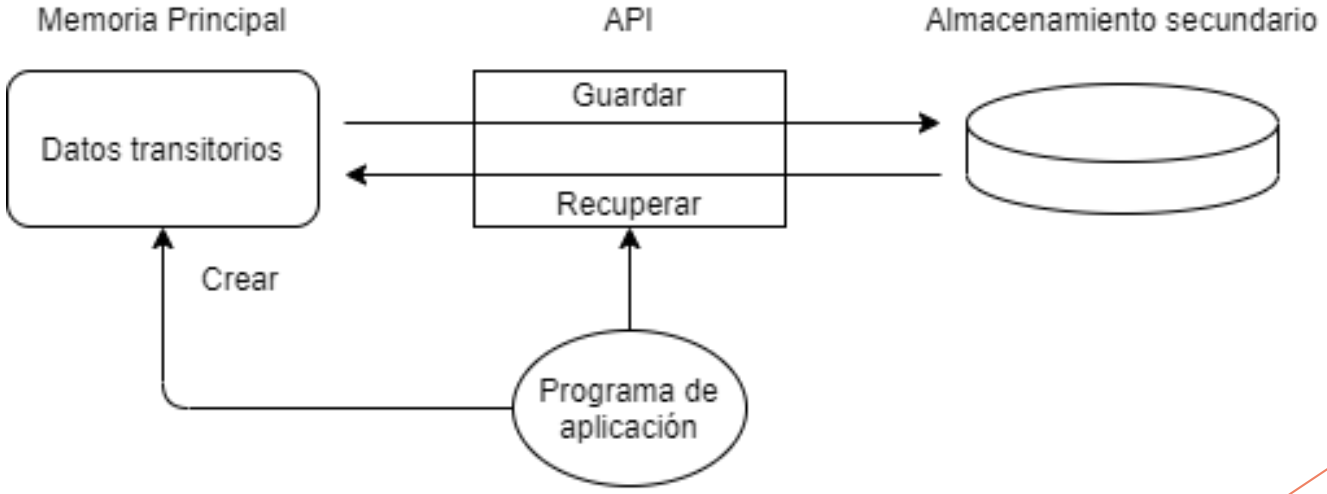
- **Almacenamiento secundario.**
  - Su contenido se guarda de forma permanente.
  - Capacidad alta.
  - Tiempos de acceso relativamente altos.
  - Los datos almacenados se consideran *datos persistentes* puesto que no se borran al apagar el ordenador

1.2 – PERSISTENCIA DE DATOS

- Los programas solo pueden consultar directamente datos de memoria principal.
- Un programa puede crear datos en memoria principal y guardarlo en memoria secundaria, haciendo los datos persistentes.

1.2 – PERSISTENCIA DE DATOS

- Es habitual que el almacenamiento secundario esté en un ordenador distinto del que ejecuta los programas.
- La comunicación se suele hacer mediante funciones de alto nivel proporcionadas por API (interfaces de programación de aplicaciones.



1.2 – PERSISTENCIA DE DATOS

- En última instancia los datos se almacenan en ficheros dentro del *sistema de archivos o ficheros* del almacenamiento secundario.
- Un Sistema de ficheros consiste en una jerarquía de directorios o carpetas en la que puede haber ficheros que consisten básicamente en una secuencia de bytes.
- Sobre este sistema de ficheros se pueden organizar el almacenamiento de diferentes formas.

1.2 – PERSISTENCIA DE DATOS

- A la hora de decidir respecto a un sistema de almacenamiento, hay que tener en cuenta, no solo las necesidades actuales, sino también las futuras (escalabilidad) así como las posibilidades de crecimiento en carga de trabajo.

