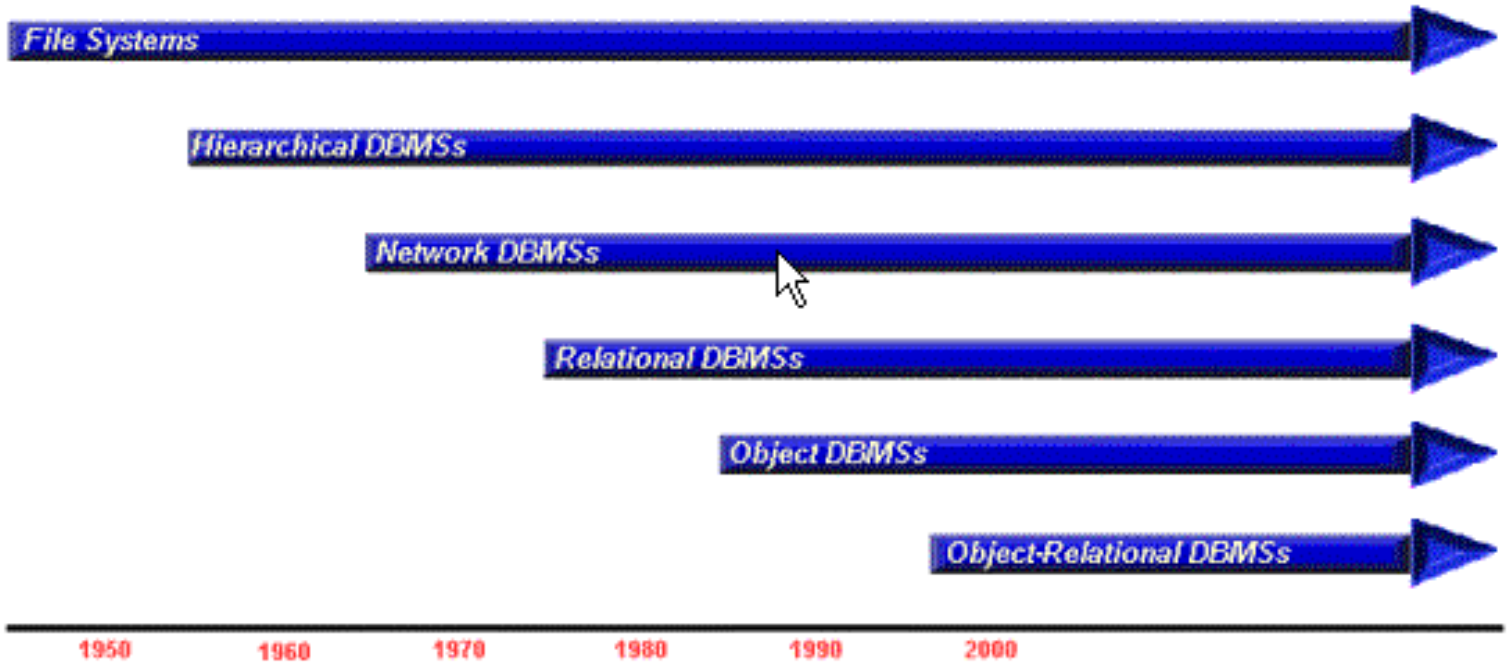
Curso de Bases de Datos

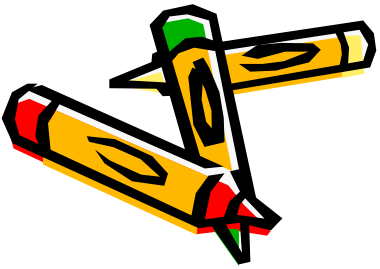
J. Antonio Camarena Ibarrola

Historia de las Bases de Datos



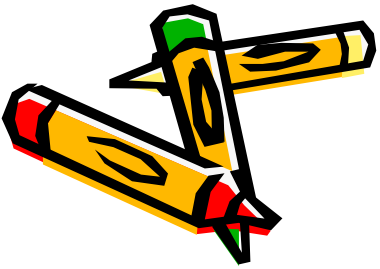
DBMS

- El software que permite usar y/o modificar una Base de Datos es llamado "Sistema Manejador de la Base de Datos" o simplemente DBMS por sus siglas en inglés.
- Un DBMS permite un manejo abstracto de los datos independiente de la manera en como realmente se almacenan físicamente
- Un DBMS actúa como interprete de un lenguaje de muy alto nivel (como SQL)



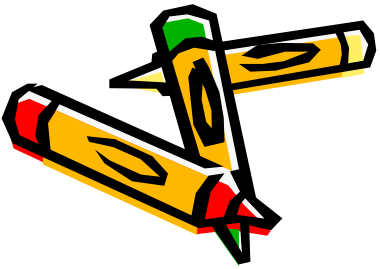
Funciones que un DBMS debe implementar

- Seguridad
- Integridad
- Sincronización
- Recuperación de datos



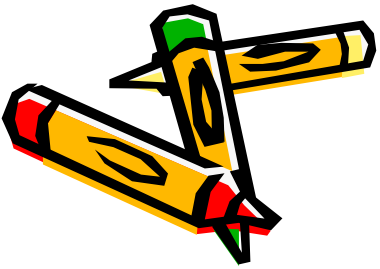
Niveles de abstracción en un DBMS

- La "Base de datos conceptual" es una representación abstracta de la "Base de datos física"
- Las "Vistas" son abstracciones de porciones de la base de datos (puede haber muchas vistas)

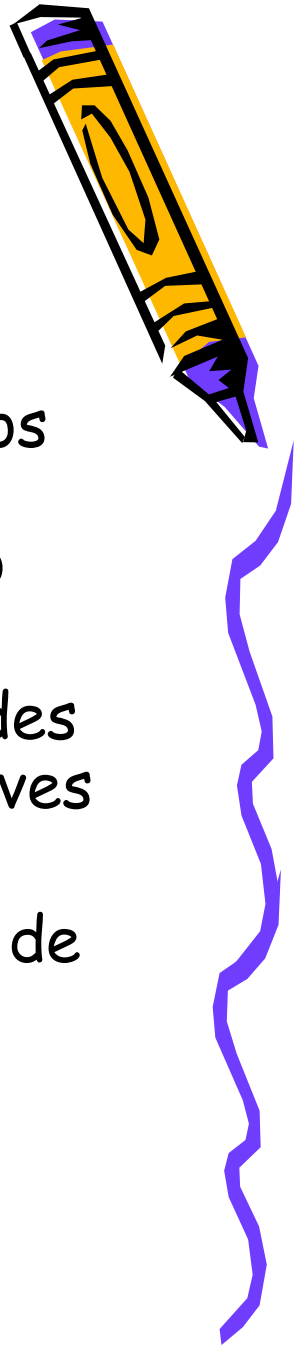


Esquemas e instancias

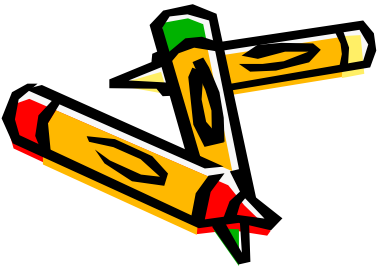
- El contenido de la Base de Datos en un momento determinado se conoce como "Instancia de la Base de datos"
- Cuando se diseña una Base de Datos se hacen planes. El término "Esquema" es utilizado para referirse a los planes relativos a la Base de Datos
- Las instancias cambian, los esquemas permanecen
- El Esquema conceptual se refiere a los planes de la base de datos conceptual, el esquema físico se refiere a los planes relativos a la base de datos física
- Los Planes relativos a las Vistas se conocen como "Subesquemas"



