

Introducción

Lunes, 27 de enero de 2020 09:33 a.m.

- Jorge Rodríguez Campos

- Base de Datos - Grupo 3

► Evaluación

- Exámenes - 55%
- Laboratorio - 10%
- Prácticas (complementarias) - 10 %

- Ejercicios - 10%
- Tareas - 5%
- Proyecto - 10 %

// Puntos extras

- Participaciones
 - Foro * • En clase
- Apuntes

// Notas

- Examen Final 1
 - 30% Promedio Semestre
 - 70% Examen
- Examen Final 2
 - 100% Examen

// Exentos

- Prom. Exámenes ≥ 7
- No reprobar más de 2 exámenes
- Existe reposición para examen 2, 3 y 4
- Entregar $\geq 80\%$ de ejercicios, prácticas y tareas
- Entregar proyecto final

► Descripción de apartados

Prácticas Complementarias

- Título
- Desarrollo
- Conclusiones

Tareas

- Conceptos a investigar

Ejercicios

- Serie por cada tema
- Se revisan y califican en clase
- * Ejercicio no calificado ≤ 4
- * Formato libre

Proyecto

- En equipo de 2

Foro

- Participar

1. Areas

- Conceptos a investigar
- En equipo de 2

Proyecto

- En equipo de 2

Toro

- Participar

* Punto extra

- Participar en grupo de usuarios - Google

<http://groups.google.com/group/bd-fi>

BD

- Teoría
 - prácticas
 - gpo - lab
- Práctica 0 (Lab)
 - Práctica Complementaria (Teo)

Libros

- └ Pedir acceso

Nota

- Práctica Laboratorio
y práctica complementaria
↳ Un sólo archivo

Correo

- bd-fi@googlegroups.com
- jorgerdc@google.com

About Course

► Objetivo

El alumno explicará los conceptos y principios en los que se fundamenta la teoría de Bases de Datos los cuales permitirán diseñar, usar e implementar sistemas de BDs.

► Temario

- 1) Sistemas de Bases de Datos
- 2) Modelo de datos
- 3) Modelo relacional
- 4) Diseño conceptual y lógico de BDs *
- 5) Modelo de datos extendido
- 6) Normalización
- 7) Lenguaje de definición de datos (DDL)
- 8) Lenguaje de manipulación de datos (DML)
- 9) Lenguaje de consulta de datos (DQL) *
- 10) Programación PL/SQL
- 11) Introducción a BN Normal

- | i) Programación PL/SQL J
- | ii) Introducción a BD NoSQL

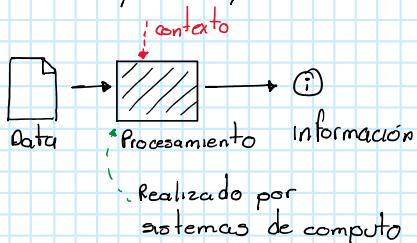
1. Sistemas de bases de datos

Tuesday, 28 January 2020 7:56 AM

► Datos e información

- Datos
 - Símbolo (Número, letra, figura, etc)
 - Empleado para representar a través de un lenguaje un hecho, condición, medida
 - Conjunto de símbolos en bruto (raw data) obtenidos a través de distintas medias

- Información
 - ¿Qué significado tiene un dato?
↳ Ninguno
 - ¿Qué sucede si a ese conjunto de datos se les aplica un procesamiento?



► Características

- Tiene un significado (Semántica)
- Tiene un valor
- Validez
- Tiene una vigencia
- Tiene una utilidad para obtener beneficios
↳ Toma de decisiones

Input Data (Entrada de datos)

Raw Data (Almacenamiento)

Estructura * Tipos de datos homogéneos

Procesamiento

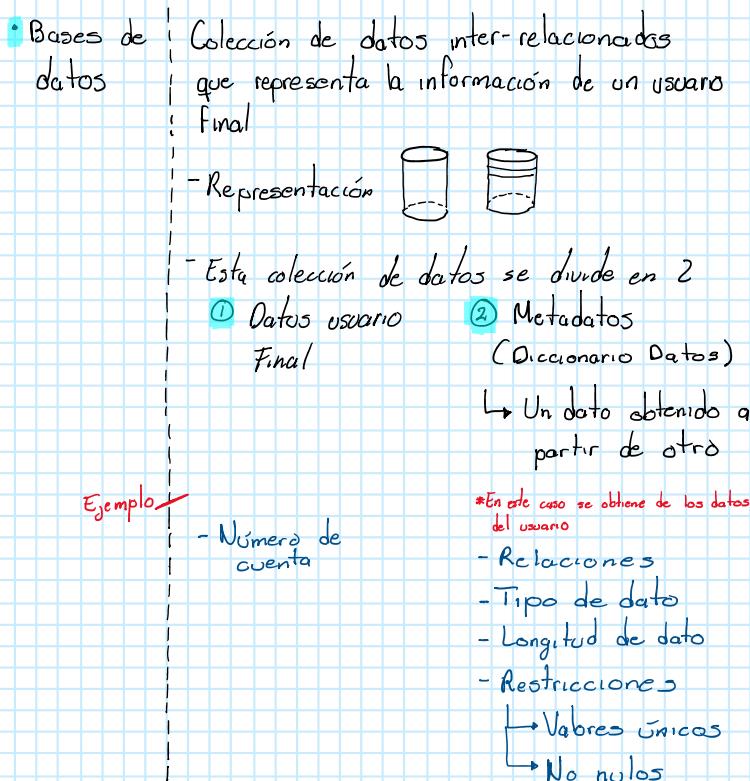
se da la toma de decisiones

1.2] Administración de los datos

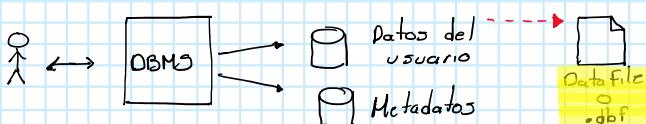
- Para obtener información útil y correcta, surgen ciertos requerimientos
 - Manejo de grandes volúmenes de datos
 - Almacenamiento
 - Datos consistentes e integros
 - Seguridad
 - Respaldos

¿Cómo implementar? → Manejador de un sistema de Bases de Datos
BDs

1) Base de datos y Sistema

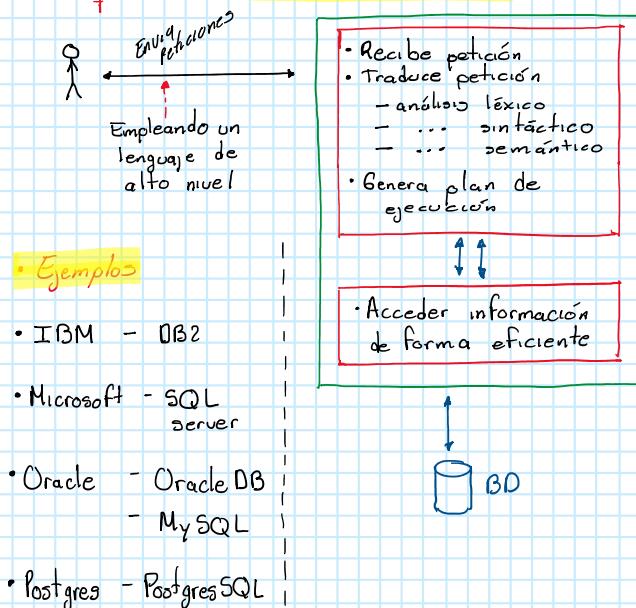


- Administrador de Bases de Datos (DBMS)
(Data Base Management System)
- Colección de programas (Software) encargado de administrar la estructura y el control de acceso de una BD
 - Actua como el intermediario entre el usuario final y la BD
 - La BD se almacena en archivos llamados **datatypes**
 - La **estructura física** de estos archivos es totalmente independiente a su **estructura lógica**

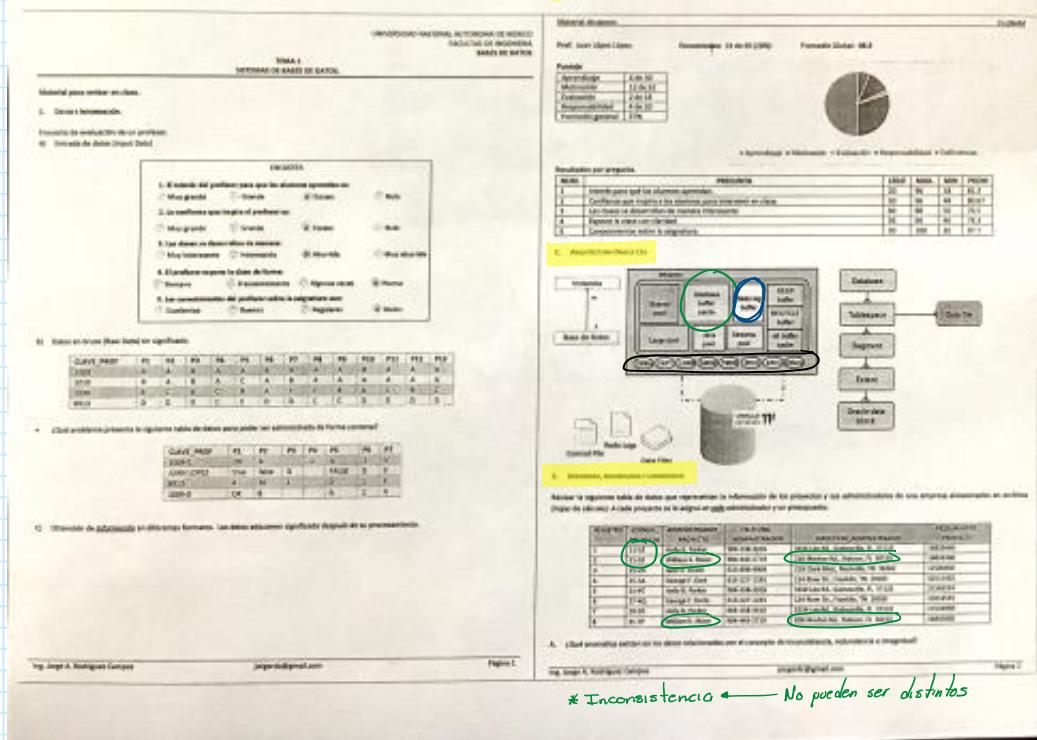


► Arquitectura Básica de un DBMS

1.4



- * Es lento consultar el disco → :: cache
- * Comando Linux → ps



- Proceso que van de fondo o Background
- Primero se checa el Database, posterior se checa en disco

* TODO se comba en el cache

- Se guarda una copia del cambio del cache

↳ Archivos pequeños, económicos

► Tipos de Bases de Datos

- | | |
|---------------------------------|---|
| → Por el número de usuarios | → Por la ubicación |
| - BD mono-usuario (H2, MySQLDB) | - BD centralizadas |
| - BD workgroup ≈ 50 personas | - BD distribuidas |
| - BD enterprise > 50 personas | <ul style="list-style-type: none"> • BD en la nube |

- Para el uso
 - BD operacionales o transaccionales (OLTP)
(On-line Transaction Processing)
 - BD analíticas o Data Warehouse (OLAP)
(On-line - Analytical Processing)

- BD operacionales o transaccionales (OLTP)
(Online - transaction Processing)
- * Gran carga de cambios o movimiento de datos
- BD analíticas o DataWarehouse (OLAP)
(Online - Analytical Processing)
- * Para tener datos sin tener más que consulta

→ Por el modelo de datos

- BD jerárquicas
- BD de Red
- BD Relacionales (RDBMS) ← El del curso
- BD Orientadas a Objetos (OOBMS)
- BD Multi-dimensional
- BD XML, JSON
- BD NoSQL

1.5] Redundancia, consistencia e integridad

- Redundancia
 - Ocurre cuando un mismo dato se almacena más de una vez de manera innecesaria en distintos lugares
 - Puede provocar inconsistencias
 - * Existe "redundancia" necesaria
 - ↳ En el modelo relational se emplea para relacionar entidades
 - La redundancia no necesariamente es mala → Ayuda a mejorar el desempeño

- Inconsistencia
 - Ocurre cuando 2 o más versiones de un mismo dato cuentan con valores diferentes (Versiones en conflicto)
 - A diferencia de Redundancia, una condición de inconsistencia NO debería existir.

- Integridad
 - Condición en la que los datos se consideran correctos, verídicos y sin inconsistencias.

Ejemplo → Checar tabla (Imagen)

Columnas Redundantes	- Nombre del Admin - Teléfono - Dirección	- Presupuesto
----------------------	---	---------------

II Eliminación de Redundancia

- Admin
 - nombre
 - teléfono
- Proyecto
 - código
 - presupuesto

nombre	código
teléfono	presupuesto
dirección	admin_id
admin_id	

- Cada proyecto tiene un admin_id

Proyectos de escritorio

PROYECTO

coo	preso	admin_id
500	\$5	3
501	\$10	4

1.6] Sistemas de Gestión de datos

Datos:

- Formado por un conjunto, por un conjunto de elementos que permiten implementar y administrar los datos de una empresa, organización, etc.

Elementos

SOFTWARE	HARDWARE
- S.O - Admin Tools	- Servidores
- DBMS	- Componentes red

PERSONAL

Nombre del rol	Descripción	Principales habilidades.
Desarrollador / Programador de bases de datos.	Crea y mantiene aplicaciones desarrolladas con herramientas, extensiones y lenguajes de programación de bases de datos.	Programación SQL y extensiones PL/SQL, etc.
Analista de bases de datos.	Generación de los artefactos necesarios para extraer y comprender los requerimientos tanto funcionales como no funcionales de un usuario o cliente final.	Facilidad de interacción con el usuario final, capacidad de conciliar y proponer estrategias y alternativas de solución al usuario final. En algunos casos representa el medio de comunicación entre el equipo de desarrollo y el usuario final.
Responsable de la base de datos.	A partir de un caso de estudio o problema real, generar diferentes modelos de datos que describen la estructura y el comportamiento de los datos tratando de conservar un equilibrio entre desempeño, niveles de normalización, etc.	Diseño de modelos de datos. Conceptos sólidos de Programación orientada a objetos, Ingeniería de software, SQL.
Administrador de la base de datos (DBA)	Encargado de mantener la buena salud del DBMS para que este siempre opere de la forma más óptima posible.	Conocimientos sólidos en arquitectura y algoritmos internos de la base de datos, estructuras físicas y lógicas tanto de almacenamiento como de memoria y procesos, además de contar con fundamentos sólidos en áreas como: Sistemas operativos, redes, diseño y programación de sistemas de software, seguridad, etc. Tarea: Leer Artículo Responsabilidades del DBA en tiempos 2.0. Obtener el artículo de: • http://thegrouperblog.com/2009/01/20/dba-responsibilities-in-the-new-world/
Especialista en análisis de datos.	Sus conocimientos le permiten aplicar tecnologías, técnicas, cálculos matemáticos y estadísticos, así como el uso de una diversidad de herramientas enfocadas al análisis de datos de grandes volúmenes. Data warehouse, Big Data, Data Science	Conocimientos sólidos en las áreas de estadística, inteligencia artificial, así como el uso de diversas herramientas y tecnologías enfocadas al análisis de datos. Este análisis permite descubrir conocimiento, y con ello realizar la toma de decisiones importantes para una empresa u organización.
Arquitecto de bases de datos.	Encargado de definir la mejor solución para un problema en particular en términos de: Infraestructura (servidores, medios de almacenamiento, respaldo y recuperación, redes, etc.), tipo de DBMS a emplear. En general el arquitecto se enfoca a proporcionar la mejor solución haciendo un amplio énfasis en los requerimientos funcionales, claro sin dejar de considerar los no funcionales.	Generalmente son personas con una amplia trayectoria y extensa experiencia en proporcionar soluciones de todo tipo y tamaño, de pequeños sistemas web hasta sistemas de milón critica.
Consultor de bases de datos	Ofrece asesoramiento y consultoría a empresas en cuanto al uso de tecnologías de bases de datos con la finalidad de mejorar sus procesos de negocio y alcanzar metas específicas.	Todas las habilidades mencionadas anteriormente.