1.3 – ALMACENAMIENTO DE DATOS

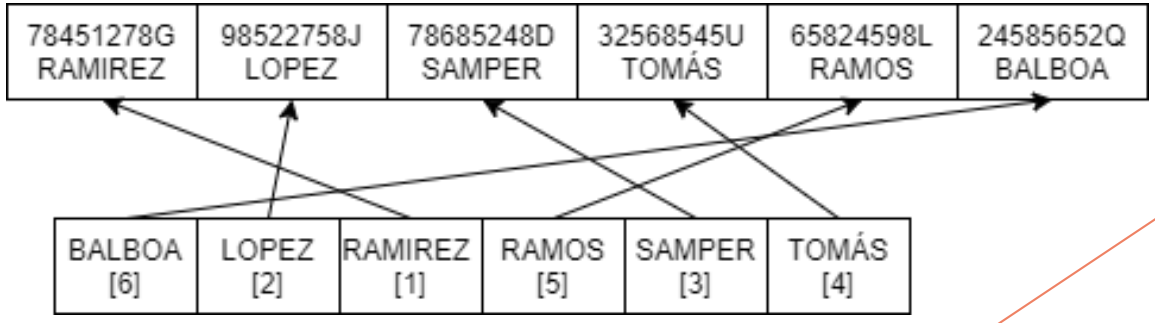
- Cada sistema proporciona una estructura básica de almacenamiento de datos.
  - Ficheros
  - BD Relacionales
  - Documentos XML
  - BD de objetos
  - BD NoSQL

1.3 – ALMACENAMIENTO DE DATOS

- Los **ficheros** proporcionan una organización secuencial de los datos.
- Son secuencias de bytes.
- Se puede representar cualquier tipo de información.
- Ampliamente utilizado hasta la irrupción de las BBDD relacionales.
- La información se suele almacenar en registros de longitud fija y cada registro compuesto por varios campos de longitud fija.
- Se le llama *fichero secuencial* porque está formado por una secuencia de registros

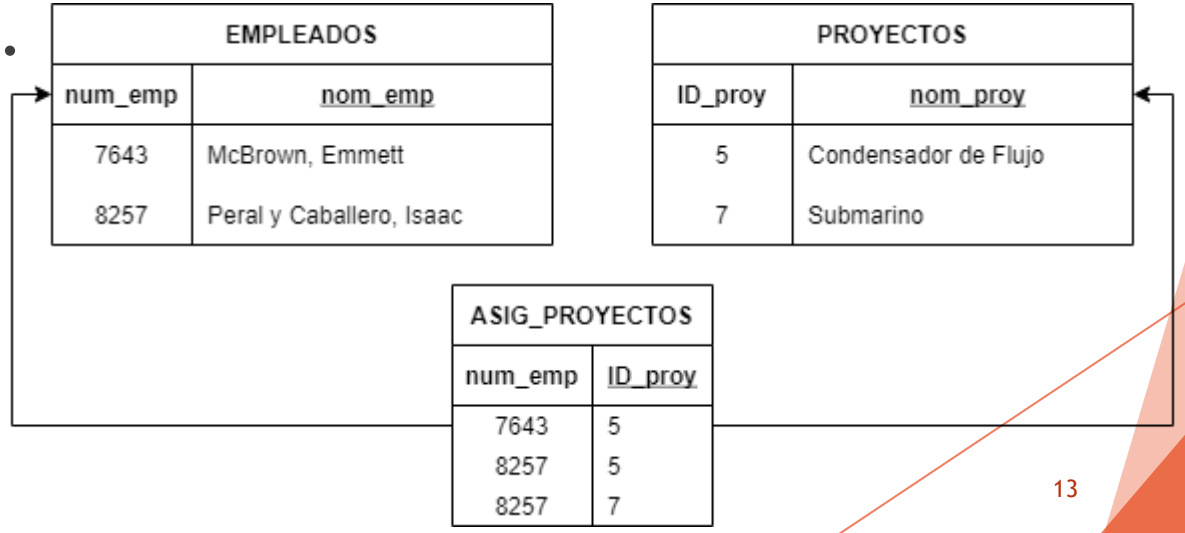
1.3 – ALMACENAMIENTO DE DATOS

- Para acelerar las consultas sobre ficheros secuenciales se suele crear un fichero de índice.
- En este caso pasan a llamarse ficheros secuenciales indexados.
- Se pueden crear tantos índices como queramos.
- El fichero índice no deja de ser un tipo particular de fichero secuencial.