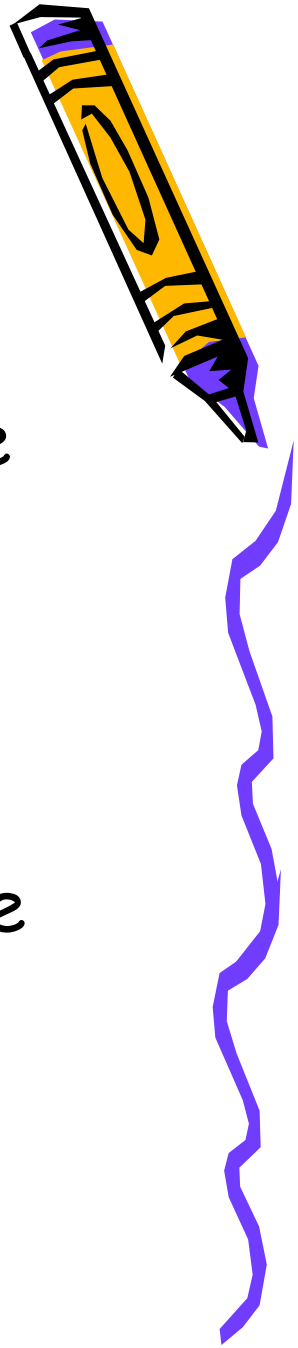
Los 3 grandes Modelos de datos



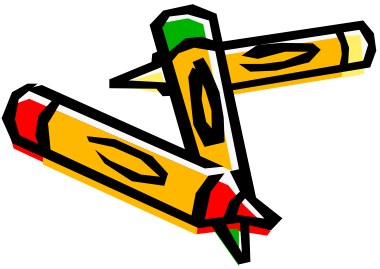
- Modelo Jerárquico. Basado en árboles donde los nodos representan entidades (Ej vuelos) y los hijos tienen una cierta relación con el, ejemplo (pasajeros que irán en el vuelo).
- Modelo de Red. Los nodos representan entidades similares y las aristas son asociaciones (Ej. Naves asignadas a vuelos)
- Modelo Relacional. Basado en la noción teórica de relaciones como conjuntos



Lenguaje de Definición de Datos o DDL

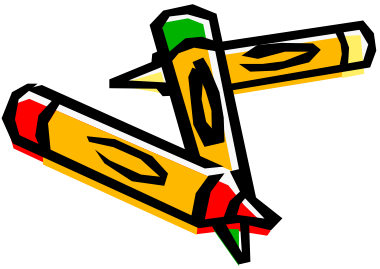


- Un DBMS debe proveer un "Lenguaje de Definición de Datos" para poder especificar el esquema conceptual en términos de un "Modelo de Datos"
- En Oracle, el DDL es un subconjunto de SQL (Create table, create index, create view, create synonym, drop table, drop index, etc..)



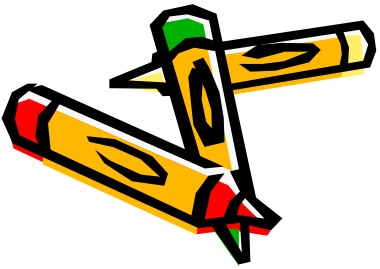
El lenguaje de Manipulación de Datos (DML)

- Para realizar consultas (lenguaje de consultas)
- Borrar, Insertar, Modificar datos
- En Oracle se utiliza un subconjunto de SQL (update, insert, delete, select, truncate, etc..)



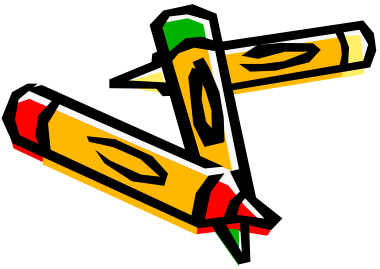
Lenguaje de Control de Datos (DCL)

- Para establecer cuales usuarios pueden acceder a cuales tablas
- Establecer roles de usuarios
- En SQL comandos create user, drop user, grant, revoke, etc



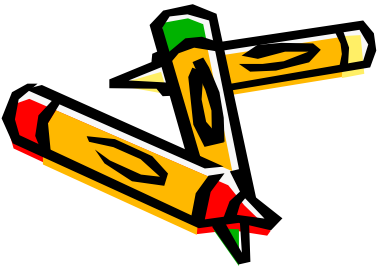
Independencia física de datos

- Las modificaciones en la organización de la base de datos física pueden afectar su desempeño pero no requerir cambios en los programas de aplicación ya que el esquema conceptual no ha cambiado solo su implementación física



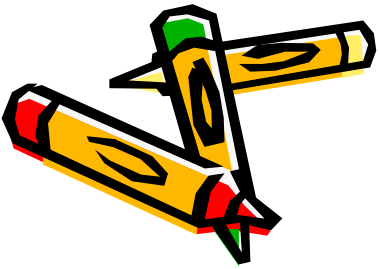
Independencia lógica de datos

- Se pueden hacer muchas modificaciones al esquema conceptual sin que implique modificación de los sub-esquemas o vistas. De nuevo, esto permite que no sea necesario reescribir programas de aplicación



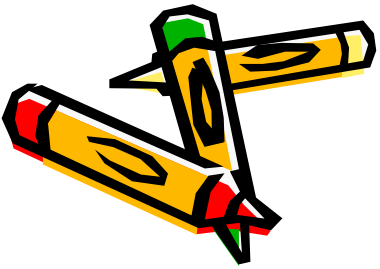
Definiciones

- Entidad. Algo que es distinguible y del cual almacenamos información
- Un grupo de entidades similares forman un "Conjunto de entidades"
- Atributos. Propiedades que tienen las entidades
- Llave primaria. Atributo o conjunto de atributos que identifican de manera única a una entidad
- Llave secundaria. Atributo o conjunto de atributos que no identifica a una entidad de manera única sino a un conjunto de entidades que comparten cierta propiedad
- Llave foránea. Es una llave secundaria que en otro conjunto de entidades es llave primaria



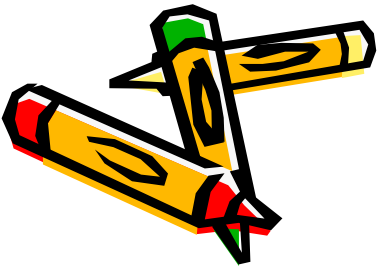
Obligaciones de un Administrador de Base de Datos

- Asegurar espacio físico para la base de datos actual y planear los requerimientos para el futuro
- Sintonizar la Base de Datos para optimizar el desempeño
- Monitorear y controlar los accesos a la base de datos
- Respaldar y en su caso recuperar la Base de Datos



Relaciones

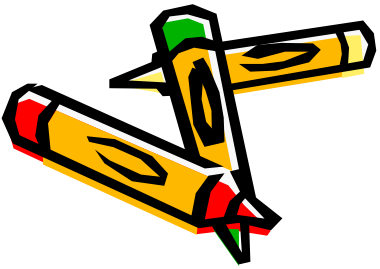
- Una relación entre 2 conjuntos A y B es un subconjunto de AxB. Los elementos de ese subconjunto son parejas ordenadas o duplas
- Una relación entre 3 conjuntos esta formada por tercias ordenadas, o triadas o 3-tuples
- Una relación entre n conjuntos esta formada por n-adas ó n-tuples
- Una relación entre un número indeterminado de conjuntos esta formada por tuples