* Funcional → No puede posar de 60 créditos / serialización // Reglas

No funcional → Requisitos de calidad, puede operar pero no de la mejor forma

No funcional → Requisitos de calidad, puede operar pero no de la mejor forma

- Id / Contraseña al ingresar
- Password cifrado
- Mantenible

* Sistemas de misión crítica → Sistemas de 24 horas donde sus consecuencias críticas danan a las personas, economía, etc.

1.7 Metodología de Diseño de Bases de Datos



- 1) Requerimiento funcionales y no funcionales
- 2) Modelo entidad-relación
- 3) Modelo relacional
- 4) Modelos particulares (Esquema Físico)

Diseño Conceptual

- Homóloga visión de datos
- Independiente al modelo de datos
- Generalmente se emplea notación CHEN

Diseño Lógico

- Depende del tipo de datos (Veremos modelo relacional)
- Independiente del RDBM's a emplear
- Notación
 - ① Crow's Foot
 - ② IDEF1X

Diseño Físico

- Dependiente del RDBM's
- Muestra la forma en la que los datos se estructuran físicamente, se incorporan estructuras de almacenamiento // Discos, particiones, Tablespace, etc.

2 Modelos

Friday, 31 January 2020 8:16 AM

2.1 Modelado de Datos

- Actividad fundamental en el proceso de construcción de una BD
- Permite construir modelos de datos que homologan la forma de visualizar datos aplicados a un problema real como "Problema de Dominio"

2.2] Modelo de Dato

- Representación generalmente gráfica que describe la estructura y características de los datos
- Representa una abstracción real
- Forma de comunicación entre involucrados (Diseño Conceptual)
- Elementos de un modelo de datos

1] Entidades

- Cualquier objeto de interés para un problema o caso de estudio apartir del cual se extraen datos

- Tipos de ① Físicas : Se pueden tocar
objetos

② Abstractas : No se pueden tener físicamente

- Se representan en **sustantivos en singular**

- Puede ser definida en términos de atributos

- Instancia de una entidad

↳ Se obtiene cuando los atributos de la entidad adquieren un valor

de la entidad adquieran un valor para un objeto en particular

EJEMPLO

- Viaje (Duración, tipo de transporte, origen, destino ...)
- Instancia → Viaje (3 días, camión, UAM, PULI)

2) Atributos

- Características de una entidad
- Definen un conjunto de atributos que definen una entidad
- Se representan mediante sustantivos

→ d Cómo distinguir? (Ejemplo)

- Alumno → Ent
- Clase → Ent
- Horario →
- Calificación → A
- Créditos → A
- Asignatura →
- Idioma → j?

• Una entidad debe tener al menos 2 atributos

SIN EMBARGO

- Hay casos donde es posible tener entidades con un atributo
 - ↳ Debido a si es frecuente
 - * Más el ID

NOTA

- Crear una entidad con el atributo y además un identificador

Idioma ('ID', 'Nombre')

↑
Se va a repetir pero será una redundancia necesaria

① Si el atributo aparece o se asocia con otras entidades de forma frecuente
↳ Redundancia

② Si el valor del atributo puede cambiar
↳ Inconsistencia

③ Su valor son cadenas relativamente largas

- Checar duplicados (Sinónimos)
- Cardinalidad única (Detectar el número de instancias de la entidad ∵ Si sólo es una, se descarta)
- Ambigüedades
 - ↳ Detectar palabras que no representan claramente a un objeto de interés
 - ↳ Ejemplo:
 - Catálogo // No es específico
 - Validar // No hacer uso de verbos

- Roles: En algunos casos una entidad puede contar con roles
- Ejemplo | Empleado, Investigador, Profesor, director, Secretario



(Descartar o conservar roles?)

- Si los roles cuentan con atributos particulares se conservan
- Se descartan si tienen atributos particulares

2. Relación-entidad | Ejercicio

Tuesday, 4 February 2020 7:12 AM

► Relaciones entre Entidades

- Un modelo de datos, prácticamente todas las entidades se relacionan con al menos una entidad

► Tipos de relaciones

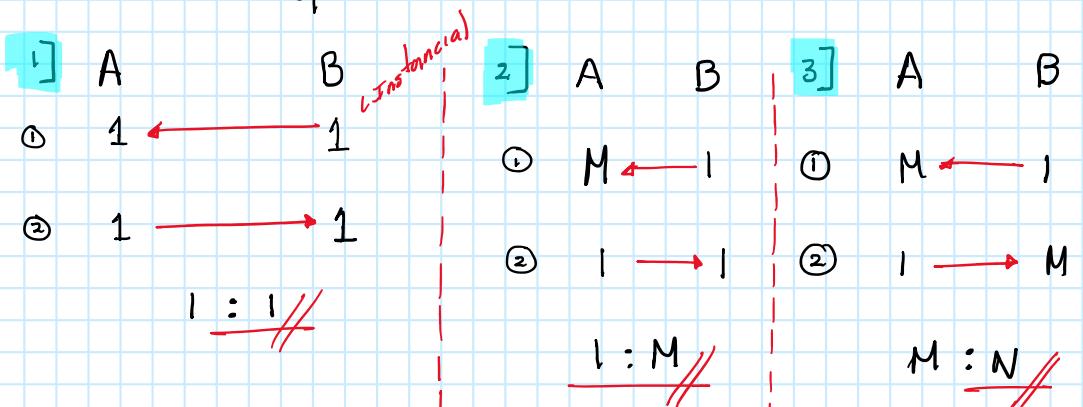
- ① Relación uno a uno (One to one) ; 1:1 . 1..1
- ② Relación uno a mucho (One to many) ; 1:M , 1..*
- ③ Relación muchos a muchos (Many to many) ; M:N , *..*

→ ¿Cómo identificar el tipo de Relación entre 2 entidades?

- Para un par de entidades A y B se realizan las siguientes validaciones:

- ① ¿Cuántas instancias de A se asocian con una de B?
- ② ¿Cuántas instancias de B se asocian con una de A?

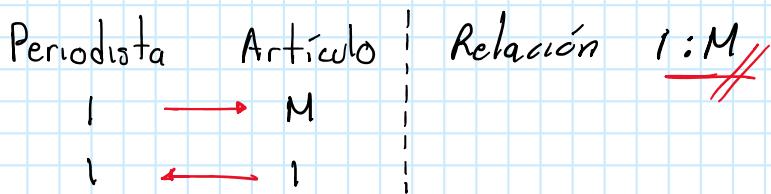
// Posibles respuestas



→ Ejemplos

Entidades: Periodista y artículo

- ① Un periodista escribe varios artículos
- ② Un artículo es escrito por un periodista



Entidades:

- ① Un alumno inscribe muchas materias 1:M
- ② Una materia es cursada por muchos alumnos 1:M

∴ Relación M:M //

Entidades:

- ① Una casa tiene una cocina 1:1 | | 1:1 //
- ② Una cocina se incluye en una casa 1:1 | |

Resumen de los
Modelos de datos

- ① Atributos
- ② Entidades
- ③ Relaciones
- ④ Restricciones

► Restricciones

- Condiciones que se aplican a una entidad o sus atributos
- Su objetivo es mantener la integridad de los datos
- Se identifican a través de las reglas de negocio del caso de estudio.
- Algunas casas son implementadas por el DBMS, otras requieren programación

2.3 Reglas de Negocio

- Son enunciados breves, concisos, sin ambigüedades

- Son enunciados breves, concisos, sin ambigüedades que describen una política, hecho, regla o principio con significado importante en un caso de estudio.
- Ordenar su correcto funcionamiento

Ejemplo

1. EJEMPLO – ELEMENTOS DE UN MODELO DE DATOS.

Considera el siguiente caso de estudio asociado con un sistema de renta de automóviles. Realice una lectura cuidadosa del caso de estudio, determinar los elementos del modelo de datos:

- A. Entidades
- B. Reglas de negocio
- C. Relaciones entre entidades

Caso de estudio, EU-Rent.

Modelo de negocio de EU-Rent:

EU-Rent es una compañía que se dedica a la renta de autos.

Sucursales:

La compañía cuenta con más de 1000 sucursales en ciudades del país. Las sucursales están distribuidas en 3 diferentes regiones: región A, región B y región C. Una región está integrada por varias ciudades, y en una ciudad pueden existir varias sucursales.

Personal de las sucursales:

Cada sucursal cuenta con un gerente y varios agentes de ventas encargados de atender a los clientes. Adicionalmente, cada sucursal cuenta con hasta 3 ingenieros mecánicos para dar mantenimiento a los autos. Los gerentes solo pueden administrar a una sucursal, aunque los agentes, de así solicitarlo pueden trabajar en diferentes sucursales. Los empleados no pueden realizar rentas de autos.

Autos:

Cada sucursal cuenta con un catálogo de autos para rentar. Un auto solo pertenece a una sucursal. Cada 3 meses o cada 10,000 Km, el auto debe recibir servicio de mantenimiento. El auto puede tener varios servicios a lo largo de su vida útil.

Clientes:

Se requiere almacenar los siguientes datos del cliente: nombre, apellido paterno, apellido materno, email, RFC, dirección y teléfono.

Rentas:

Un cliente puede rentar hasta 3 autos al mismo tiempo (máximo 3 por renta). Se registran los datos de la renta: fecha de solicitud, periodo de tiempo de la renta el cual puede ser hasta por un mes, el cliente que la solicita, y se asocian los autos que el cliente selecciona. Si el auto está en servicio, este no podrá ser rentado.

Facturas:

Cada vez que el cliente realice una renta, se genera una factura, se requiere almacenar los siguientes datos: monto total, renta asociada y la fecha de elaboración.

• Entidades

– Empleado	→ Agente Ventas	• No tienen atributos particulares ∴ Se generaliza como empleado
	→ Gerente	
	→ Ing. Mecánico	
– Servicio / Mantenimiento	• Están duplicados	
– Ciudad	• Al repetirse la ciudad comúnmente se mantiene como Entidad ∴ Puede generar inconsistencia al repetirse erróneamente	
– Región	• Sus posibles valores son "A", "B" y "C"	

- Región
 - Sus posibles valores son "A", "B" y "C" por lo que no hay posibilidad de inconsistencia
 - ∴ Atributo de la Ciudad

- Sucursal
 - Cliente
 - Auto
- Renta
 - Factura

- ~~X~~ Compañía
 - Se desartar al ser una sola instancia
- ~~X~~ Catálogo
 - Ambigüedad
- ~~X~~ Rol
 - Ambigüedad

- Reglas de negocio
 - No deben tener ambigüedades.

- Un cliente puede rentar hasta 3 autos al mismo tiempo
- Un empleado no puede rentar autos
- Una renta genera una factura
- Un auto recibe servicio de mantenimiento cada 3 meses o cada 10 km
- Existen 3 regiones, A, B y C
- Una sucursal cuenta con un sólo gerente

• Relaciones entre entidades

- Ciudad > 1:M
Sucursal > 1:1 $\Rightarrow \cancel{1:M}$
 - Sucursal > 1:1
Empleado > 1:1 $\Rightarrow \cancel{1:1}$
 - Factura > 1:1
Renta > 1:1 $\Rightarrow \cancel{1:1}$
 - Auto > 1:M
Servicio > 1:1 $\Rightarrow \cancel{1:M}$
 - Cliente > 1:M
Auto > 1:M $\Rightarrow \cancel{1:M}$
 - Cliente > 1:1
Renta > 1:M $\Rightarrow \cancel{1:M}$
 - Auto > 1:M
Renta > 1:M $\Rightarrow \cancel{M:M}$
- * Una sola relación hasta

■ Renta > 1:M
Servicio > 1:1 => 1:M //

Renta' 1:M //

* Una renta incluye hasta
3 autos

• Sucursal > 1:M
Auto > 1:1 => 1:M //

• Sucursal > 1:M => M:M //
Empleado > 1:M
(Agente)

• Sucursal > 1:M
Empleado > 1:1 => 1:M //

2. Evolución Modelos

Thursday, 6 February 2020 7:07 AM

- Los conceptos vistos hasta ahora son independientes al modelo de datos
- La siguiente tabla muestra su evolución a lo largo del tiempo

Tiempo	Nombre
$50 < t < 70$	Sistemas de Archivos
$50 < t < 80$	Modelo jerárquico
$60 < t < 90$	Modelo de RED (RDBMS)
$70 < t$	Modelo Relacional
$75 < t$	Modelo Entidad-Relación
$80 < t$	Modelo Orientado a Objetos (OODBMS)
$90 < t$	Modelo Objeto-Relacional <ul style="list-style-type: none">- ORM / Object Relational Mapping- Bases Datos XML / JSON
	Modelo "Bases de datos Multidimensionales"
$2000 < t$	Modelo NoSQL

1] Sistemas de Archivos

Históricamente no existían sistemas, dichas tareas se realizaban de forma manual. Los datos de las empresas se escribían en papel y se almacenaban en folder o carpetas almacenadas en grandes estantes. Esta estrategia se vuelve problemática conforme la empresa crece y por lo tanto la cantidad de datos a mantener. Por lo anterior se opta por hacer uso de tecnologías computacionales.

Almacenamiento empleando Sistemas Computacionales.