1.3 – ALMACENAMIENTO DE DATOS

- Las **BBDD relacionales** organizan los datos en tablas y permiten especificar las relaciones entre dichas tablas.
- A partir de los 80 desplazaron a los ficheros como almacenamiento.
- SQL es el lenguaje estándar, por ello las BBDD no relacionales son llamadas NoSQL.



EMPLEADOS

num_emp	nom_emp
7643	McBrown, Emmett
8257	Peral y Caballero, Isaac

PROYECTOS

ID_proy	nom_proy
5	Condensador de Flujo
7	Submarino

ASIG_PROYECTOS

num_emp	ID_proy
7643	5
8257	5
8257	7

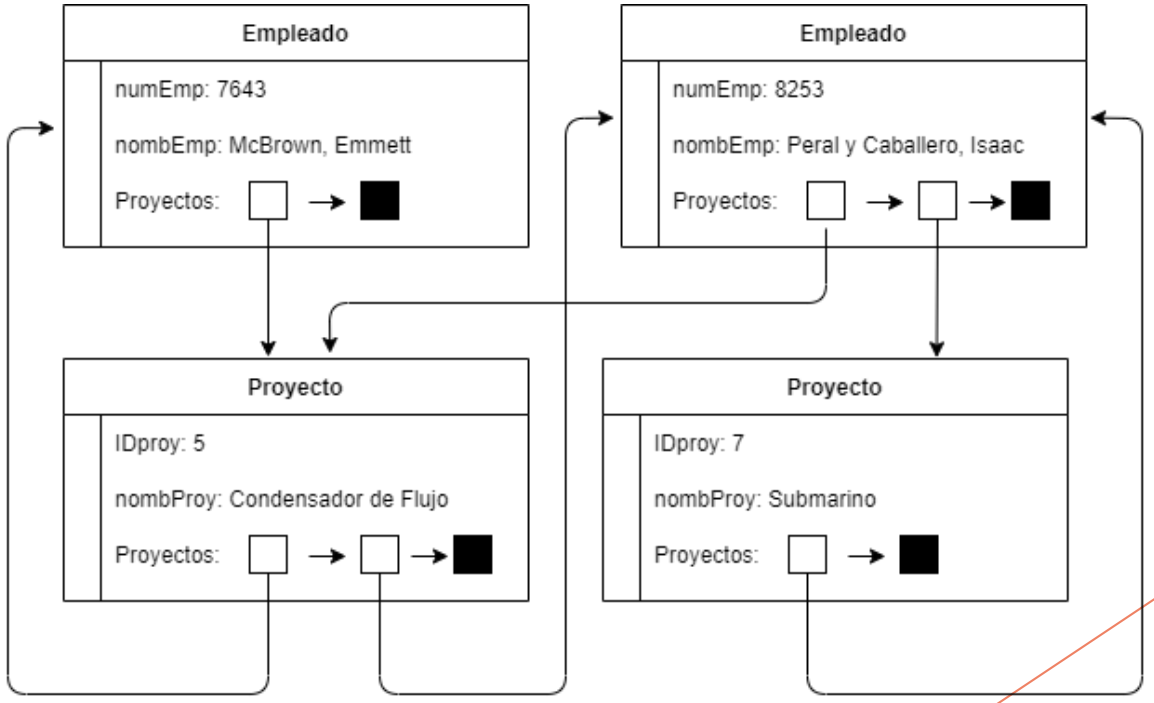
1.3 – ALMACENAMIENTO DE DATOS

```
<?xml version="1.0" encoding="UTF-8"?>
<Dioses>
  - <DIOSES Número="1">
        <DIOS_GRIEGO>ZEUS</DIOS_GRIEGO>
        <DIOS_ROMANO>JÚPITER</DIOS_ROMANO>
        <GENERACIÓN>1</GENERACIÓN>
        <AMBITO>REY_DE_DIOSES</AMBITO>
    </DIOSES>
  - <DIOSES Número="2">
        <DIOS_GRIEGO>HERA</DIOS_GRIEGO>
        <DIOS_ROMANO>JUNO</DIOS_ROMANO>
        <GENERACIÓN>1</GENERACIÓN>
        <AMBITO>REINA_DE_DIOSES</AMBITO>
    </DIOSES>
  - <DIOSES Número="3">
        <DIOS_GRIEGO>POSEIDÓN</DIOS_GRIEGO>
        <DIOS_ROMANO>NEPTUNO</DIOS_ROMANO>
        <GENERACIÓN>1</GENERACIÓN>
        <AMBITO>SEÑOR_DE_LOS_MARES</AMBITO>
    </DIOSES>
  - <DIOSES Número="4">
        <DIOS_GRIEGO>APOLO</DIOS_GRIEGO>
        <DIOS_ROMANO>FEBO</DIOS_ROMANO>
        <GENERACIÓN>2</GENERACIÓN>
        <AMBITO>MÚSICA_Y_ARTES</AMBITO>
    </DIOSES>
</Dioses>
```