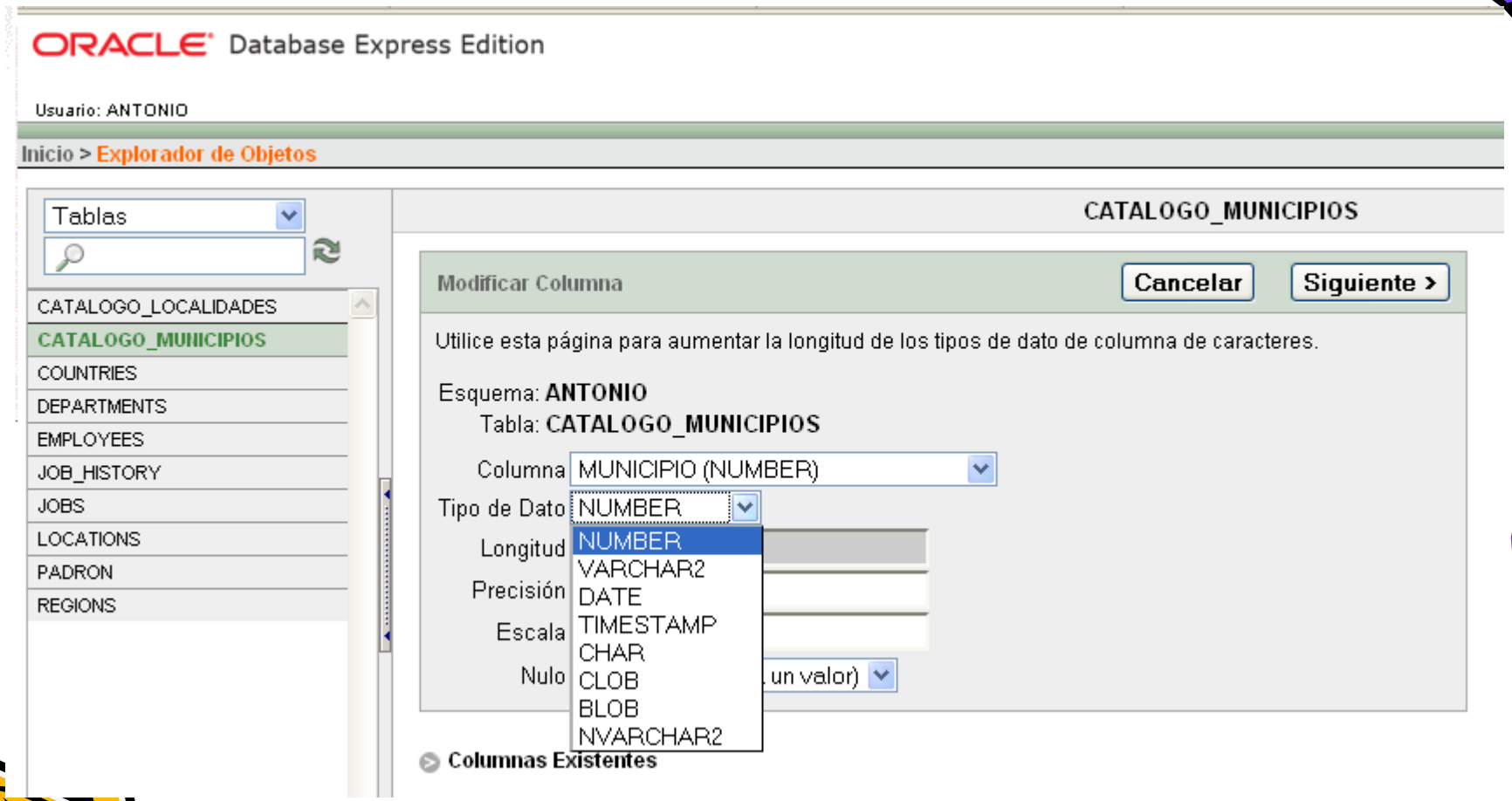
Las entidades se almacenan en tablas como renglones (tuples)

Los atributos de las entidades son columnas de la tabla

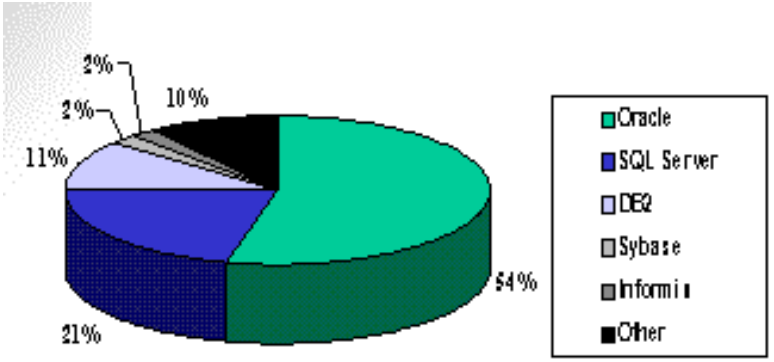
VENUE_ID	NAME	CITY	STATE	COUNTRY
1	3com Park	San Francisco	CA	USA
2	Agenda Lounge	San Jose	CA	USA
3	Altamont Raceway Park	Tracy	CA	USA
4	Amador Theatre	Pleasanton	CA	USA
5	Arco Arena	Sacramento	CA	USA
6	Argent Hotel	San Francisco	CA	USA
16	BR Cohn Winery	Glen Ellen	CA	USA
8	Backflip	San Francisco	CA	USA
9	Bay Meadows Racetrack	San Mateo	CA	USA
10	Berkeley Community Theatre	Berkeley	CA	USA



Tipos de datos



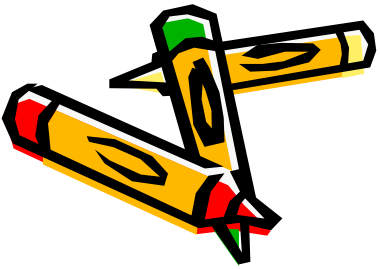
¿Por qué Oracle?

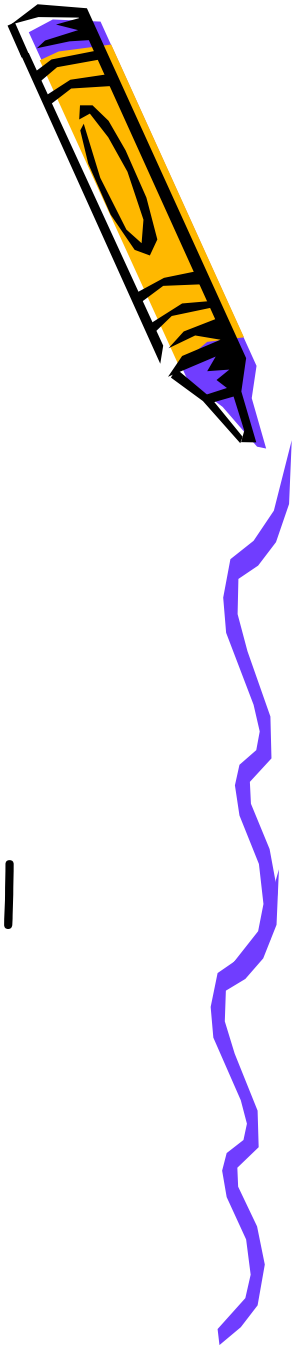


Ventajas en el manejo de seguridad
(Roles)
Ventajas en la garantía de la integridad de la Base de Datos
(Triggers)
Ventajas en la garantía de la recuperación de Información
(modo archivelog)
Ventajas para Administrar el espacio
(Tablespaces)