C_NAME	C_PHONE	C_ADDRESS	C_ZIP	A_NAME	A_PHONE	TP	AMT	REN
Alfred A. Rames	615-844-2573	218 Park Rd., Bates, TN	36123	Leah P. Hahn	615-882-1244	T1	100.00	05-Apr-2014
Leona K. Dunn	713-594-1238	Box 12A, Fox, KY	25246	Alex B. Alby	713-226-1249	T1	250.00	16-Jun-2014
Kathy W. Smith	615-894-2385	125 Old Ln., Bates, TN	36123	Leah P. Hahn	615-882-2144	S2	150.00	29-Jan-2015
Paul F. Olivenski	615-894-2180	217 Lee Ln., Bates, TN	36123	Leah P. Hahn	615-882-1244	S1	300.00	14-Oct-2014
Myron Orlando	615-222-1672	Box 111, New, TN	36155	Alex B. Alby	713-226-1249	T1	100.00	26-Dec-2014
Amy B. O'Brien	713-442-3381	387 Troll Dr., Fox, KY	25246	John T. Olson	615-123-5588	T2	550.00	22-Sep-2014
James G. Brown	615-287-1228	21 Eye Rd., Nash, TN	37118	Leah P. Hahn	615-882-1244	S1	120.00	25-Mar-2015
George Villanz	615-290-2556	155 Main, Nash, TN	37119	John T. Olson	615-123-5588	S1	250.00	17-Jul-2014
Anne G. Fariss	713-362-7185	2119 Elm, Creek, KY	25432	Alex B. Alby	713-226-1249	T2	100.00	03-Dec-2014
Oletha K. Smith	615-297-3809	2782 Main, Nash, TN	37118	John T. Olson	615-123-5588	S2	500.00	14-Mar-2015

Ing. Jorge A. Rodríguez Campos

jorgerdic@gmail.com

Página 1

Material de apoyo.

FI-UNAM

- La generación de reportes empleando sistemas de archivos manuales como los explicados anteriormente tomaba semanas en desarrollarlos.
- Surge entonces un nuevo rol llamado "Especialista de procesamiento de datos" (DP) encargado de crear un sistema basado en computadora encargado de almacenar los archivos en la computadora y generar los reportes.
 - La figura anterior muestra un ejemplo de un archivo almacenado en un sistema de cómputo.
 - Se establecía un lenguaje de traducción para entender el contenido de cada columna. Por ejemplo:
 - C_NAME = Nombre del cliente.
 - TP = Tipo de seguro, etc.
- Cuando el usuario final requiere datos de dichos archivos, este se los solicita al DP.
- Con base a los requerimientos del reporte solicitado, el DP crea una serie de programas para obtener los datos de los archivos, manipularlos y presentarlos en el formato solicitado.
- Si el reporte se vuelve a solicitar, el DP reutiliza los programas, pero si el usuario cambia requerimientos solicitando formatos más complejos, el DP deberá crear nuevas versiones de sus programas.
- Estos programas definen todas las características y estructura de los archivos, en un principio esta técnica puede ser buena por la rapidez de los programas al estar hechos totalmente "a la medida" con respecto a las características de un archivo.
- En general, el uso de sistemas de cómputo para almacenar archivos de datos representó un avance significativo, en especial por la decisión de incorporar el uso de tecnologías computacionales para procesar datos, pero el principal problema fue que no se disponía de las herramientas necesarias para convertir los datos en la información que un usuario final necesitaba.
- Lo anterior puede traer diversos inconvenientes como los siguientes:
 - Dificultad para crear nuevos programas los cuales deben adaptarse a archivos y programas existentes, su ejecución se puede volver lenta.
 - Al crear nuevos archivos, fácilmente puede generar duplicación de datos sobre todo al momento de relacionarlos y como consecuencia,

incorporar el uso de tecnologías computacionales para procesar datos, pero el principal problema fue que no se disponía de las herramientas necesarias para convertir los datos en la información que un usuario final necesitaba.

- Lo anterior puede traer diversos inconvenientes como los siguientes:
 - Dificultad para crear nuevos programas los cuales deben adaptarse a archivos y programas existentes, su ejecución se puede volver lenta.
 - Al crear nuevos archivos, fácilmente puede generar duplicación de datos sobre todo al momento de relacionarlos y como consecuencia, el riesgo de inconsistencias.
 - Al existir varias aplicaciones o programas que manejan sus propios archivos, fácilmente pueden caer en una situación de duplicidad al ser dichas aplicaciones independientes entre sí.
 - Control deficiente de datos. No existe un control centralizado de los datos, un mismo dato podría aparecer en diversos archivos. Por ejemplo, el nombre de un empleado podría aparecer en los archivos del departamento de RH, en el de ventas, o en el de finanzas.
 - Dificultad para acceder a los datos. Cada consulta de datos o búsquedas requiere la creación de más programas encargados de realizar la búsqueda muy particular. Una búsqueda puede implicar procesar varios archivos que pudieran tener diferente formato y estructura lo que complica su lectura.
 - Dificultad para administrar los archivos tan pronto como su cantidad aumenta considerablemente, adicional a las operaciones de mantenimiento que requieren: inserciones, actualizaciones, eliminación de registros.
 - Falta de medidas de seguridad y control de accesos, en especial con datos sensibles y con datos que deben ser compartidos inclusive con usuarios geográficamente dispersos.
- Las bases de datos permiten en buena medida que los "programas" sean independientes a la definición de la estructura de los datos, esto debido principalmente a la capacidad de proporcionar una fuente común y centralizada de los datos, así como el soporte de lenguajes encargados de realizar el acceso y manipulación de los datos.

2] Modelo Jerárquico

Modelo de datos Jerárquico.

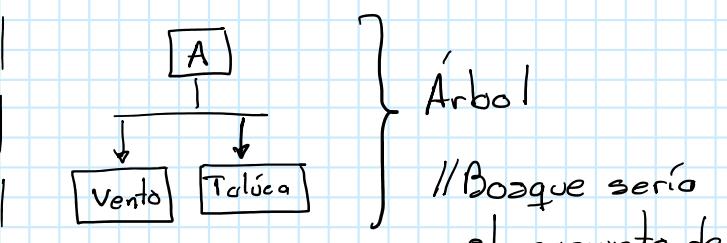
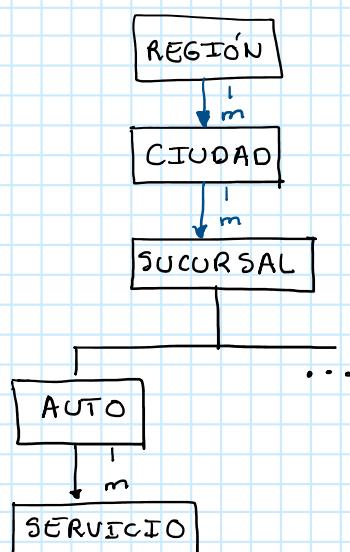
- Desarrollado en la década de los 60s para administrar grandes cantidades de datos para proyectos complejos como fue el proyecto del Cohete Apollo Rocket que aterrizó en la luna en 1969.
- Representa el primer modelo de datos empleado en un DBMS comercial formado por una estructura de árbol.
- No existen estándares ni formalizaciones del modelo jerárquico como en el caso del relacional.
- Sistema principal que implementa a este modelo es IMS (Information Management System) de IBM el cual emplea un lenguaje de datos DL/I
- El modelo hace uso principalmente de punteros y fue desarrollado especialmente para representar relaciones 1:M. Se emplea una estructura PADRE/HIJO donde el nodo PADRE puede tener a varios nodos HIJO, mientras que un nodo HIJO solo puede tener a un nodo PADRE.
- A las entidades se les conoce como SEGMENTOS, a sus instancias se les conoce como OCURRENCIAS, y a los atributos como CAMPOS.
- Solo permite representar relaciones 1:1, 1:M
- Una vez creada la estructura jerárquica, está ya no se puede modificar. Si se desean aplicar cambios, se debe eliminar

La liga entre nodos determina el camino único de acceso a los datos llamado CAMINO SECUENCIA JERÁRQUICA.
Se realiza un recorrido arriba, abajo, de izquierda a derecha, adelante y atrás.



① Ejemplo

Entidades



Problemas

- Desempeño en búsquedas disminuye respecto a la profundidad
- Redundancia con relación M:N debido a que no era soportado
 - ↳ No podía tener más de un nodo parente, sin embargo,

→ No podía tener más de un nodo padre, sin embargo, redundancia de nodos

- Dificultad para eliminar datos

3] Modelos de Red

Modelo de Red.

- Creado para representar relaciones complejas de una forma más efectiva con respecto al modelo jerárquico principalmente para mejorar desempeño.
- Similar al jerárquico, las entidades se representan como nodos y sus relaciones son las líneas que los unen.
- El usuario final percibe a los datos como una colección de relaciones 1:M
- Un registro puede tener **varios** padres.
- Ejemplos de este modelo: CODASYL fue desarrollado en 1971 por un grupo conocido como CODASYL (Conference on Data System Languages, Data Base Task Group).

Conceptos del modelo de red.

- Las relaciones entre registros deben ser descompuestas en un conjunto de listas.
- Cada lista está formada por:
 - Una única instancia de la entidad padre llamada Dueño.
 - Un conjunto de instancias hijas, todas ellas asociadas a la instancia padre llamadas miembros.
- Un mismo registro puede ser **dueño** de varias listas.
- Cada lista debe tener un nombre asignado.



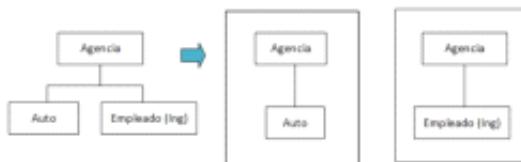
CARACTÉRÍSTICAS

- Tiene ventajas sobre el modelo jerárquico
- Permite relaciones M:N

X Programación complicada

Ejemplo:

- En una sucursal pueden existir varios autos, en una sucursal pueden laborar hasta 3 Ing. Mecánicos [empleados].
- El modelo jerárquico para estas 2 reglas es el siguiente:
- La descomposición en 2 nuevas listas se muestra a continuación:
 - Lista 1: Inventario. Corresponde a la lista de autos que pertenecen a una Agencia en particular.
 - Lista 2: Mecánicos. Conjunto de Ingenieros que trabajan en una Agencia en particular.



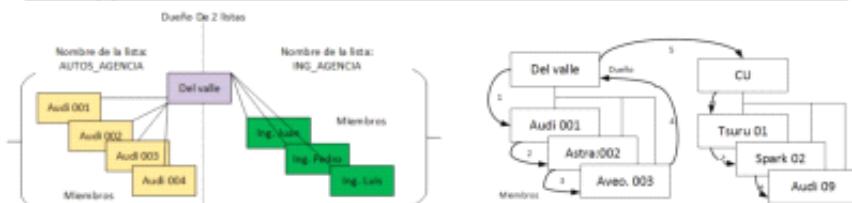
- En la figura anterior se tiene que una instancia de Agencia llamada Dueño (Owner) se asocie con un conjunto de Autos llamados Miembros. Lo mismo ocurre para los Ing. Mecánicos.

Ing. Jorge A. Rodríguez Campos

jorgeardc@gmail.com

Página 3

Material de apoyo.



- En la figura del lado izquierdo se muestra un ejemplo de las instancias de las entidades Auto y Agencia con sus respectivas relaciones.
- El usuario del BD podría navegar entre estos 2 conjuntos a través del lenguaje DML (Data Manipulation Language) proporcionado por la implementación de este tipo de DBMS. Empleando para ello punteros que indican el registro actual que se está accediendo (figura del lado derecho).

4] Modelo Relacional

Siguiente tema...

5] Modelo Entidad - Relacional

- Complemento al modelo relacional, dedicado para ser **conceptual**, dicho de otra forma, se dedicó a una visión menos "técnica".

6] Modelo Orientado a Objetos

Bases de datos orientadas a objetos (OODBMS).

- En estos manejadores ya no se requiere realizar un mapeo entre el modelo de objetos y el modelo relacional.
- Los objetos de una aplicación se guardan como tal en la base de datos.
- **No existen tablas, ni SQL relacional.**
- Existe un **solo modelo**, que en este caso es el modelo de objetos.
- Implementaciones:
 - Versant Object Database (antes DB4O) <http://www.actian.com/products/operational-databases/versant/>
 - ObjectDatabase++ (ODBPP) <http://www.ekysoftware.com/ODBPP>
 - ObjectDB <http://www.objectdb.com/>
 - Objectivity/DB <http://www.objectivity.com/products/objectivitydb/>

Tarea:
Versant Object Database

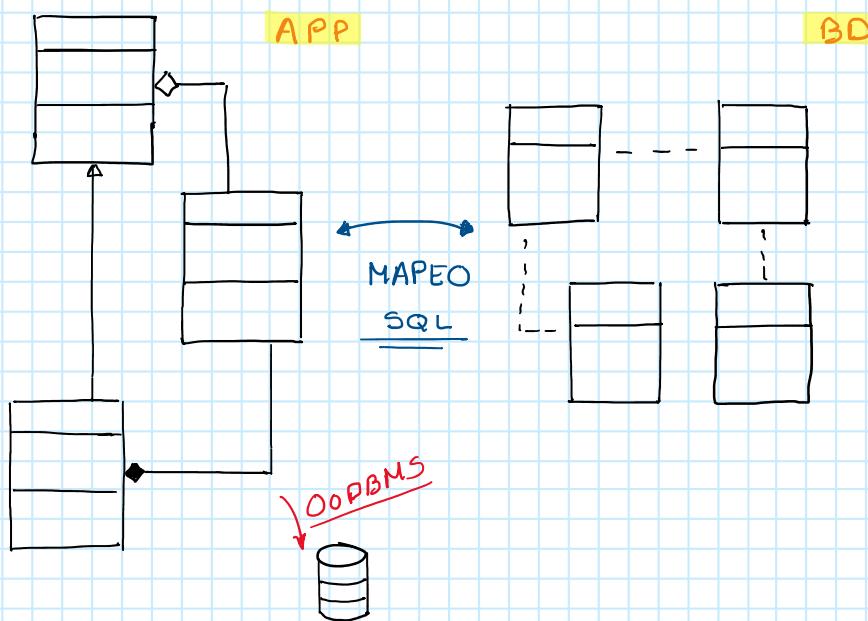
Ejemplo:

Guardando objetos en Versant Object Database (antes DB4O):

```
Piloto piloto = new Piloto("Schumacher", 100);
db.store(piloto);
System.out.println("Stored " + piloto);
```

Recuperando objetos:

```
Piloto pilotoEjemplo = new Piloto("Juan", 2);
ObjectSet result =
db.queryByExample(pilotoEjemplo);
listResult(result);
```



7] Modelo Objeto - Relacional

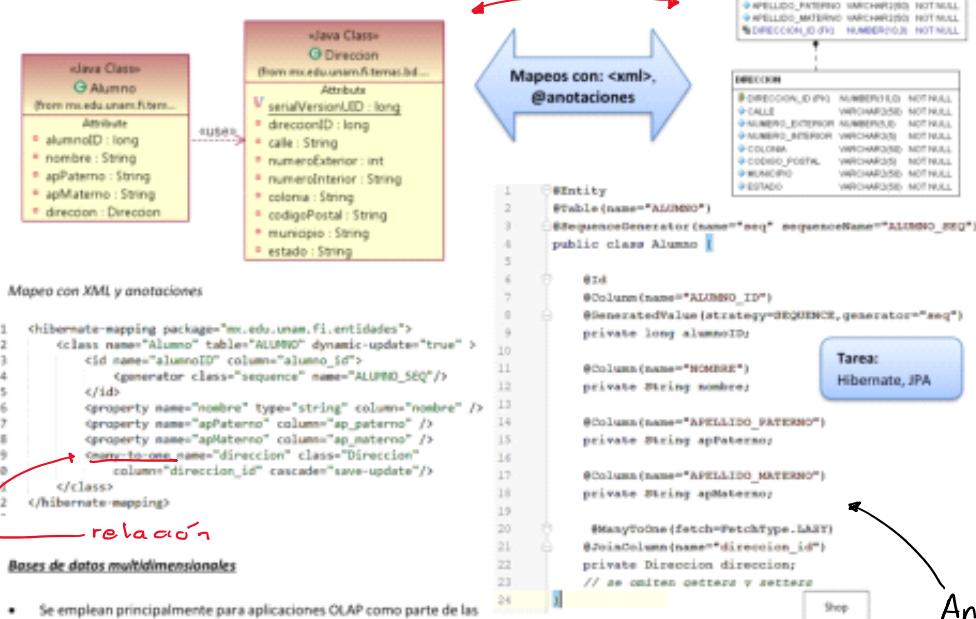
Modelos Objeto - Relacional (ORDBMS).