- Los **documentos XML** organizan la información de forma jerárquica, es decir en forma de árbol.
- El modelo DOM en un modelo estándar de representación.
- Se puede almacenar en un fichero de texto.
- También se puede almacenar en una BD de XML.

1.3 – ALMACENAMIENTO DE DATOS

- Las **BD de objetos** almacenan objetos.
- Un objeto complejo puede contener referencias a otros objetos.
- Una colección de objetos complejos tiene una estructura de grafo.



1.3 – ALMACENAMIENTO DE DATOS

- Las diversas **BBDD NoSQL** representan la información de manera diversa.
- En general lo hacen de forma sencilla pero flexible.
- **Restricciones de integridad**: Son condiciones que siempre deben cumplir los datos almacenados. Pueden ser para datos particulares (valor no nulo) o datos relacionales (el cliente de una factura debe existir).
- Se utilizan **iteradores** (a veces llamados cursores) para obtener resultados de consultas a diversos sistemas de persistencia.
- Un iterador se crea para una consulta (o modificación) y permite obtener los resultados uno a uno.

1.3 – ALMACENAMIENTO DE DATOS

- El acceso concurrente
  - Los datos almacenados pueden compartirse con lo que pueden haber lecturas y escrituras concurrentes (simultáneas).
  - Se debe gestionar los accesos para evitar problemas de modificaciones simultáneas.
- Las transacciones
  - Es muy habitual que un grupo de operaciones sobre datos relacionados formen un todo que debe llevarse a cabo conjuntamente y de manera aislada con respecto a otras operaciones.

1.3 – ALMACENAMIENTO DE DATOS

- Las BBDD Relacionales proporcionan, por lo general, soporte para transacciones, mientras que las BBDD NoSQL no lo hacen.
- Las características de una transacción (ACID) son:
  - Atomic (atómica) – Ejecución completa y sin errores.
  - Consistent (consistente) – Cumple restricciones de integridad.
  - Isolated (aislada) – No interfieren otras ejecuciones.
  - Durable (duradera) – Al finalizar se confirman los cambios.
- Desventaja de las transacciones: pueden disminuir el rendimiento.

1.4 – PERSISTENCIA EN FICHEROS

- Almacenamiento más elemental.
- En esencia una secuencia de bytes, con lo que puede almacenar cualquier tipo de información.
- Un fichero tiene un nombre y está situado en un directorio dentro de la jerarquía de directorios.
- En última instancia, todos los sistemas de almacenamiento utilizan ficheros.
- En aplicaciones sencillas se siguen utilizando ficheros de texto.

1.4 – PERSISTENCIA EN FICHEROS

- Problemas con ficheros de Texto.
  - Consultas complejas o relaciona mucha información diversa.
  - El volumen de datos a manejar es muy grande.
  - Muchas operaciones de borrado o modificación.
  - Permitir el acceso simultáneo o transacciones complica mucho los programas.
  - Evitar redundancias e inconsistencia de datos así como preservar restricciones de integridad.

1.4 – PERSISTENCIA EN FICHEROS

- Por todos estos motivos, han sido relegados en favor de las bases de datos relacionales.
- Antes de las BBDD Relaciones se almacenaba de forma habitual en ficheros indexados. Ejemplo lenguaje COBOL.
- Se siguen utilizando por ejemplo con en los archivos XML, procesos masivos que se ejecutan periódicamente o puntualmente y para copias de seguridad.