Propiedades de las Relaciones como Conjuntos de entidades

- Los elementos son tuples del mismo tipo
- Como en cualquier conjunto, los elementos no están ordenados ni existe ningún tipo de "cercanía" entre ellos
- Como en cualquier conjunto, no puede haber 2 elementos idénticos
- A diferencia de las n-adas ordenadas, los atributos tienen un nombre y no una posición, no hay "cercanía" entre columnas





Las tablas son relaciones impropias o corruptas …. E.F. Codd

- Permiten duplicados
- Existe el identificador de renglón (rowid)
- SQL no implementa toda la funcionalidad del álgebra relacional

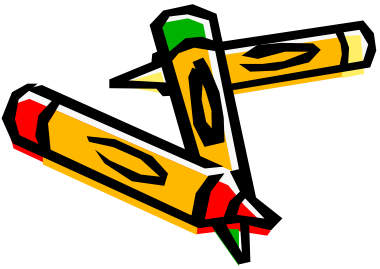
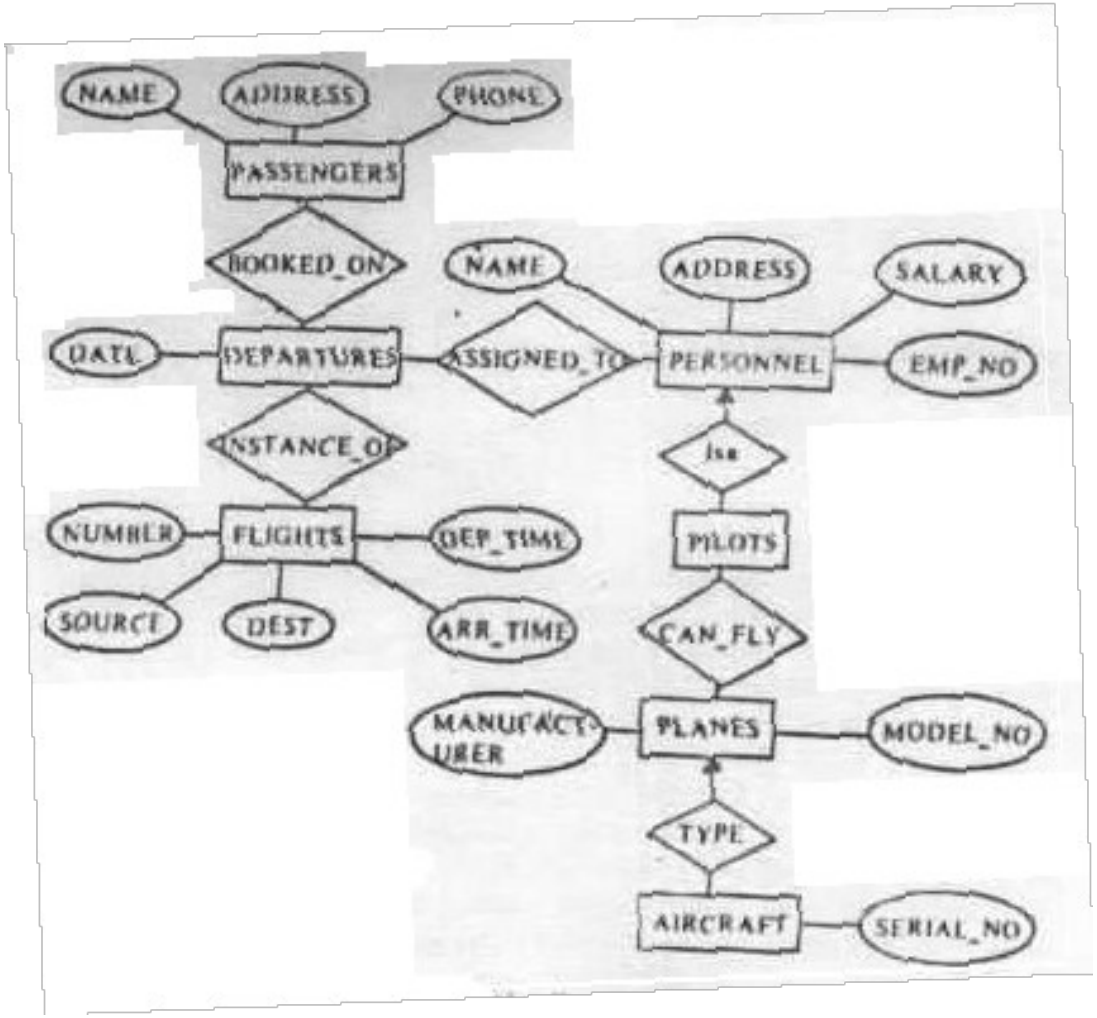
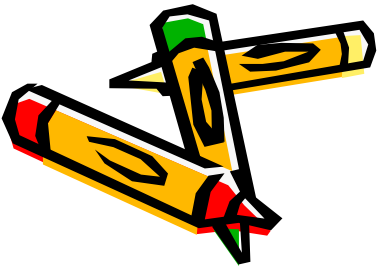