- Representan una extensión de un RDBMS agregando funcionalidades asociadas con conceptos de la programación orientada a objetos (POO).
- Tipos de datos personalizados: `create type Persona (nombre varchar(20), dirección varchar(20))`
- Herencia entre tablas (no todos los manejadores la implementan)
`create type Estudiante under Persona (curso varchar(20), departamento varchar(20))`
- Funciones definidas por el usuario.
`create function myFunction(p1,P2) BEGIN ... END;` *extend*
- En un ORDBMS todos los elementos de la base de datos son considerados como objetos con sus 2 elementos similar a un objeto en POO
 - **Características -> atributos.**
 - **Comportamiento -> métodos.**
- Una tabla, registro, columna, vista, índice, usuario, todos son considerados como objetos.

* Actualmente se usa mucha en los sistemas clásicos

8] Modelo → Object Relational Mapping

- Para poder sincronizar el estado de un conjunto de **objetos** en memoria (valores de sus atributos) con los valores de las columnas de un conjunto de **tablas** en el modelo relacional se requiere programar y ejecutar sentencias SQL que permitan comunicar a ambos modelos: PDO y RDBMS.
- Un ORM permite realizar esta actividad sin la necesidad de realizar programación de sentencias SQL **específicos ya sea en XML o por anotaciones**.
- Ejemplos de implementación de ORMs: Hibernate, JPA.



* Se suele utilizar todavía, JPA → Oracle

9] Base de datos XML

Bases de datos XML

- Son RDBMs con extensiones para almacenar documentos XML en columnas con tipo de dato XML.
- La gran mayoría de los manejadores actuales soportan tipos de datos XML.

Para realizar operaciones sobre los datos que contiene el documento XML se emplea el estándar SQL/XML y **herramientas como XPath, XQuery**.

Ejemplo:

```

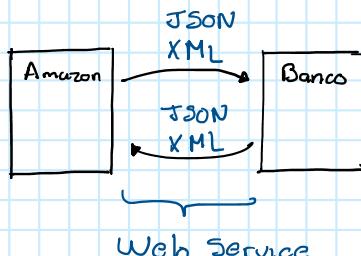
CREATE TABLE empleado (
    empleado_id NUMBER(3),
    datos_empleado XMLTYPE
);

INSERT INTO empleado (empleado_id, datos_empleado)
VALUES (1, XMLTYPE(
<empleado>
    <nombre>Daniel</nombre>
    <fecha_nacimiento>12/1/51</fecha_nacimiento>
    <email>damorgan@u.washington.edu</email>
</empleado>)
);

SELECT empleado_id, XMLQuery(
    'for $i in /empleado
    where $i /nombre = "Daniel"
    order by $i/nombre
    return $i/nombre'
    passing by value datos_empleado
    RETURNING CONTENT) XMLData
    FROM datos_empleado;

```

Modelos NoSQL.



- XML → Información más delicada
- JSON → Aplicaciones móviles

16] BD Multidimensionales

- Tiene que ver con analítica de datos
 - ↳ Sistemas OLAP → Online Analytical Processing
 - ↳ Bases estáticas

Bases de datos multidimensionales

- Se emplean principalmente para aplicaciones OLAP como parte de las herramientas empleadas en el área de Business Intelligence.
- Procesan una gran cantidad y volumen de información con la finalidad de realizar análisis.
- Los resultados son vitales para realizar la toma de decisiones.
- Las dimensiones del BD multidimensional se implementan a través del concepto de Cubo OLAP.
- A partir de la construcción del cubo se generan tablas (generalmente cantidades mayores de columnas) con las que se realiza el análisis y consultas de forma rápida y eficiente.
- Los datos de la tabla resultante ya no se pueden modificar. Se debe regenerar o rediseñar el cubo.

Ejemplo:

Una empresa podría analizar algunos datos financieros por producto, por periodo, por ciudad, etc.

- Los parámetros en función de los cuales se realiza el análisis de datos se les conoce como **dimensiones**.

- Cada una de las dimensiones está formada por una **Jerarquía de datos**.

Dimension: (Por Periodo, Por Producto)

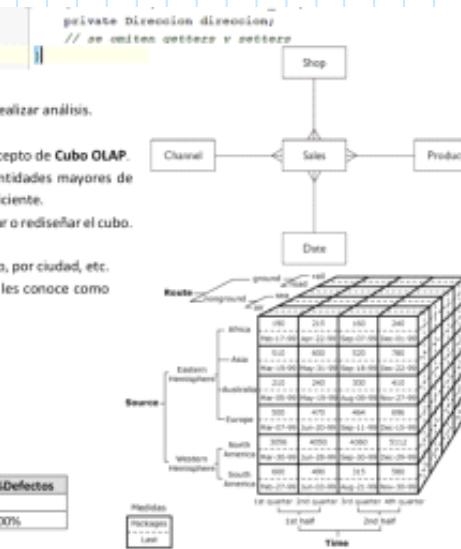
Jerarquía: (Año->Semestre->Mes->Semanal), (Categoría->Línea->Marca)

Hechos: (Ventas, Inventario, Defectos, Devoluciones)

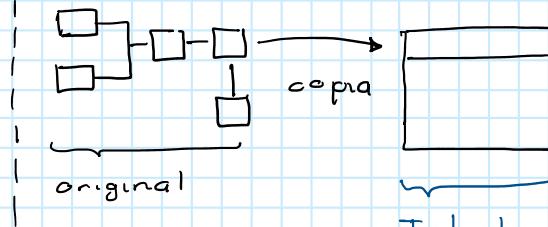
Métricas: (PD:=Defectos/Ventas, %Defectos)

Tabla resultante:

Tiempo	Productos	Ventas	Inventario	Defectos	Devoluciones	%Defectos
2006	Todos	1000	200	50	1/100	5
Enero06	Laptop	10	100	10	10/10	100%



Analítico → Muchísimas consultas



Toda la información necesaria

• Deposito de } • Minería de } **Datos** **Datos**

17] Modelos SQL

Requerimientos

- Incrementos de volúmenes
- Incrementos en costos
- Escalabilidad
 - Escalamiento vertical
 - Escalamiento horizontal

Problemas con las relaciones

- Solo un servidor
 - ↳ Se incrementa su capacidad (Costoso)
- Muchas pc pequeñas
 - ↳ cluster

Surgen como respuesta para implementar nuevos requerimientos de una forma más eficiente y menos costosa de lo que un RDBMS puede normalmente realizar.

- Incremento considerable del volumen de información a almacenar: Terabytes por día o por horas.
- Facilidad para consultar grandes cantidades de datos de una forma simple y eficiente.
- Imaginar, analizar cada clic y actividad que realiza un cliente en un sitio web altamente concursado para presentarle ofertas con base a sus gustos.
- Necesidad de distribuir el procesamiento a través de clusters para mejorar el desempeño (escalamiento horizontal).

Ejemplo:

Modelo de datos en Cassandra

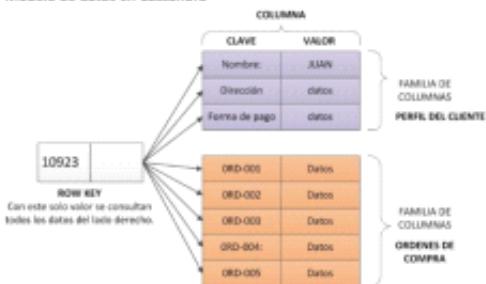
columna

Ejemplos de implementaciones NoSQL

- Necesidad de distribuir el procesamiento a través de clusters para mejorar el desempeño (escalamiento horizontal).

Ejemplo:

Modelo de datos en Cassandra



Modelo de datos en MongoDB: Formato JSON

SON

Ejemplos de implementaciones NoSQL

- Orientadas al manejo de documentos:

- CouchDB (de apache).
 - MongoDB (de empresa llamada 10gen).
 - SimpleDB (de amazon).
 - IBM Lotus Domino
 - Terrastore

- Orientadas al manejo de clave/valor

- Cassandra (de apache)
 - BigTable (de google)
 - Dynamo (de amazon)
 - Project Voldemort (de LinkedIn)

- Orientadas al manejo de Grafos

- Neo4J
 - Allegro
 - Virtuoso

DATOS INTERESANTE

- Big Table → 6 petabytes
 - Cassandra → Almacena 50 GB de datos en 0.12 [ms]

```
1 { "continente": {
2   "nombre": "América",
3   "habitado": "SI",
4   "paisenes": [
5     "país": [
6       {"nombre": "Costa Rica", "capital": "San José"},
7       {"nombre": "Méjico", "capital": "DF"},
8       {"nombre": "Argentina", "capital": "Buenos Aires"}
9     ]
10   ]
11 }
```

* 1:M relación
Entidad - Relacion

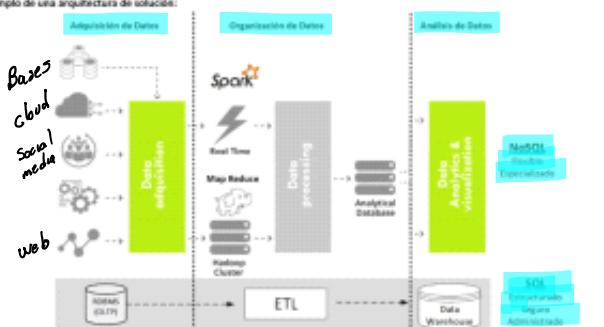
} Ejemplo de
MONGO DB
con JSON

EXTRA

BIG DATA

10

- Al hablar de este término, lo primero que se viene a la mente es pensar en **volumenes**, en gran cantidad de datos.
 - Volumenes grandes de datos no es la única característica de BigData. Existen 4 principales características que definen el término:
 - Volumen, Velocidad, Variedad y diversidad, Valor.
 - **Datos Estructurados**
 - Tienen un **relativamente bien definido** a través de un **esquema** [modelo relacional].
 - Datos tabulares generados comúnmente por sistemas transaccionales.
 - **Datos Semiestructurados**
 - Nuevas fuentes de información: **documentos de texto**, **aplicaciones estructuradas**, **videos**, **emails**, **audio**, **búsquedas**, etc.
 - Datos generados por interacciones a sesiones: **bitácoras**, **Call Detail Records** (Datos relativos a llamadas telefónicas).
 - Datos sociales: **Redes sociales**, **micro-blogging** (**Twitter**).
 - Elemento de una **concentración** de actividad:



- A nivel básico y en especial para realizar el análisis de datos estructurados, se requieren 3 principales etapas:
 - Adquisición e extracción de datos de diversas fuentes.
 - Organización y transformación de los datos para poder realizar el análisis.
 - Almacenamiento de los datos para su análisis en sistemas especializados.
 - En un escenario tradicional estas 3 etapas típicamente se tienen los siguientes componentes (parte inferior de la imagen):
 - Los datos se encuentran en un RDBMS, en un sistema OLAP, generalmente con niveles altos de normalización.
 - Se requiere de un proceso **ETL (Extractar - Transformar - Cargar)** principalmente para realizar la extracción de los datos del **RDBMS**, aplicar operaciones de transformación, limpieza o denormalización que permitirá la carga de los datos en otro sistema para facilitar y optimizar su análisis.

BIG DATA

“V’s”

- Volumen
 - Velocidad ← Tiempo real
 - Variedad
 - Valor (Streaming)

• Variedad

- Datos estructurados {• Son aquellos que tienen un esquema o patrón definido
 - Datos semi-estructurados {• Texto plano, correo
 - Datos no estructurados {• Imágenes, videos, audio

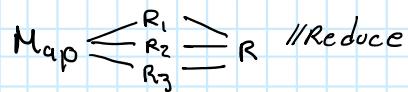
• Computer Hardware

- Spark
 - Hadoop // Framework para clustering (^{Hecho en} Java)
 - ↳ Núcleo → Cada nodo se, le

DISTRIBUIDO

MapReduce // Framework para clustering | Java |

↳ Mapo → Cada nodo se le asigna una tarea



Kafka // Procesamiento in Real-Time

- Colas de mensajes

HADOOP → Sistema de Archivos Distribuidos

↳ En cada máquina tengo mi sistema pero se puede replicar a posar de que esté en un nodo o pc doble tinto.

- Finalmente, los datos son almacenados en sistemas específicos o adecuados para análisis. Tipicamente un Data Warehouse, listos para ser analizados por diversas herramientas entre otras, minería de datos.
- ¿Qué sucede con los datos semi – estructurados o estructurados?
 - Para realizar análisis con este tipo de información se emplean conceptos de BigData. Ahora los datos pueden localizarse en bases de datos NoSQL principalmente por sus características de ser semi o no estructurados, altos volúmenes, etc.
 - El proceso ETL que tipicamente se realiza con datos estructurados se sustituye ahora por soluciones Map-Reduce, empleando frameworks y herramientas de procesamiento masivo y distribuido de datos como son: Hadoop, Spark, etc.