1.5 – PERSISTENCIA EN BBDD

- La persistencia en BBDD Relaciones en con muchísima diferencia, los más utilizados en la actualidad.
- Basados en el modelo relacional son a la vez sencillos e intuitivos.
- La estructura básica de almacenamiento es la tabla.
- Existe un lenguaje estándar y soportado universalmente: SQL.
- SQL permite crear esquemas (colección de tablas), definir restricciones de integridad, etc.
- SQL es un lenguaje declarativo de alto nivel, definiendo qué se quiere y que sea el sistema el encargado de realizarlo de la forma más eficiente.

1.5 – PERSISTENCIA EN BBDD

- El sistema crea índices automáticamente pero se pueden definir manualmente para optimizar nuestras consultas.
- Las BBDD Relacionales actuales son muy escalables y tienen muy buen soporte para transacciones.
- Tienen mecanismos de copia de seguridad y restauración ante fallos.
- Existen API para BBDD relacionales para todos los lenguajes de programación ampliamente utilizados. P.ejem: JDBC para Java.

1.5 – PERSISTENCIA EN BBDD

- Las principales BBDD Relacionales se han dotado de funcionalidades para facilitar la persistencia de datos en formato XML.
- Permite realizar consultas SQL tanto sobre tablas como sobre documentos XML.
- Se puede obtener resultados en forma de tabla o como XML.
- Las BBDD que implementan SQL/XML se califican como *XML-enabled*.

1.5 – PERSISTENCIA EN BBDD

- El inconveniente de usar BBDD relacionales con lenguajes de objetos es que almacenar objetos no es sencillo, sobre todo con objetos complejos.
- Han surgido varios planteamientos:
  - BBDD Objeto-relacionales: con capacidades para gestionar objetos (introducido en SQL:99). Por ejemplo Oracle y PostgreSQL.
  - Correspondencia objeto-relacional o mapeo objeto-relacional (ORM): Solución más flexible, con soporte para más BBDD. Existen múltiples herramientas entre las que destaca Hibernate.

1.5 – PERSISTENCIA EN BBDD

- Las BBDD de objetos permiten almacenar directamente objetos.
- Como ejemplos tenemos Matisse y db40.
- Inconvenientes:
  - Falta de modelo formal y ampliamente aceptado.
  - Falta de estándares ampliamente adoptados.

1.5 – PERSISTENCIA EN BBDD

- La importancia del XML ha hecho que se desarrollen BBDD de XML nativas.
- Tienen estructuras y mecanismos de almacenamiento de datos diseñados y optimizados específicamente para XML.
- Tienen soporte para lenguajes estándares que permiten operaciones sobre documentos XML (Xpath, Xquery).
- Suelen organizar los documentos en colecciones y algunas permiten almacenar no solo documentos XML.

1.5 – PERSISTENCIA

ACTIVIDADES PROPUESTAS.

a) Haz un esquema con las distintas posibilidades disponibles para almacenar tanto objetos como documentos XML en diversos tipos de bases de datos, al menos: relacionales, de objetos, objetos-relacionales y de XML nativas. Si existe un nombre específicopara la tecnología que hace posible el almacenamiento de un tipo particular de datos en un tipo particular de BD, indícalo.

1.5 – PERSISTENCIA

ACTIVIDADES PROPUESTAS.

b) Para cada tipo de base de datos vista hasta ah
una lista, por una parte, de estándares y por otra parte, de API, e indica en pocas palabas su propósito o utilidad. Con frecuencia, una API proporciona la implementación de uno o varios estándares. Indica, siempre que se pueda y sea relevante, los estándares con los que se relaciona, se implementa o donde se basa una API. Por ejemplo: la API XQJ (Xquery for Java) es una API que permite ejecutar sentencias del lenguaje estándar Xquery.

1.5 – PERSISTENCIA EN BBDD

- Dentro de las BBDD NoSQL se puede incluir las BBDD de Objetos y XML. Nosotros las excluiremos.
- El auge de estas BBDD se debe a la necesidad de recopilar, gestionar y analizar gigantescos conjuntos de datos heterogéneos que crecen continuamente. *Big Data*.
- Dan servicio a IoT, y aplicaciones que ofrecen servicio a través de la web a usuarios que no solo consultan información sino que añaden y modifican de forma continua.
- En sentido amplio, NoSQL es “todo aquello que no se adhiere 100% al modelo relacional y sus características”. Es decir:
  - Almacenamiento basado en tablas.
  - SQL como lenguaje vehicular.
  - Soporte para restricciones de integridad y transacciones.