Diagrama Entidad-Relación de Aerolinea



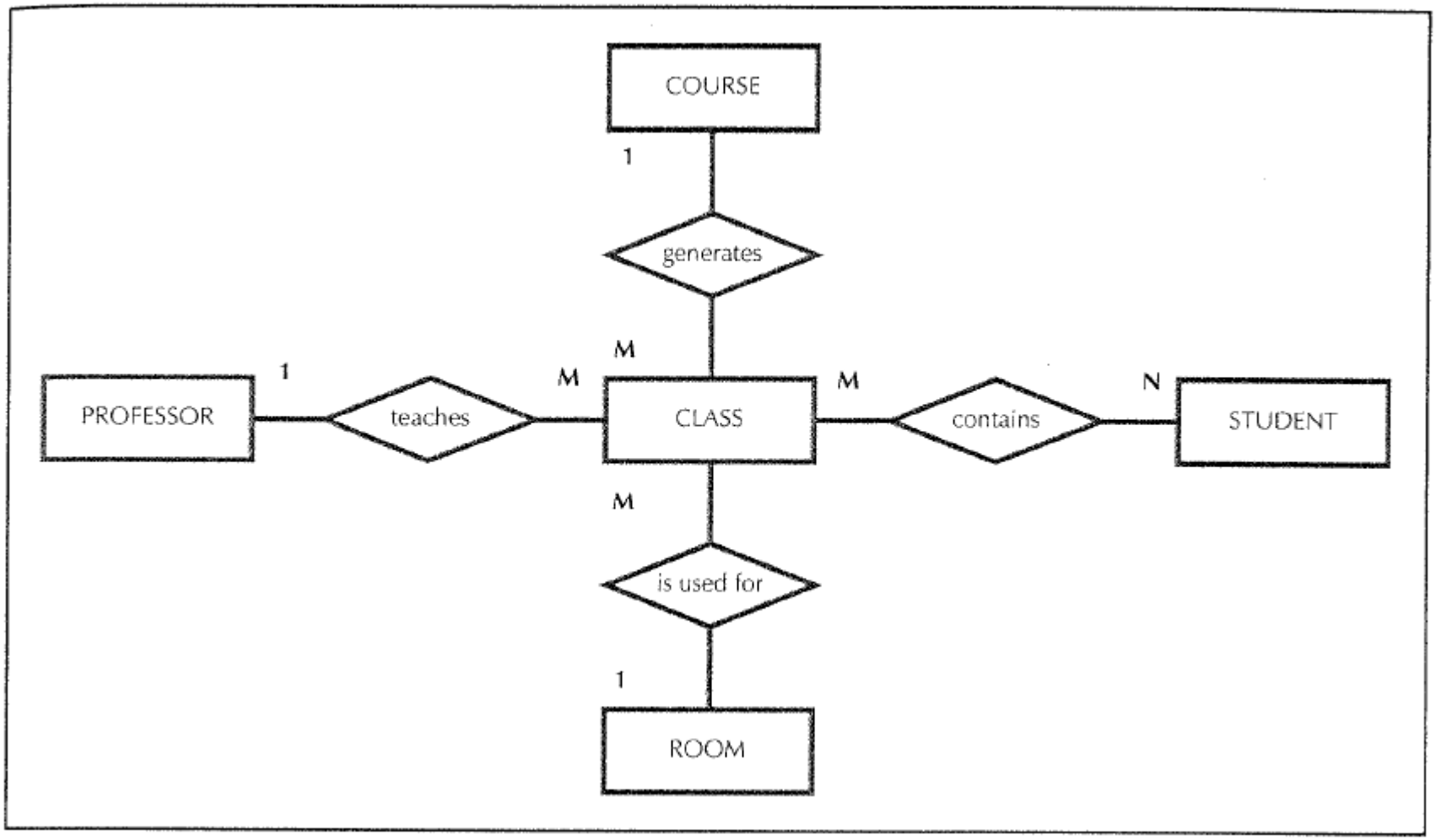
Tipos de Relaciones

- Uno a uno (Ej. Depto. tienen Jefe de Depto.)
- Muchos a uno (Ej. Vehículos pertenecen a propietarios)
- Muchos a muchos (Ej. Países exportan productos)



Nota: Tomar en cuenta la posibilidad de ninguno (cero)

Diagramas E-R de Chen



Atributos

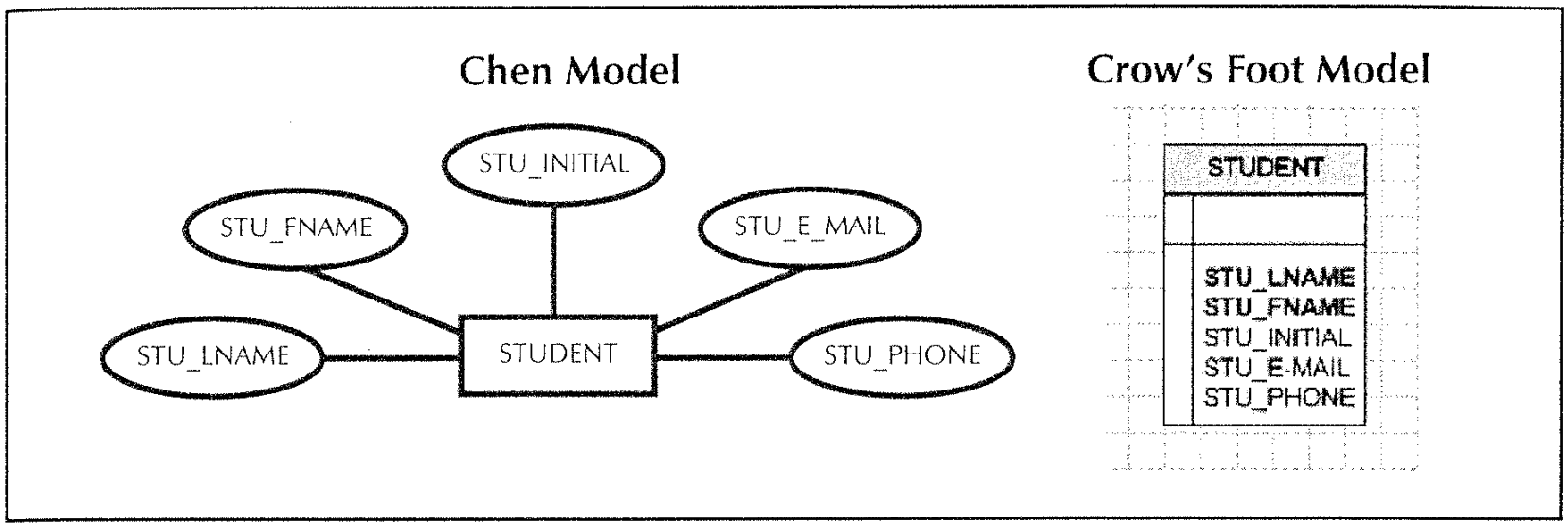
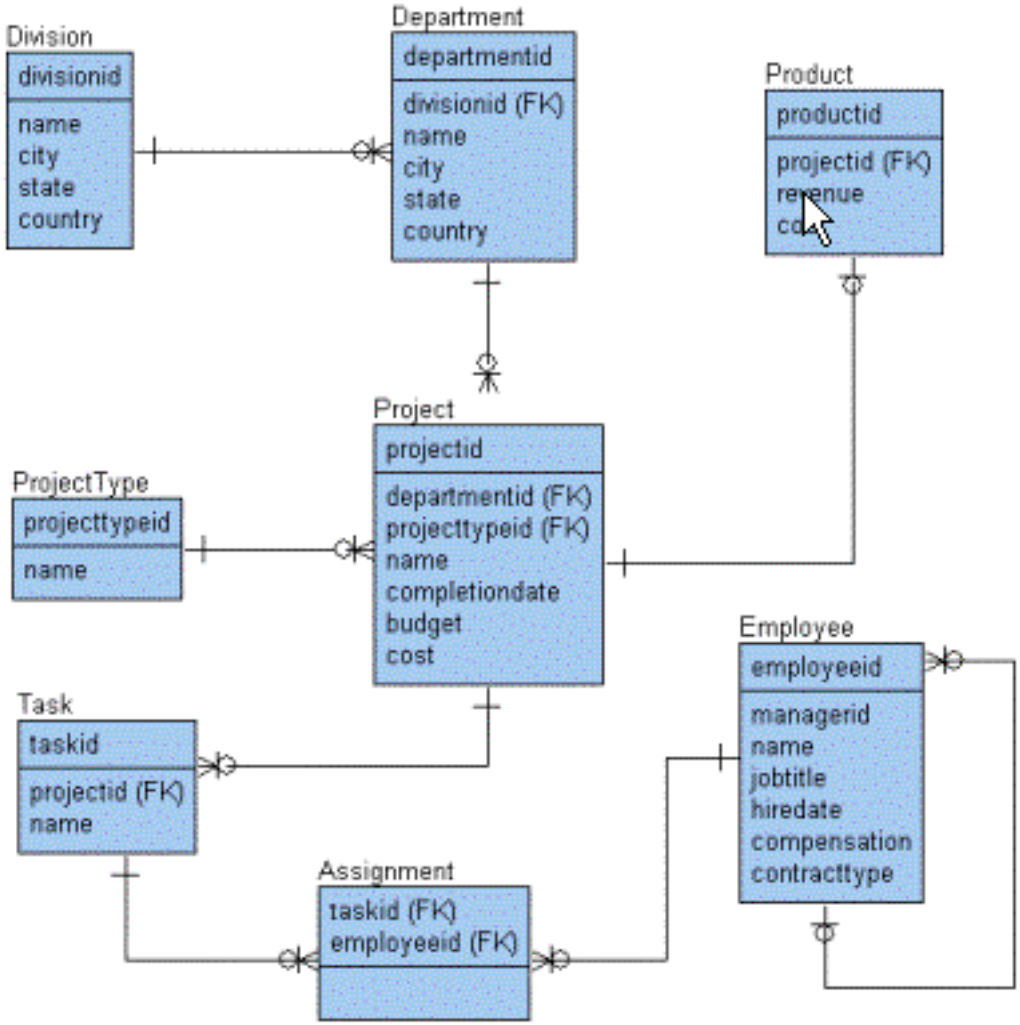
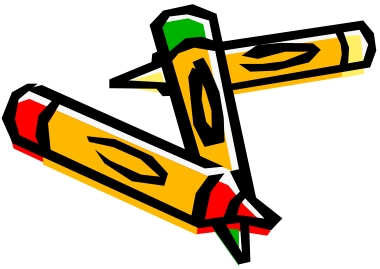