En el siguiente enlace se muestra como Oracle está implementando el concepto de Big Data así como los productos que ofrece: <http://www.oracle.com/technetwork/es/articles/database-performance/big-data-oracle-hadoop-2813760-es.html>

Ciencia de los datos.

Tarea:
Resumen/opinión

Una forma alternativa para realizar análisis de datos a las opciones vistas anteriormente es la llamada "Ciencia de los datos".

- La ciencia de los datos es un campo interdisciplinario que abarca el procesamiento y los sistemas para extraer conocimiento de datos que se pueden encontrar en varios formatos: Datos estructurados y datos no estructurados.
- En especial, los datos no estructurados representan la continuación de áreas de análisis de datos (Business Intelligence) como estadística, minería de datos, Análisis predictivo, Descubrimiento de conocimiento en bases de datos [Knowledge Discovery in Databases - KDD].
- Principales áreas de conocimiento participan en Ciencia de datos:
 - Ingeniería de Software
 - Expertos en el dominio del problema (dominio de las reglas de negocio y operación asociadas con los datos).
 - Investigación de operaciones
 - Matemáticas
 - Estadística.
- Las 2 principales categorías de un científico de datos:
 - Analistas / Expertos en estadísticas
 - Ingenieros.

Los siguientes artículos muestran un panorama básico sobre esta área.

<https://dzone.com/articles/an-introduction-to-data-science>

https://es.wikipedia.org/wiki/Ciencia_de_datos

Tarea:
Resumen/opinión

3. Modelo relacionado

Friday, 7 February 2020 8:29 AM

► 3.1] Características

- Introducido en 1970 por F. Codd
- Basado en 2 principios matemáticos
 - Lógica de predicados
 - Teoría de conjuntos
- El modelo define 3 principales componentes
 - Estructura lógica de los datos
 - Conjunto de reglas de integridad
 - Conjunto de operaciones que definen la forma de manipular datos
- Hace uso de 2 lenguajes
 - Álgebra relacional } Ambos definen
 - Cálculo relacional } operadores lógicos

3.2] Estructura lógica de los datos

- La estructura lógica en el modelo relacional se fundamenta en:
 - Relación ≈ tabla ≈ Entidad
 - Tupla ≈ Registro ≈ Renglón ≈ Instancia de una entidad
 - Atributo ≈ columna ≈ campo
 - Cardinalidad = Número de tuplas de una relación
 - Grado = Número de atributos de una relación
 - Llave
 - Primaria
 - Foranea

Llave →
- Foranea

- Dominio → Conjunto de posibles valores que puede adquirir un atributo

Ejemplo	Dominio de calificación	[5, 10], N.P.
	Dominio de salario	[120, 120000]

► TABLAS

- Representan a la unidad básica de almacenamiento
- Representa al concepto de entidad
- Su nombre debe ser único al menos a nivel de esquema
- Los datos de una columna deben ser del mismo tipo
 - Cada columna define a un dominio
 - Cada tabla puede contar con una llave primaria (PK)

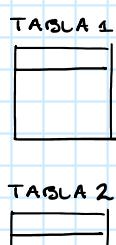
* Dato → Valor de una celda

↳ Cruce de una tupla y un atributo

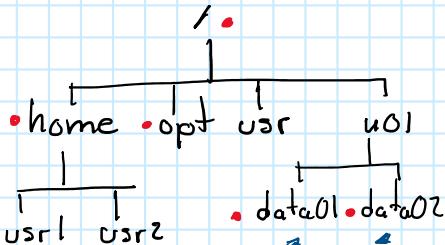
► Estructura Física de Almacenamiento de una tabla

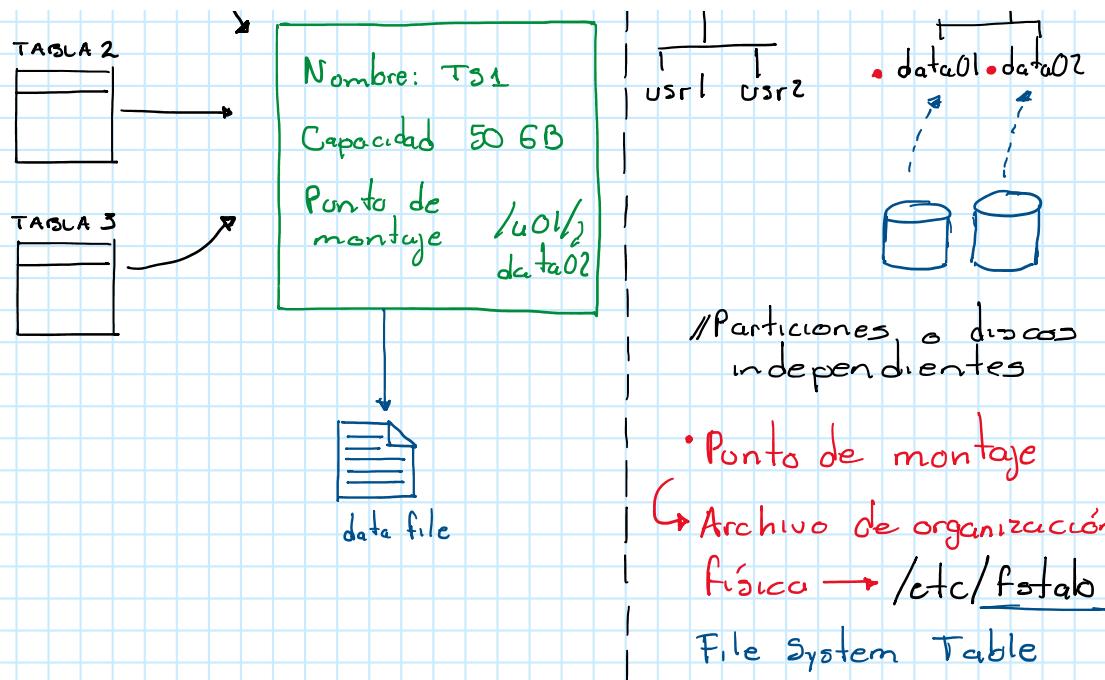
- Forma parte del diseño físico de una BD.
- Un ejemplo para Oracle:

CASO 1



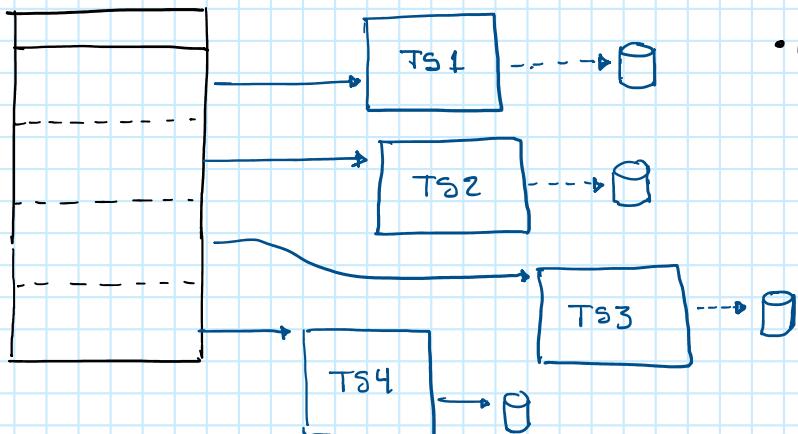
SISTEMA ARCHIVOS





CASO 2

TABLA 1



PARTICIÓN

- Cuando la tabla tiene muchísimos registros

VENTAJAS

```
select * from ciudadano
where nombre='Juan'
and edo = 'edo-mex';
```

- Busqueda rápida al ser sólo de una partición

```
select * from ciudadano
where nombre='Juan';
```

- Va a realizar búsquedas en paralelo

- ① Diseño
 - ② Esquema de indexado
 - ③ Particionamiento
 - ④ Hacer distribución física de los datos
- } Mejorar rendimiento de la BD

3.3] Restricciones (Constraints)

• Nombre de la restricción	• Implementación (Objetos del RDBMs)
- Restricción de llave primaria	→ Primary key (PK)
- Restricción de referencia	→ Foreign key (FK)
- Restricciones de integridad	→ <ul style="list-style-type: none"> - check - null, not null - unique
- Validación por trigger	→ trigger

► Restricción de llave primaria

- Es una columna o conjunto de valores cuyos valores identifican de manera única a todos los registros de una tabla.
- El RDBMS implementa este concepto mediante 2 restricciones:

- not null
- unique (Genera un índice único)

// Ejemplo

NUMERO_MATRICULA	NUMERO_MOTOR	MARCA	MODELO	AÑO	PAÍS_ORIGEN
CCA-341	91234908123	Toyota	Yaris	2005	Japón
OFG-851	53489787679	Fiat	Fiorino	2008	Tailandia
XTV-657	30752312386	Ford	Mustang	2011	México
WGB-959	50934187123	Toyota	Avensis	2010	Tailandia

Pueden ser 2 columnas
asociados al PK

¿Cuál es?

No son llaves primarias

Recuerda que puede tener
varias columnas pero sólo
es una llave

Aspectos para determinar cuál es la "PK":

① Dependencia Funcional

• Un atributo B es funcionalmente dependiente de A si cada valor de A determina uno y sólo un valor de B.

$$A \rightarrow B$$

• A nivel general se expresa:

$$\underbrace{A_1, A_2, \dots, A_n} \rightarrow \underbrace{B_1, B_2, \dots, B_n}$$

Atributos determinados Atributos dependientes

• De lo anterior, los atributos del lado izquierdo (Atributos determinantes) pueden formar a la PK de una tabla

EJEMPLO

Considerar a las columnas del catálogo auto C_1, C_2, \dots, C_6 , e indicar si las expresiones son True or False

NUMERO_MATRICULA	NUMERO_MOTOR	MARCA	MODELO	AÑO	PAÍS_ORIGEN
C ₁	C ₂	C ₃	C ₄	C ₅	C ₆

C ₁	C ₂	C ₃	C ₄	C ₅	C ₆
NUMERO_MATRICULA	NUMERO_MOTOR	MARCA	MODELO	AÑO	PAÍS_ORIGEN
CCA-341	91234908123	Toyota	Yaris	2005	Japón
OFG-851	53489787679	Fiat	Fiorino	2008	Tailandia
XTV-657	30752312386	Ford	Mustang	2011	México
WGB-959	50934187123	Toyota	Avensis	2010	Tailandia

- C₃ → C₄ // Falso
- C₄ → C₁ // Falso
- C₁ → C₄ // Verdadero
- C₂ → C₄ // Verdadero
- C₁ → C₂, C₃, ..., C₆ // Verdadero
- C₂ → C₁, C₃, ..., C₆ // Verdadero

// No es que salgan los datos sino que con eso podemos buscar

② Consideraciones requeridas

- Los valores de PK NO debería cambiar una vez asignado
 - No deben ser nulos
 - No deben duplicarse
- // Si puede cambiar el valor entonces no dejarla como llave primaria.

③ Consideraciones Opcionales

- Debido a que el DBMS genera un índice de tipo unique, en particular para validar duplicados, pueden existir algunos detalles de desempeño.
- El mejor desempeño para manejar este tipo de índice se obtiene:
 - 1) El índice se aplica a una sola columna
 - 2) El tipo de dato de la columna es numérico
 - 3) Los valores son enteros, preferentemente consecutivos

Números enteros

// Si ningún atributo cumple con estas

// Si ningún atributo cumple con estas consideraciones, se crea una llave primaria o PK llamada PK artificial o subrogado

④ Tipos de llaves primarias

Llaves primarias naturales

- Atributo que cumple con definición de llave primaria y tiene significado para las reglas de negocio del caso de estudio
// Por ejemplo mi número de cuenta

Llaves primarias candidatas

- Cumple con requisitos de PK pero simplemente no se seleccionó

Llaves primarias compuestas

- Formada por 2 o más atributos
- Lo anterior implica que la unicidad* de los registros se verifica con base a las combinaciones de valores de la PK

Suponer: num.motor num.matricula marca modelo

Redundancia e inconsistencia !!	PK	10	505505	Toyota	Yaris
		10	505506	Toyota	Yaris

Llaves primarias artificiales

- Es un PK que no cuenta con significado para las reglas de negocio
- Se usa para mejorar desempeño

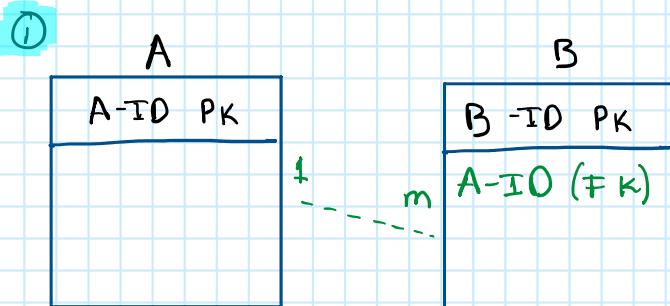
- | de uso para mejorar desempeño
- | - Convención para nombrar PK
 - | <NOMBRE_TABLA>_ID

3. Restricciones de referencia

Thursday, 13 February 2020 7:08 AM

- En un RDBMS se implementan a través del concepto de clave foránea (Foreign Key - FK)
- Una FK es un atributo o conjunto de atributos ubicados en una tabla B cuyos valores hacen referencia a registros de una tabla A
- Tipicamente a la tabla A se le conoce como tabla PADRE y a la tabla B como tabla HIJA
- Esta restricción se emplea para relacionar tablas o entidades
- El manejador verifica que los valores de la FK existen en la tabla PADRE.
- A nivel general, si un constraint no se cumple, el manejador genera un error de "Violación de restricción"

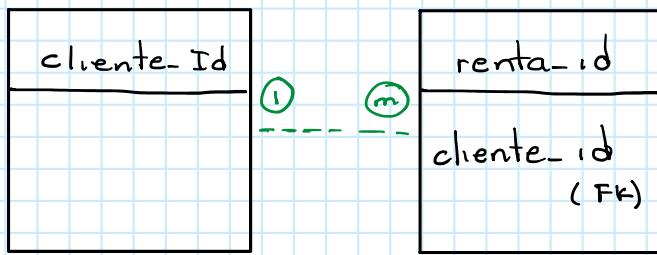
Ejemplos:



- Un registro de la tabla padre puede mandarte o asociarte a "n" de la tabla hija