1.5 – PERSISTENCIA EN BBDD

- Algunas características comunas a estas BBDD NoSQL son:
  - Almacenamiento no basado en tablas. Arrays asociativos clave-valor(Redis), documentos (Mongo-DB) o en columnas (Cassandra).
  - No usan SQL. Cada uno usa su propio lenguaje, por lo general no declarativo. Puede necesitar escribir programas con instrucciones muy detalladas.
  - Prima la disponibilidad en lugar de las transacciones ACID. Conocido como BASE (basic availability, soft state, eventual consistency).

1.5 – PERSISTENCIA

ACTIVIDADES PROPUESTAS.

a) Haz un cuadro resumen en forma de tabla incluyendo filas para los distintos tipos de sistemas de persistencia de datos, y columnas para diversos aspectos relevantes tales como: estructura básica para almacenamiento de datos, soporte para transacciones, grado de estandarización (es decir, existencia de estándares establecidos y ampliamente adoptados), madurez (es decir, si las tecnologías se han venido utilizando durante un periodo largo de tiempo, aunque no sea de manera muy extendida, con lo que ha habido tiempo para corregir errores, perfilar características, hacer ajustes, optimizar, adapta a nuevas situaciones), grado de implantación (uso extendido en la actualidad), y cualquier otro que consideres relevante. Se trata de dar una idea aproximada y muy sintética, con lo que para algunos de estos criterios basta indicar posibles valores como “alta”, “media”, “baja”, o bien “siempre” , “a veces”.

RESUMEN

- ✓ Los programas de aplicación trabajan con datos transitorios almacenados en memoria principal, y con datos persistentes almacenados en medios de almacenamiento secundario.
- ✓ Existen diversos sistemas de persistencia que permiten grabar datos o recuperarlos entre la memoria principal y el almacenamiento secundario.
- ✓ Las aplicaciones suelen usar API para usar los servicios de persistencia proporcionados por servidores a través de protocolos de red estándar.
- ✓ Hay distinto tipos de sistemas de persistencia de datos: basados en ficheros, BBDD Relacionales, Objeto-relacionales, BBDD de objetos, BBDD XML y otras denominadas NoSQL.

RESUMEN

- ✓ Cada sistemas de persistencia proporciona una estrucutra básica para el almacenamiento de datos: BBDD Relacionales → Tabla, BBDD XML → Docs. XML, BBDD de Objetos → Objetos complejos, NoSQL → diversas estructuras y ficheros → secuencias de bytes.
- ✓ Las API de acceso a datos persistentes suelen facilitar iteradores para obtener uno a uno los resultados de consultas.
- ✓ El control y sincronización de accesos concurrentes y transacciones suelen estar soportado por los sistemas de persistencia como las BBDD relaciones. No así por las BBDD NoSQL, donde lo que prima es la disponibilidad.
- ✓ Las BBDD Relacionales son, con diferencia, las más utilizadas. Se han incluido revisiones para permitir el almacenamiento de objetos y documentos XML.

RESUMEN

- El almacenamiento de objetos en BBDD Relacionales plantea problema, conocidos como desfase objeto-relacional, pero existen herramientas y técnicas ORM que lo hacen posible.
- Las BBDD de objetos permiten almacenar directamente objetos complejos que contienen referencias a otros objetos y a colecciones de objetos. Muchas dan soporte al estándar ODMG 3.0 (ODL para definición de datos y OQL para consulta).
- Las BBDD XML nativas suelen organizar los documentos en una jerarquía de colecciones que pueden contener documentos XML y de otros tipos. Proporcionan soporte para lenguajes estándar del W3C, como Xquery, XML Schema y SXL y para API estándares tales como XML:DB y XQJ.

Acceso a Datos

FIN DE LA UNIDAD

GRACIAS