Chen Model
STU_INITIAL
STU_FNAME
STU_E_MAIL
STU_LNAME
STUDENT
STU_PHONE
Crow's Foot Model
STUDENT
STU_LNAME
STU_FNAME
STU_INITIAL
STU_E-MAIL
STU_PHONE

Diagrama E-R tipo Crow's Foot
Base de datos empleados



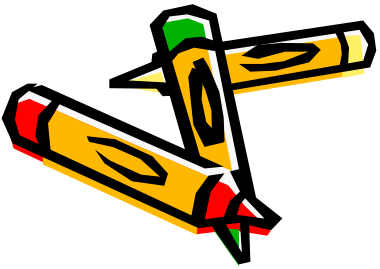
Atributos compuestos

- Clave electoral CMIBAN64071116H800
- Alfa_clave_electoral CMIBAN
- Fecha_nacimiento 640711
- Lugar_nacimiento 16
- Sexo H
- Digito_verificador 8
- Clave_homonimia 00

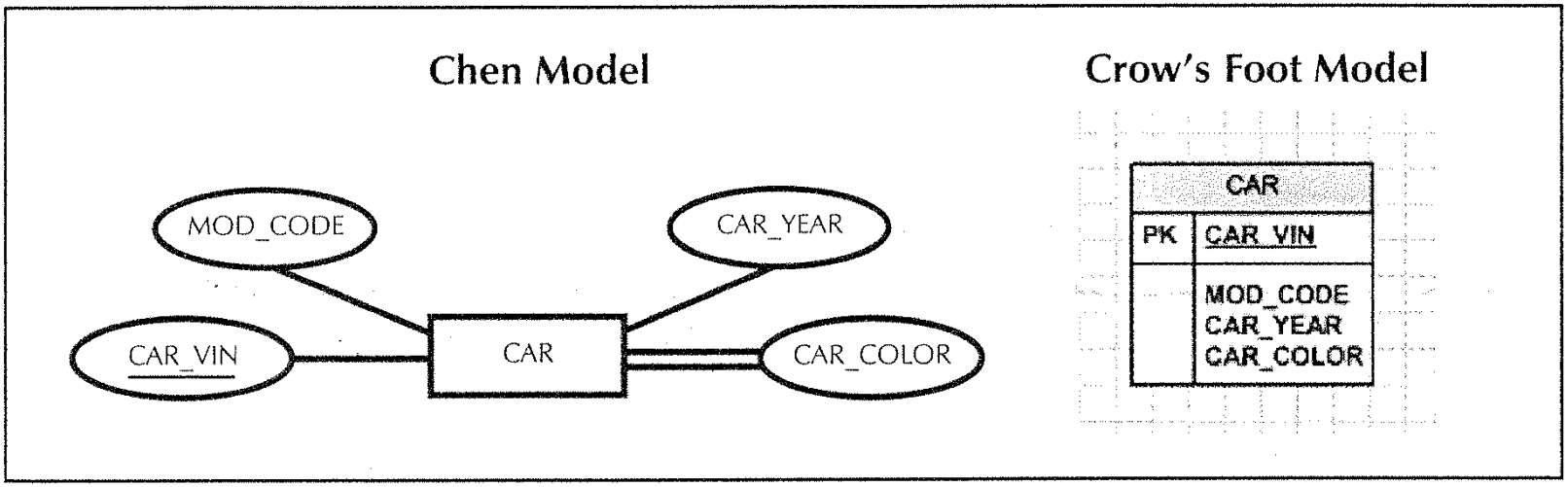


Atributos

- Multivaluados. Un atributo que puede tomar varios valores. Ej un Libro puede tener varios autores
- Monovaluados. Un atributo que puede tomar un solo valor para una entidad. Ej Un libro puede tener un solo título

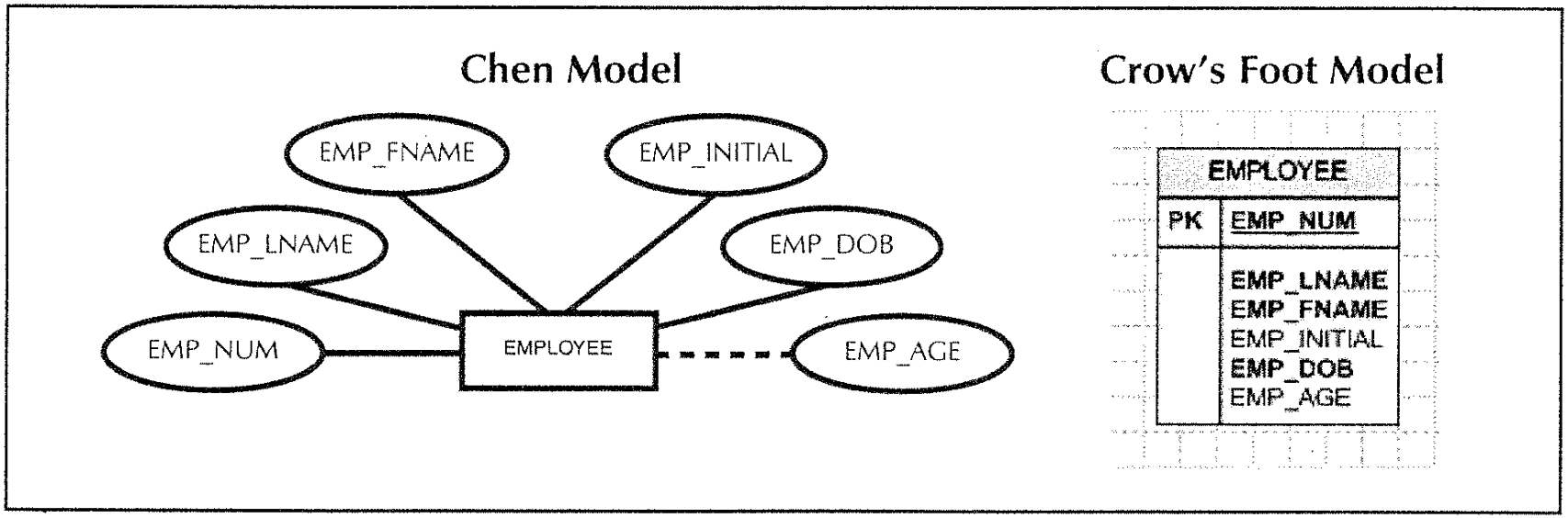


Atributos monovaluados y multivaluados



Atributos derivados

Son atributos que se pueden determinar de otros atributos.
El modelo Crow's Foot no tiene una representación especial
Para atributos derivados ni para atributos multivaluados