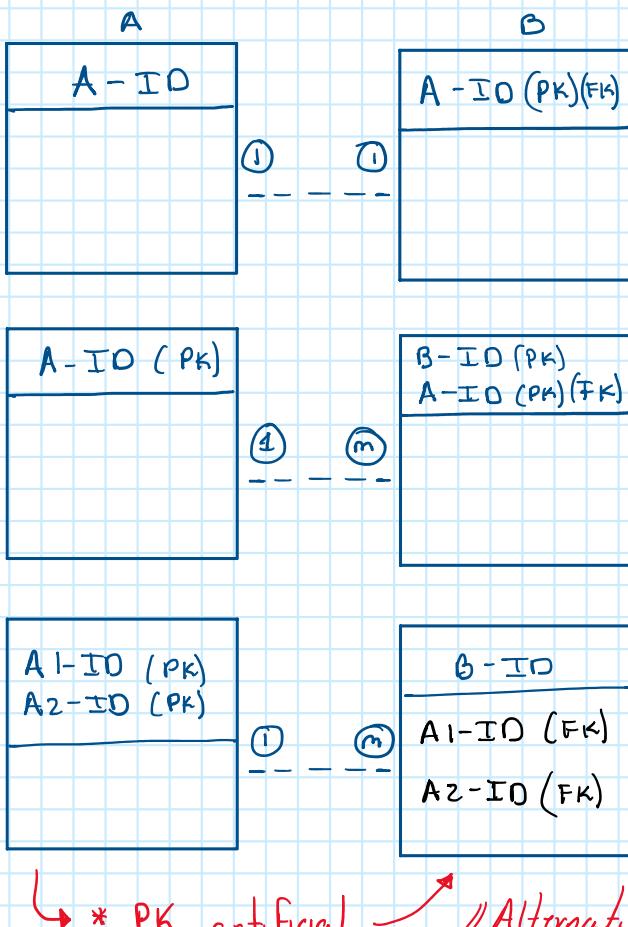
CLIENTE

RENTA



* Si no existe la PK no puede existir la FK en otra tabla

② Otra variante



* Hay que tratar de evitar este caso debido a que implica mayor trabajo

↳ * PK artificial ↳ //Alternativa

- Para que un atributo A pueda ser propagado como una FK, se debe cumplir:
 - A debe ser "PK"
 - A tiene una restricción **unique** y **not null**

► Restricciones de Integridad

→ NULL

→ NULL

- ¿Qué es un valor nulo?
 - La ausencia de valor para un atributo
 - Ojo: NULL no es
 - " ", null, '\0', "null"

Ejemplo

Obtener alumnos sin email | select * from alumno
| where email is null;
| not null

→ CHECK

- Permite asociar una expresión booleana a un atributo de una tabla que se evalúa al aplicar una inserción o actualización, si al evaluarla se obtiene:
 - true => sin error
 - false => Error de violación de restricción
- Este tipo de restricción tiene los siguientes limitantes.
 - La expresión booleana no puede hacer referencia a atributos de otras tablas
 - Generalmente para estos casos, se utiliza programación mediante un **trigger**

Ejemplo

Puesto

puesto_id
nombre
sueldo

→ (sueldo >= 8000
and
sueldo <= 10000)

→ UNIQUE

- Verifica que los valores de un atributo no se duplique

Pregunta
examen

- ¿Qué diferencia existe entre un constraint "unique" y uno "PK"?

- Que el constraint unique puede ser nulo

- Para implementar esta restricción el manejador crea un índice implícito

↳ La búsqueda para validaciones se hace mediante "índices"

→ Validación por Trigger

- Programa que se ejecuta de forma automática al ocurrir un evento en la BD.
- No se limitan como los "check"

3.4 Reglas de Codd

En 1985 Edgar Frank Codd publicó 12 reglas que en su conjunto definen las características que debe cumplir un sistema de bases de datos para que este sea considerado como relacional.

Regla 1: Información

Toda información contenida en una base de datos relacional debe ser representada de manera lógica como los valores de una columna contenidos en los registros de una tabla.

Regla 2: Acceso garantizado

Cada valor en una tabla está garantizado a ser accesible a través de la combinación nombre de la tabla, nombre de la llave primaria y nombre de la columna.

Regla 3: Tratamiento sistemático de valores nulos

Valores nulos deben ser tratados de manera sistemática independiente del tipo de dato.

Diccionario
METADATOS - de Datos

Regla 4: Existencia de un catálogo dinámico basado en el modelo relacional.

La información del modelo relacional (metadatos) debe ser almacenada en tablas y administrada como cualquier dato ordinario contenido en la base de datos. Esta información debe estar disponible a usuarios y serán accedidos de la misma forma que los datos ordinarios, empleando un lenguaje relacional estándar.

Regla 5: Soporte de varios lenguajes de datos.

La base de datos relacional puede tener soporte para varios lenguajes, sin embargo, esta debe contar con el soporte para un lenguaje bien definido, lenguaje declarativo, que tenga soporte para definición de datos, definición de vistas, manipulación de datos, constraints de integridad, autorización, y administración de transacciones (commit, rollback, begin).

Regla 6: Actualización de vistas.

Cualquier vista que teóricamente es actualizable debe hacerse a través del sistema.

Regla 7: Insert, update y delete de alto nivel.

La base de datos debe contar con un soporte para realizar operaciones de insert, update, y delete con un lenguaje de alto nivel (al igual que select), fácilmente.

Regla 8: Independencia física de los datos.

Aplicaciones y herramientas que interactúan con la base de datos no deben sufrir cambios si se altera la estructura y los métodos de acceso a datos.

fácilmente.

Regla 8: Independencia física de los datos.

Aplicaciones y herramientas que interactúan con la base de datos no deben sufrir cambios si se altera la estructura y los métodos de acceso a datos.

Regla 9: Independencia lógica de los datos.

Aplicaciones y herramientas no deben ser afectadas al realizar cambios a tablas y su estructura mientras sigan conservando los datos antes de dicha modificación. Ejemplo:

- Cambiar el orden de columnas
- Insertar una columna.

Regla 10: Independencia de la integridad.

Las restricciones de integridad deben ser definidas a través del lenguaje relacional y almacenadas en el sistema (Diccionario de datos), y no deben estar definidas del lado de la aplicación.

Regla 11: Independencia de la distribución.

Tanto usuarios como programas no deben tomar en cuenta o resultar afectados si se cambia la ubicación o si se realiza una distribución física de los datos (Bases de datos distribuidas vs locales)

Regla 12: No subversión.

Si el sistema tiene lenguajes de bajo nivel, estos lenguajes de ninguna manera pueden ser usados para violar la integridad de las reglas y restricciones expresadas en un lenguaje de alto nivel (SQL).

Regla 0: Regla cero.

Para que un sistema de bases de datos relacional se le denomine como tal, deberá usar (exclusivamente) sus capacidades relacionales para gestionar la base de datos.

Pregunta

Estructura lógica

Examen

→ Como visualizamos nuestra información,
tablas, tuplas, etc.

► 3.5 Índices

- Son estructuras de datos
 - Requieren almacenamiento
- Su utilidad es ubicar a un dato de forma eficiente
- Ordenar datos ← El manejador crea un índice implícito al momento de buscar
- Reduce los tiempos y recursos requeridos para procesar una consulta
- Los índices se aplican a las columnas de una tabla

¿Qué ocurre para agilizar las búsquedas?

Pregunta

Consulta se tarda por la cantidad de accesos
a "disco"

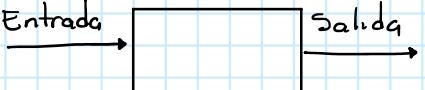
Examen

- El tiempo requerido para localizar a un dato, prácticamente es el tiempo que se requiere para

- El tiempo requerido para localizar a un dato, prácticamente es el tiempo que se requiere para acceder al disco
- El índice evita o reduce el número de lecturas a disco requeridos para localizar un dato.

Estructura Básica

→ BD

- Llave (Key) ↔ Valores de las columnas indexadas. (No necesariamente PK)	- KEY → Palabra
- Valor ↔ Puntero, dirección o ubicación exacta en disco.	- Valor ↔ Núm. página
en el que se encuentran los registros con los valores de la columna indexada solicitada	Entrada →  Salida
Entrada →  Salida	* Base de datos Padrón Electoral ↳ Pachuca, Hidalgo
* En Oracle este valor se le llama ROW-IO	

Indexado, tabla cliente sin rowid

CLIENTE

CLIENTE_ID	NOMBRE	AP_PATERNO	AP_MATERNO	ENTIDAD	SEXO	CODIGO_POSTAL	EMAIL
1000	JUAN	LUNA	JUAREZ	ZAC	M	32948	juan@m.com
1002	MARIO	MARTINEZ	AGUILAR	AGS	M	00293	mario@h.com
1003	ALEJANDRO	PEREZ	MORALES	GRO	M	02934	ale@hd.com
1004	HUGO	LINARES	HURTADO	DF	M	20398	hg@aol.com
1005	TANIA	SANCHEZ	GUTIERREZ	GRO	F	98909	ts@em.com
1006	ALONSO	LUGO	OLVERA	AGS	M	02934	alo@yo.com
1007	MIRIAM	MONDRAGON	RUBIO	YUC	F	02934	lug@tr.com
1008	JULIETA	ZAVALA	YAÑEZ	CHIH	F	02342	miri@msn.com
1009	JUAN	BENITEZ	LOPEZ	NL	M	02349	beno@su.com
1010	MARIA	JIMENEZ	BALDERRAMA	GRO	F	03453	mja@aol.com
1011	PEDRO	LUNA	JIMENEZ	NULL	M	02343	pe@mail.com

Indexado, tabla cliente con rowid

CLIENTE

ROW_ID	CLIENTE_ID	NOMBRE	AP_PATERNO	AP_MATERNO	ENTIDAD	SEXO	CODIGO_POSTAL	EMAIL
AAASSdAAEAAAASdAAA	1000	JUAN	LUNA	JUAREZ	ZAC	M	32948	juan@m.com
AAASSdAAEAAAASdAAB	1002	MARIO	MARTINEZ	AGUILAR	AGS	M	00293	mario@h.com
AAASSdAAEAAAASdAAC	1003	ALEJANDRO	PEREZ	MORALES	GRO	M	02934	ale@hd.com
AAASSdAAEAAAASdAAD	1004	HUGO	LINARES	HURTADO	DF	M	20398	hg@aol.com
AAASSdAAEAAAASdAAE	1005	TANIA	SANCHEZ	GUTIERREZ	GRO	F	98909	ts@em.com
AAASSdAAEAAAASdAAF	1006	ALONSO	LUGO	OLVERA	AGS	M	02934	alo@yo.com
AAASSdAAEAAAASdAAG	1007	MIRIAM	MONDRAGON	RUBIO	YUC	F	02934	lug@tr.com
AAASSdAAEAAAASdAAH	1008	JULIETA	ZAVALA	YAÑEZ	CHIH	F	02342	miri@msn.com
AAASSdAAEAAAASdAAI	1009	JUAN	BENITEZ	LOPEZ	NL	M	02349	beno@su.com
AAASSdAAEAAAASdAAJ	1010	MARIA	JIMENEZ	BALDERRAMA	GRO	F	03453	mja@aol.com
AAASSdAAEAAAASdAAK	1011	PEDRO	LUNA	JIMENEZ	NULL	M	02343	pe@mail.com



- Cadena de 18 caracteres

- 000000 | Data Object number, indica el segmento (Tabla)
| donde se encuentra registro

- FFF | Número de data file donde se encuentra el registro

- BBBB | Número de bloque de datos

- RRR | Número de registro dentro del bloque

3. Index

Friday, 14 February 2020 7:11 AM

→ d) Que acciones realiza el DBMS?

Cuando decide hacer uso de un índice para ejecutar una sentencia SQL?

Select * from cliente where email = 'm@m.com'

- 1) El DBMS realiza un escaneo del índice el cual puede ser:
 - Full index scan
 - Partial index scan (Hojas del árbol)
 - Unique index scan (Hojas del árbol)
- 2) Al escanear, se obtiene los ROW_IDs de los registros de interés
- 3) Condiciona lista, se accede a la tabla realizando una operación de:
 - "table Access by index ROW_ID"Y se recuperan los registros solicitados

→ d) Que tanto hay que indexar?

// Recuerda que la PK ya está indexada

• Existen predicados que involucran columnas que se usan con alta frecuencia

- ① Identificadores (PKs)
- ② RFC, EMAIL, CURP, FOLIOS, ETC

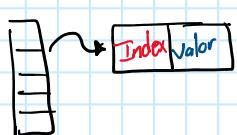
- La diversidad de valores de la columna es muy alta, cercana a ser valores únicos
- El número esperado de registros es bajo $\approx 2 \leq x \leq 4$
(De los registros de la tabla)
- La cardinalidad de la tabla debe ser alta. ($\frac{\text{Cuantos registros}}{[\text{Registros existentes}]}$)
- Si el índice no se usa, se realiza una operación denominada "Full Table Access"
- Impacto al tener múltiples índices
(Problemas de desempeño en operaciones de inserción, actualización o eliminación)

(Problemas de desempeño en operaciones de inserción, actualización o eliminación)

→ Tipos de Índices



- Hacen uso de un hash table



$$\text{Key} = f_{\text{hash}}(\text{dato})$$

Funcióñ hash

- Búsqueda por llaves:

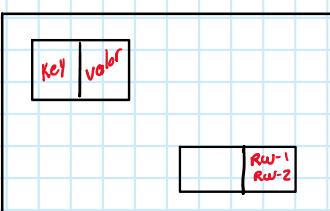
$$\text{search}(\text{Key}) = \{\text{valores}\}$$

- Búsqueda tienen una complejidad algorítmica constante

- Características:

$$\text{Key} = f_{\text{hash}}(\text{Columna Indexada})$$

$$\text{Valor} = \text{Lista De Row-IDs}$$



- * Advantage

① Búsqueda eficiente

- * Disadvantage

① Memoria limitada (Costo)

② Keys no pueden ser nulos

③ No soporta ordenamiento

Ejemplo

Buscar registro con 'm@m.com'

$$\text{search}(f_{\text{hash}}('m@m.com')) = \{\text{ROW-ID}\}$$

→ Bit Map Index

- Formado por una matriz o mapa de bits en donde:
 - Columnas representan a cada uno de los valores
 - En la fila se indica si el valor es - o no - en la tabla

- Columnas representan a cada uno de los valores de la tabla (existe una columna por cada valor de frente)
- Renglones, existe un renglón por renglón por cada región de la tabla