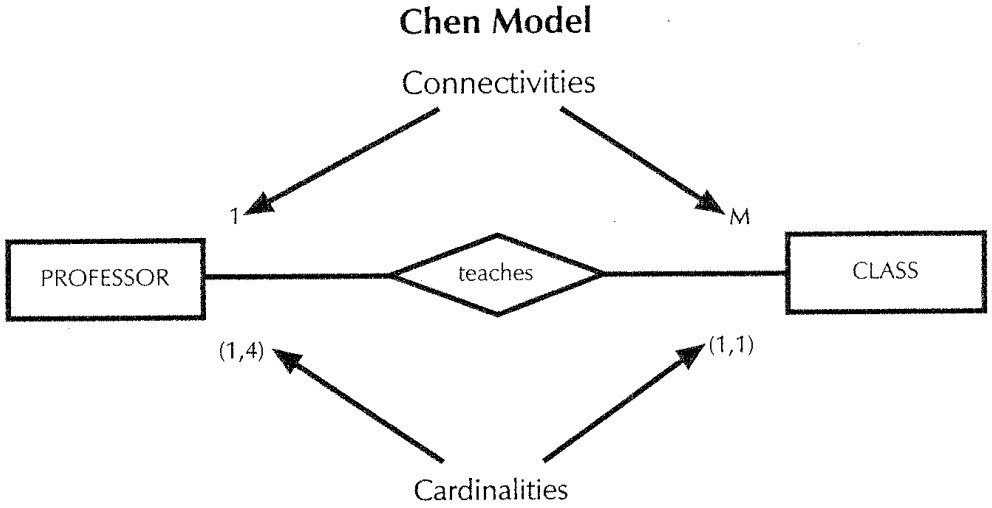
Cardinalidad

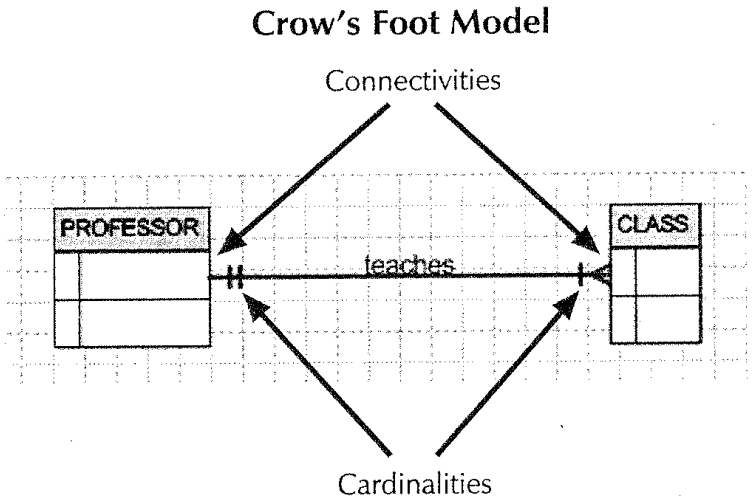
Cero o más

Uno o más

Uno y solo uno

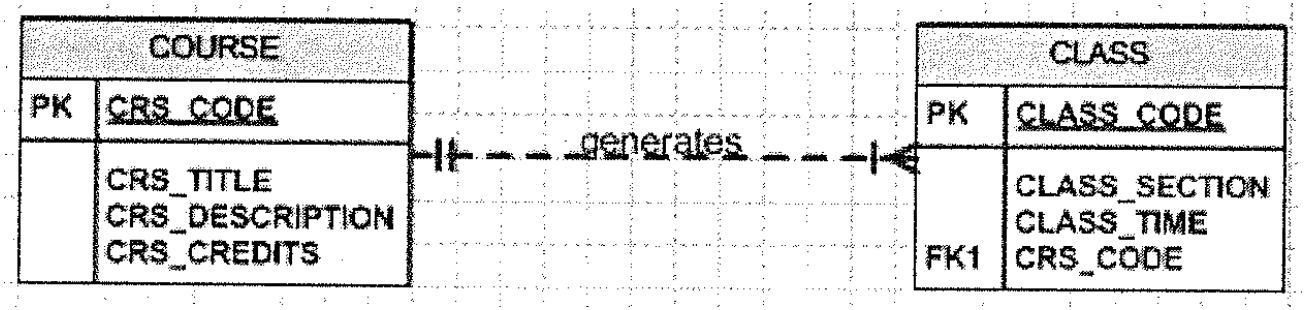
Cero o uno



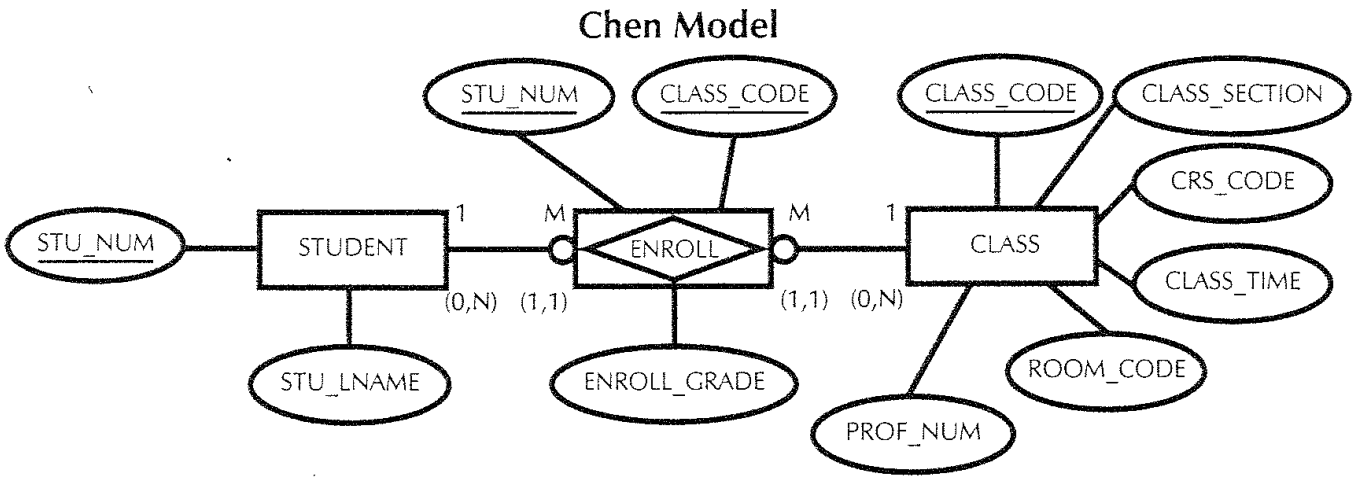


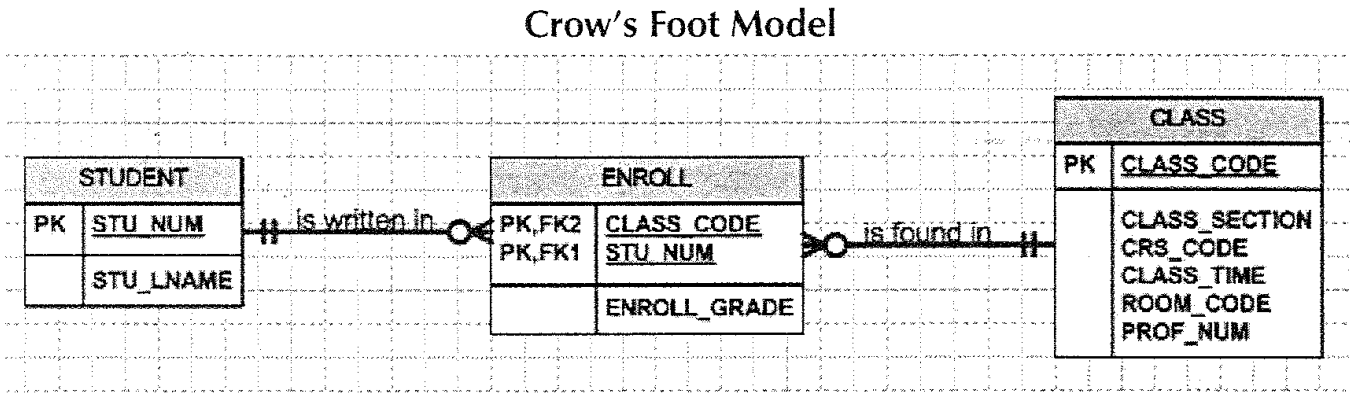
Relaciones débiles o fuertes

- Débil cuando la existencia de una entidad en un conjunto de entidades no esta condicionada a la existencia de otra entidad en el conjunto de entidades relacionado
- Fuerte en caso contrario

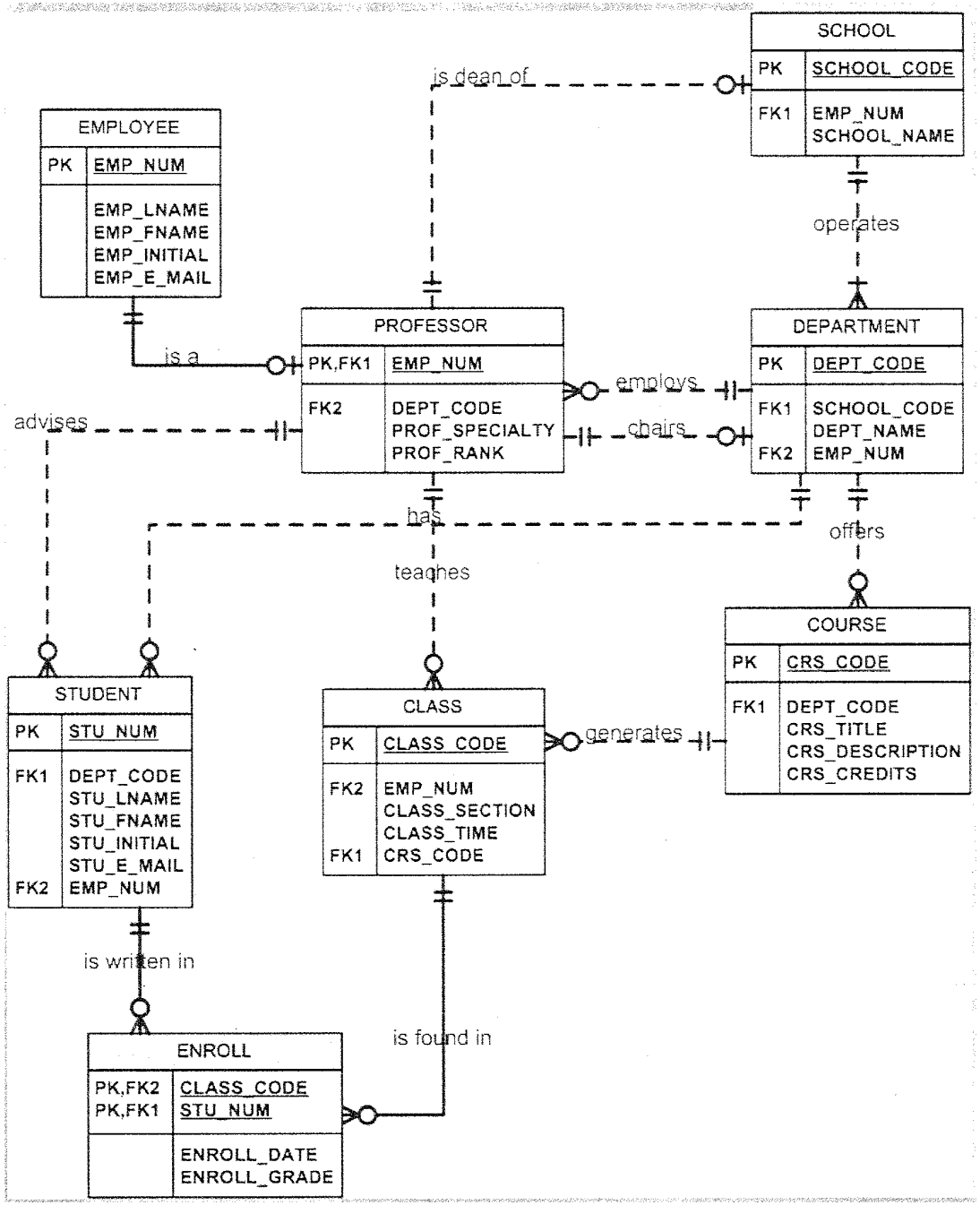


Implementando relaciones M:N



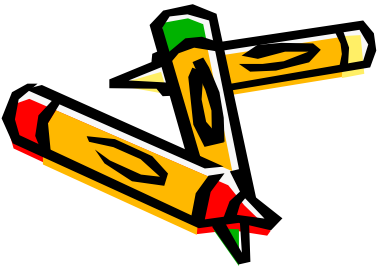


BD Escuela


SCHOOL
PK SCHOOL_CODE
FK1 EMP_NUM
SCHOOL_NAME
is dean of
operates
EMPLOYEE
PK EMP_NUM
EMP_LNAME
EMP_FNAME
EMP_INITIAL
EMP_E_MAIL
is a
PROFESSOR
PK,FK1 EMP_NUM
FK2 DEPT_CODE
PROF_SPECIALTY
PROF_RANK
employs
chairs
DEPARTMENT
PK DEPT_CODE