ROW_ID	CLIENTE_ID	NOMBRE	AP_PATERNO	AP_MATERNO	ENTIDAD	SEXO	CODIGO_POSTAL	EMAIL
AAASSdAAEAAA5dAAA	1000	JUAN	LUNA	JUAREZ	ZAC	M	32948	juan@m.com
AAASSdAAEAAA5dAAB	1002	MARIO	MARTINEZ	AGUILAR	AGS	M	00293	mario@h.com
AAASSdAAEAAA5dAAC	1003	ALEJANDRO	PEREZ	MORALES	GRO	M	02934	ale@fd.com
AAASSdAAEAAA5dAAD	1004	HUGO	LINARES	HURTADO	DF	M	20398	hg@aol.com
AAASSdAAEAAA5dAAE	1005	TANIA	SANCHEZ	GUTIERREZ	GRO	F	98909	ts@em.com
AAASSdAAEAAA5dAAF	1006	ALONSO	LUGO	OLVERA	AGS	M	02934	alo@yo.com
AAASSdAAEAAA5dAAH	1007	MIRIAM	MONDRAGON	RUBIO	YUC	F	02934	lug@tr.com
AAASSdAAEAAA5dAAI	1008	JULIETA	ZAVALA	YANEZ	CHIH	F	02342	miri@msn.com
AAASSdAAEAAA5dAAJ	1009	JUAN	BENITEZ	LOPEZ	NL	M	02349	beno@su.com
AAASSdAAEAAA5dAAK	1010	MARIA	JIMENEZ	BALDERRAMA	GRO	F	03453	mja@aol.com
AAASSdAAEAAA5dAAK	1011	PEDRO	LUNA	JIMENEZ	NULL	M	02343	pe@mail.com

Row-ID

R1	ZAC	AGS	GRO	DF	YUC	CHIH	NL	NULL
R1	1	0	0	0	0	0	0	0
R2	0	1	0	0	0	0	0	0
R3	0	0	1	0	0	0	0	0
R4	0	0	0	1	0	0	0	0
R5	0	0	1	0	0	0	0	0
R6	0	1	0	0	0	0	0	0

① Ubicar la columna = 'GRO'

② Escanea la columna obtiene Row-ID, donde bit = 1

③ Empleando la lista de Row-IDs se accede a la tabla y se recuperan datos

← Esto lo hace atractivo en sistemas OLAP

Analytical ↑

* Ventajas

- Soportan nulos
- Poca memoria
- Útil aunque exista baja frecuencia de valores diferentes

* Desventajas

- No soporta ordenamiento

→ B-tree Index

- B ≈ Balanceado

- Características

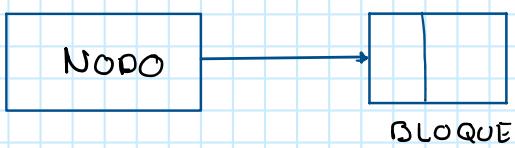
① Nodos tienden a tener el número de elementos

② Su crecimiento es más hacia los lados

Los nodos tienen que tener el número de elementos

② Su crecimiento es más hacia los lados

③ Cada nodo tiene una capacidad máxima de almacenamiento, se asocia a un bloque de datos



- Esta asociación permite que con una sola lectura a disco se obtiene el contenido del nodo

④ Por su estructura, el árbol se vuelve productivo en búsquedas

// Ejemplo

• Si la tabla contiene datos, se realiza un ordenamiento inicial.

ROW_ID	CLIENTE_ID	NOMBRE
1	3	JUAN
2	2	EVA
3	3	SARA
4	4	ELI
5	5	HUGO
6	6	JULIO
7	7	IVAN
8	8	IRMA
9	9	ARA
10	10	MIRA
11	11	VIRI
12	12	BALO
13	13	BILLY
14	14	JULY
15	20	OMAR
16	21	GIL
17	22	LUIT
18	25	MARA

// Ordenamiento

$$L = \{ \text{ara}, \text{bily}, \text{eva}, \dots, \text{yuri}, \text{zuly} \}$$

// Creación del árbol

① Suponer capacidad máxima de 4 elementos por nodo

② En un árbol "B tree", cada elemento de un nodo está formado por

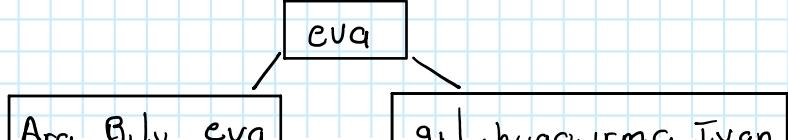
$$E = \{ \text{etiqueta}, \text{Puntero} \} \quad \text{Nodo que no es hoja}$$

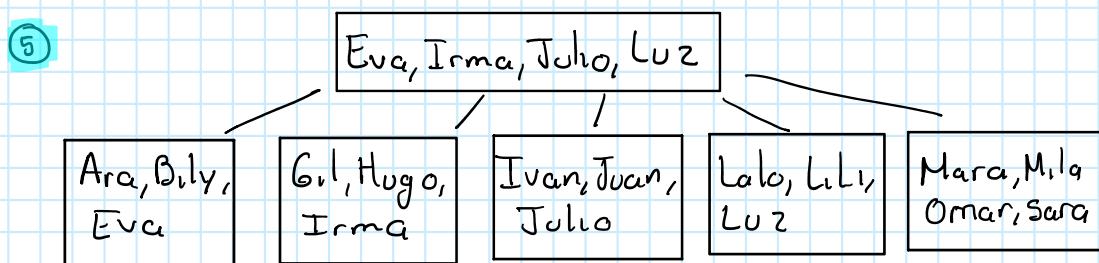
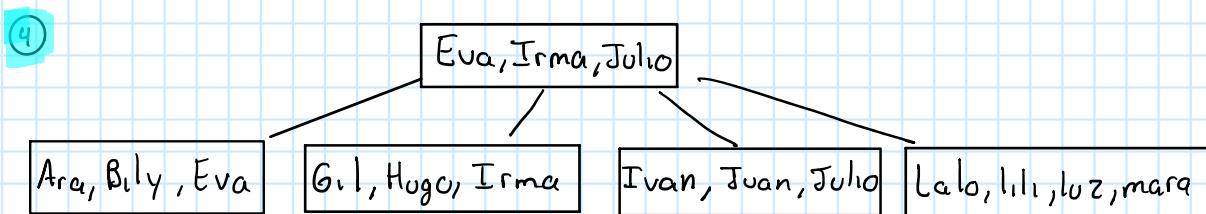
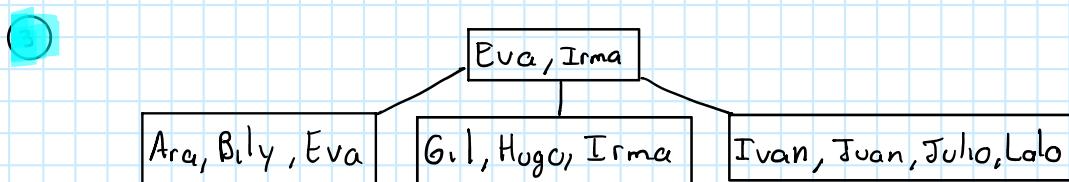
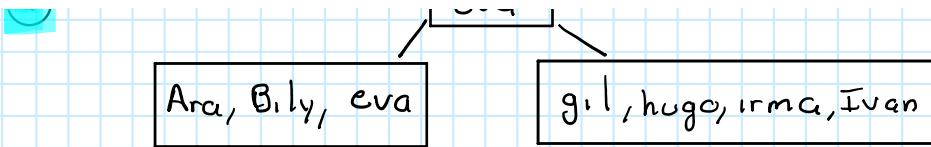
$$E = \{ \text{etiqueta}, ? \} \quad \text{Nodo que es hoja}$$

1

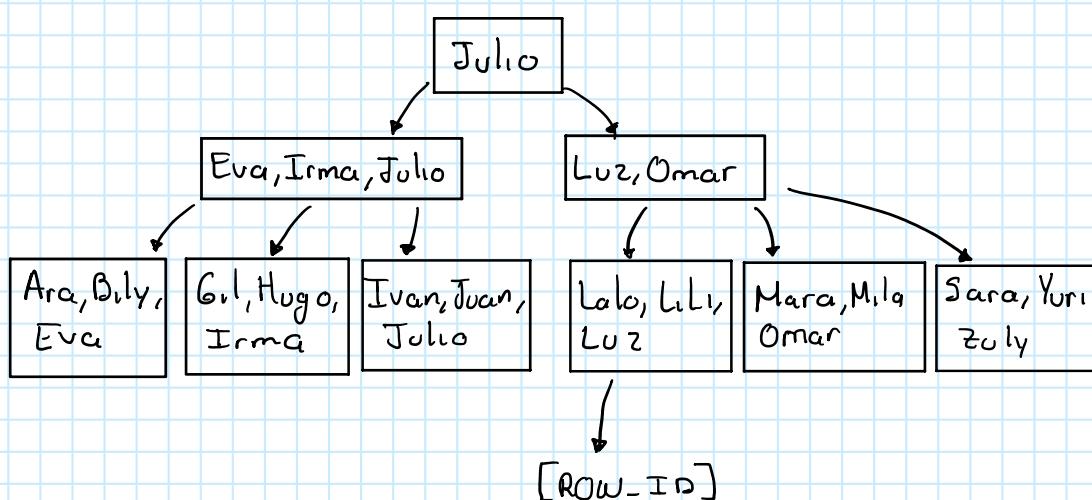
Ara, Bily, eva, gil

2



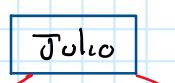


6 // Paso mortal



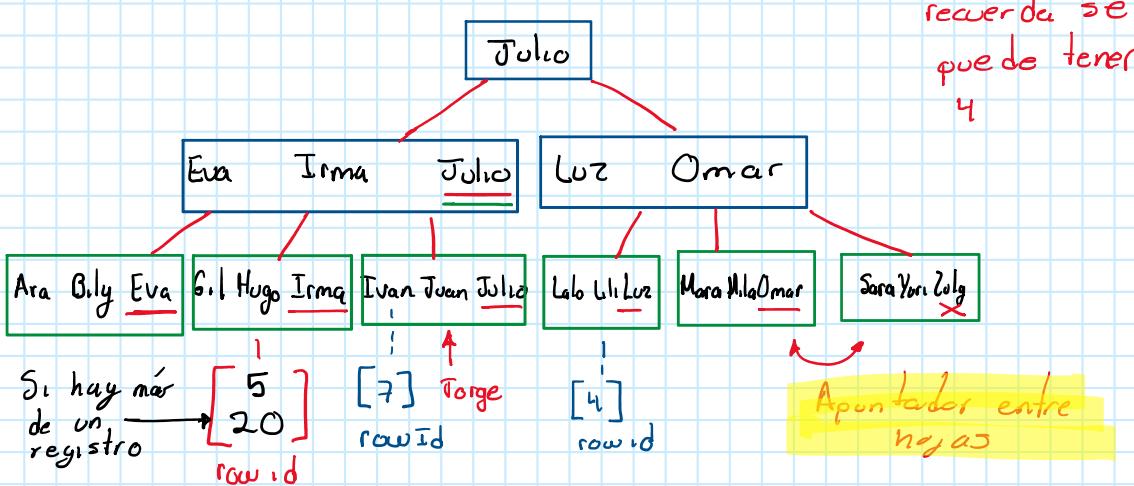
// Algoritmo Esq.

// Si es de 4 entonces los cuadros tienen $(n-1)$



NOTA

Para insertar
recuerda se
que de tener



- ¿Cuántas lecturas se hacen para obtener algún integrante del árbol?
 - Checamos la profundidad del árbol y por lo tanto = 3 //

// Algoritmo de Búsqueda

- Recorrer el árbol hasta llegar a un nodo hoja
(Si el valor es menor checamos apuntador y hasta checar coincidencia)
- Leer los ROW-ID asociados a las etiquetas
- Acceder a la tabla con los ROW-ID y recuperar los registros (Table Access by Row-ID)

Nota: Si en vez de cadenas tenemos las llaves primarias artificiales (Números), podemos tener mejor rendimiento.

- ¿Cuántos elementos máximo?

Árbol 1 Profundidad = 1
Elementos = 4

$$\textcircled{1234} \quad \therefore 4$$

Árbol 1 Profundidad = 2
Elementos = 4

∴ 20 elementos totales $(2)(4+1) = 20 + 4 = 24$

$$\therefore 20 \text{ elementos totales} \quad (2)(4+1) = 20 + 4 = 24$$

Árbol | Profundidad = 3
| Elementos = 4

$$(2)(4)(4+1) = 40 + 24 = 64$$

• d) Que pasaría si select * from cliente where nombre is null?

R= El manejador no usará el índice debido a que no soporta nulls, en cambio, utilizaría "Table Access Full"

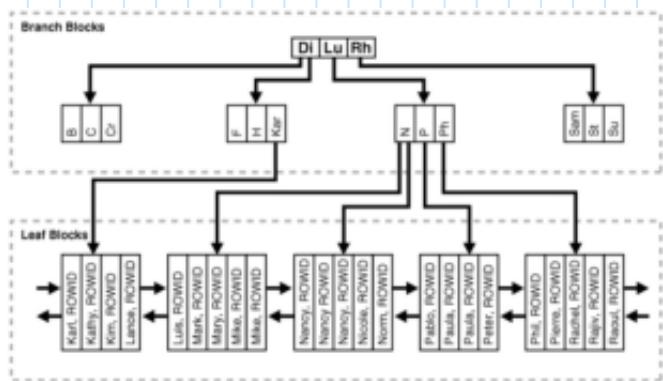
• c) Y si es ... is not null?

R= Si usará el índice debido a que si es soportado utilizaría "Index Access Full"

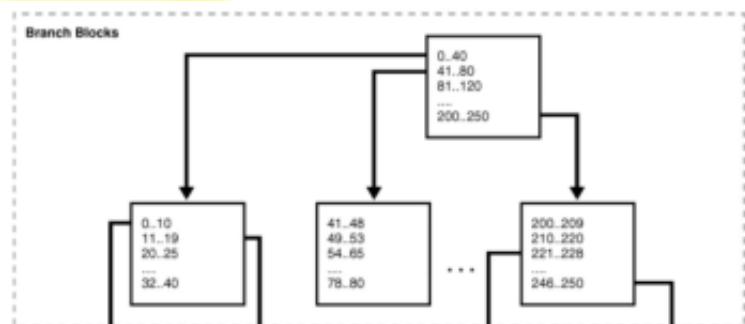
• d) Qué pasaría si en un index tipo Hash usamos select * from cliente where nombre is null?

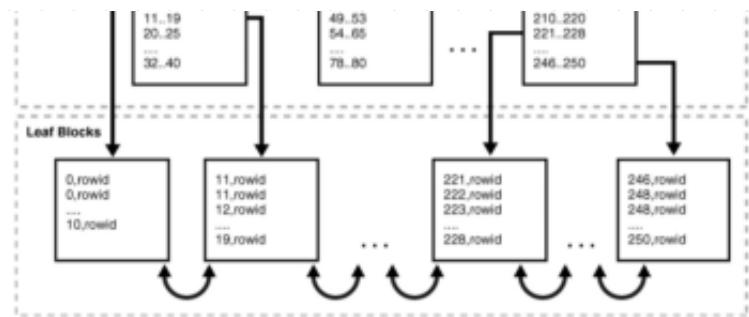
R= No funcionan los buscadores si no tienen un operador de comparación

A) Estructura con Palabra



B. ESTRUCTURA INTERNA DE UN ÁRBOL B+ EN ORACLE CON DATOS NUMÉRICOS





RESUMEN

Thursday, 27 February 2020 4:44 PM

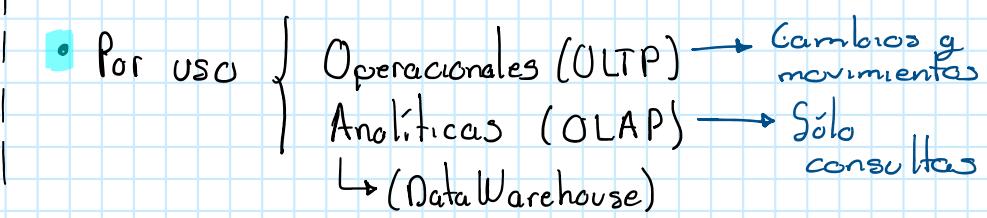
- Dato**
- Empleado para representar a través de un lenguaje un hecho, condición
 - Conjunto de símbolos en crudo (raw data)

- Información**
- Tienen significado y valor
 - Sirven para asumir tomas de decisiones
- Data → Procesamiento → Información

- Bases de datos**
- Colección de datos inter-relacionados que representa información de usuario final
 - Se divide
 - Datos Usuarios
 - Metadatos → Datos obtenidos mediante otras
 - ↳ Lista - Relaciones
 - Tipos de datos
 - Longitud dato

- DBMS**
- Data Base Management System
 - Colección programas para administrar estructura y control
 - BD se almacena en data files (.dbf)

- Tipos BD**
- Por número de usuario
 - ↳ Mono-usuario/workgroup/enterprise
 - Por ubicación
 - ↳ Centralizadas/distribuidas
 - Por uso
 - ↳ Operacionales (OLTP) → Cambios y movimientos



EXAMEN 1

Thursday, 27 February 2020 4:28 PM

1. Para cada una de las preguntas del lado izquierdo, seleccione una respuesta del lado derecho. (las respuestas se emplean una sola vez, no contestar en esta hoja).

30 p

- A. Conjunto de valores que se obtienen al leer el contenido de un índice, fundamentales para obtener los datos solicitados de una consulta.
B. Elemento del modelo relacional empleado para validar la no duplicidad de valores en una columna sin importar que existan valores nulos.
C. Elemento del modelo relacional empleado para minimizar el número de accesos a disco requeridos al localizar un dato. Por sus características permiten el manejo de valores nulos de forma eficiente.
D. Condición de los datos no deseada ya que puede generar conflictos de valores distintos para un mismo dato, sin embargo, en algunas ocasiones se permite para mejorar el desempeño de las consultas.
E. Colección de datos organizados y almacenados sistemáticamente que describen la estructura física y lógica de los datos del usuario final.
F. Objeto de un ORDBMS cuya principal característica es su uso en búsquedas ya que el tiempo de respuesta tiende a ser el mismo sin importar el tamaño de los datos, pero su debilidad es el ordenamiento de datos y el uso de memoria requerido.
G. En el modelo relacional, elemento cuyos valores nunca pueden ser nulos, tienen significado para las reglas de negocio e identifican de manera única a los registros de una tabla.
H. Formado por elementos como hardware, software, personal con roles especializados en bases de datos, componentes de red, etc.
I. En la siguiente relación: Un curso es impartido por un profesor y un profesor imparte varios cursos, los atributos de cada tabla en términos de PKs, y FKs al asociarse serán:
J. Describe la estructura y las características de los datos, formado por entidades, atributos y relaciones entre entidades.
K. Principal artefacto que se genera dentro del proceso de diseño de una base de datos. En él se especifican los elementos particulares (dependientes) del modelo seleccionado, y es totalmente independiente de la estructura física de almacenamiento de los datos.
L. Artefacto generado durante el proceso de diseño de una base de datos cuyo objetivo principal es homologar la visión de los datos sin importar el modelo en el que será implementada
M. Contenido de un nodo intermedio en un árbol B-tree (el nodo no forma parte de las hojas del árbol).
N. Manejador de base de datos que implementa el modelo relacional y hace uso de los conceptos de la programación orientada a objetos (POO) para referirse a cada uno de sus elementos.
O. Se obtiene como resultado de aplicar una interpretación o procesamiento de un conjunto de símbolos, signos, o mediciones que son de interés para un usuario final.
1. Diseño lógico
2. Modelo lógico
3. Modelo conceptual
4. Modelos de datos
5. Modelos relacionales
6. Modelo en Red
7. Modelo jerárquico
8. Data
9. Información
10. Redundancia
11. Inconsistencia
12. Falta de integridad de datos
13. (KEY,ROW_ID)
14. (ROW_ID, PUNTERO)
15. (ROW_ID, VALOR DE LA PI)
16. (KEY, PUNTERO)
17. PROFESOR{profesor_id(PK)} y
CURSO{curso_id(PK)}, profesor_id(PK)
18. PROFESOR{profesor_id(PK)} y CURSO
(curso_id(PK), profesor_id(PK))
19. PROFESOR{profesor_id(PK), curso_id(PK)} y
CURSO{curso_id (PK)}
20. PROFESOR{profesor_id(PK), curso_id(PK)} y
CURSO{curso_id (PK)}
21. Llave primaria natural
22. Llave primaria artificial
23. Llave primaria compuesta
24. Llave foránea
25. Base de datos
26. Diccionario de datos
27. Sistema de base de datos
28. ROW_ID
29. Valor de la columna indexada
30. Diseño conceptual
31. Medio-entidad-relación
32. DBMS
33. RDBMS
34. ODBCMS
35. ORDBMS
36. Índice tipo Hash
37. Índice B&Tree
38. Índice con árboles B+
39. Índice tipo "único"

- Conjunto de valores que se obtienen al leer contenido de un índice, fundamentales para obtener los datos solicitados de una consulta

↳ ROW-ID

- Elemento del modelo relacional empleado para validar la no duplicidad de valores en una columna (sin importar existir valores nulos)

↳ Unique

- Elemento del modelo relacional empleado para minimizar el núm. de accesos a disco requeridos al localizar un dato. Por sus características permiten manejar valores nulos de forma eficiente

↳ Índice Bit Map

- Condición de los datos no deseada ya que puede generar conflictos de valores distintos para un mismo dato, en algunas ocasiones, permite mejorar desempeño

↳ Redundancia

- Colección de datos organizados y almacenados sistemáticamente

↳ Data

- Colección de datos organizados y almacenados sistemáticamente describen la estructura física y lógica de los datos del usuario → Diccionario de Datos

- Objeto de un ORDBMS cuya principal función es su uso en búsquedas (tamaño no importa en el tiempo) su debilidad es el ordenamiento y uso de memoria

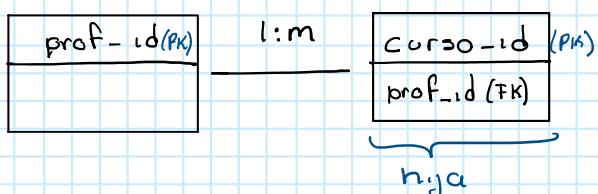
↳ Índice tipo flash

- En el modelo relacional, elementos cuyos valores nunca pueden ser nulos, tienen significado para reglas de negocio e identifican de manera única a los registros de una tabla → Llave primaria natural

- Formado por elementos como hardware, software, personal con roles especializados en BDs

↳ Sistema de Base de datos

- Un curso es impartido por un profesor y un profesor imparte varios cursos



- Describe la estructura la estructura y datos formados por entidades, atributos y relaciones entre entidades

↳ Modelo datos

- Principal artefacto que genera dentro del proceso de diseño de una BD.

- Se especifican los elementos particulares (Dependientes) del modelo seleccionado
- Es totalmente independiente de la estructura física de almacenamiento de los datos

↳ Modelo relacional → ¿Modelo lógico?

- L: Artefacto generado durante el proceso de diseño de una BD cuyo objetivo es homologar la visión

de una BD cuyo objetivo es homologar la visión de los datos sin importar el modelo

↳ Modelo Conceptual

- Contenido de un nodo intermedio en un árbol B tree (Nodo no forma parte de las hojas)
 - ↳ (KEY, PUNTERO)

Si formara parte de las hojas : (Key, Row-ID)

- Manejador de la base de datos que implementa el modelo relacional y hace referencia a POO

↳ ORDBMS

- Se obtiene como resultado de aplicar una interpretación o procesamiento de un conjunto de símbolos, signos o mediciones que son de interés para usuario final → Información

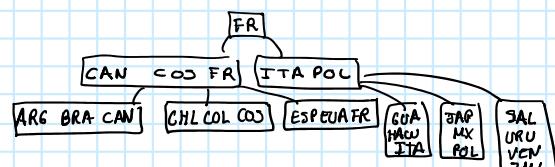
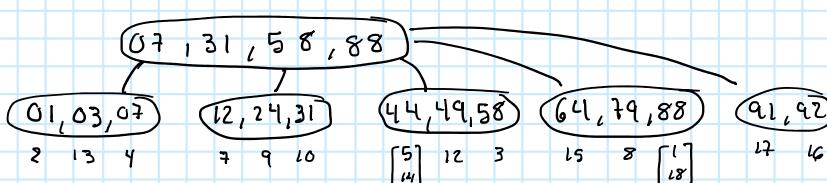
② 1 2 3 4 5 6 7 8 9 ← Row-ID
88 01 58 07 44 92 12 79 24

10 11 12 13 14 15 16 17 18 19 20 ← Row-ID
31 null 41 9 03 44 64 null 91 88 null null

01, 03, 07, 12, 24, 31, 41 4, 9, 58, 64, 7 9, 88, 91, 92

Máximo nodo 4K
cada dato = 1K

a)



b) ▶ select * from ____ where ____ is null

∴ Manejador no usará índice, ya que no soporta nulos → **Full Table Access**

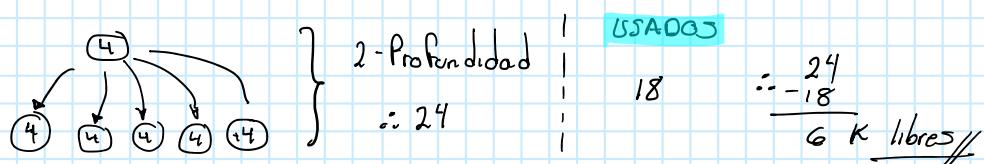
▶ select * from ____ where ____ is not null

nulos

Full Table Access

- ▶ select * from ____ where ____ is not null
↳ Si usaría el índice pero con "Index Access Full"
- ▶ Index tipo Hash
↳ No funciona la búsqueda si no tiene operador de comparación
- ▶ Acceder a la tabla con los Row-ID y recuperar registros → Table Access by Row-ID

c) Espacio Libre



d) Primero leería la consulta, para posteriormente realizar realizar un "Table Access by Row-ID" una vez realizada esta operación, nos mostraría $(1, 18) \leftarrow$ Lista Row ID

El recorrido del árbol sería a través de la comparación de las "Key" mediante su tamaño menor a la izquierda y mayor a la derecha hasta llegar al nodo hoja correspondiente.



Bit Map → Soporta nulos

↳ Malo para ordenar

Hash Map → Eficiente en búsqueda

↳ Requiere muchos recursos

B-tree → Bueno en búsquedas

↳ No soporta nulos

a)

Row-ID	OP	H	O	PL	TM	B	E	AM	NULL
001	0	0	0	0	0	0	1	0	0
002	0	0	0	0	0	1	0	0	0
003	0	0	0	0	1	0	0	0	0
004	0	0	0	0	0	0	0	0	1
005	0	0	0	0	0	0	0	0	1
006	0	0	0	0	0	1	0	0	0
007	0	0	0	0	0	0	1	0	0
008	0	0	0	1	0	0	0	0	0
009	0	0	0	0	0	0	0	1	0
010	0	0	0	0	0	0	0	0	1

008	0	0	1	0	0	0	0
009	0	0	0	0	0	1	0
010	0	0	0	0	0	0	1

b) select * from cliente where tipo = 'BE'
or tipo is null

- * Por un lado respecto a la cadena 'BE' devolverá 002, 005 y por lado de null 004, 005 y 010
- * Tomar en cuenta que Bit Map soporta valores nulos por lo que la búsqueda será eficiente.



a) Alumno (Nombre, apellido, fechaNacimiento)
 Curso (fechaInicio, fechaFin, idiomaImpartido)
 Profesor (nombre, apellido, tipo)
 Credencial (folio, vigencia, fechaExpedicion)
 Idioma (NivelRequerido, nombre, clave)
 ↳ idioma

Curso > M:N
 Alumno > - Alumno puede tomar muchos cursos
 - Cada curso puede tener hasta 15 alumnos

Curso > 1:M
 Profesor > - Cada curso lo imparte un profesor
 - Hay profesores que imparten al menos 2 cursos

Curso > 1:M
 Idioma > - Puede abrirse varios cursos para un mismo idioma
 ↳ Un curso → Un idioma
 Un idioma → Muchos cursos

Credencial /
 Alumno > 1:1



Restricciones

- Restricción Llave primaria (PK)
 - Natural
 - Artificial
 - Compuesta
- Emplea "not null" y "unique"
- Restricción de referencia (FK)
 - La FK debe existir en la tabla padre

• Restricción de integridad

- Not null/nonnull → Ausencia valor
- Unique → Valor único (No duplicados)
- Check → Validar mediante expresión booleana

• Validación triggers

a) Se desea que las búsquedas de alumnos empleando su CURP sea eficiente. Algunos no tienen null no pueden existir duplicados

- Para la duplicidad podemos usar restricción de integridad mediante Unique, por otro lado, para hacer la búsqueda eficiente y considerando que hay nulos podríamos usar un Bit Map

b) Clave_puesto asignada a un empleado ver si existe en el catálogo de puesto de la empresa además de validar que tenga un valor

- Para validar si tiene o no valor podemos usar una restricción de integridad y en específico Not Null/Null dependiendo de la forma de buscar

- Probablemente podría usarse una restricción de referencia mediante una FK