FK1 SCHOOL_CODE
DEPT_NAME
FK2 EMP_NUM
advises
has
offers
teaches
COURSE
PK CRS_CODE
FK1 DEPT_CODE
CRS_TITLE
CRS_DESCRIPTION
CRS_CREDITS
STUDENT
PK STU_NUM
FK1 DEPT_CODE
STU_LNAME
STU_FNAME
STU_INITIAL
STU_E_MAIL
FK2 EMP_NUM
CLASS
PK CLASS_CODE
FK2 EMP_NUM
CLASS_SECTION
CLASS_TIME
FK1 CRS_CODE
generates
is written in
is found in
ENROLL
PK,FK2 CLASS_CODE
PK,FK1 STU_NUM
ENROLL_DATE
ENROLL_GRADE

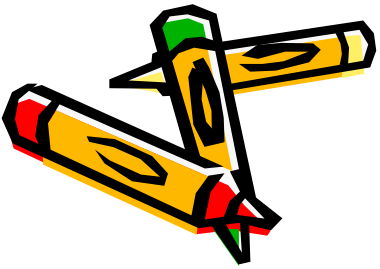
Problemas originados por un mal diseño de BD

- Ej. PROVEEDORES(NomProveedor,DomProveedor,Artículo,precio)
- **Redundancia**
- **Inconsistencia potencial**
- **Anomalías de Inserción** (Ej. No se tiene información de un proveedor hasta que se le compra algo)
- **Anomalías de Borrado** (Ej. Al eliminar registros correspondientes a artículos de determinado proveedor podemos sin intención borrar la información relativa a dicho proveedor)



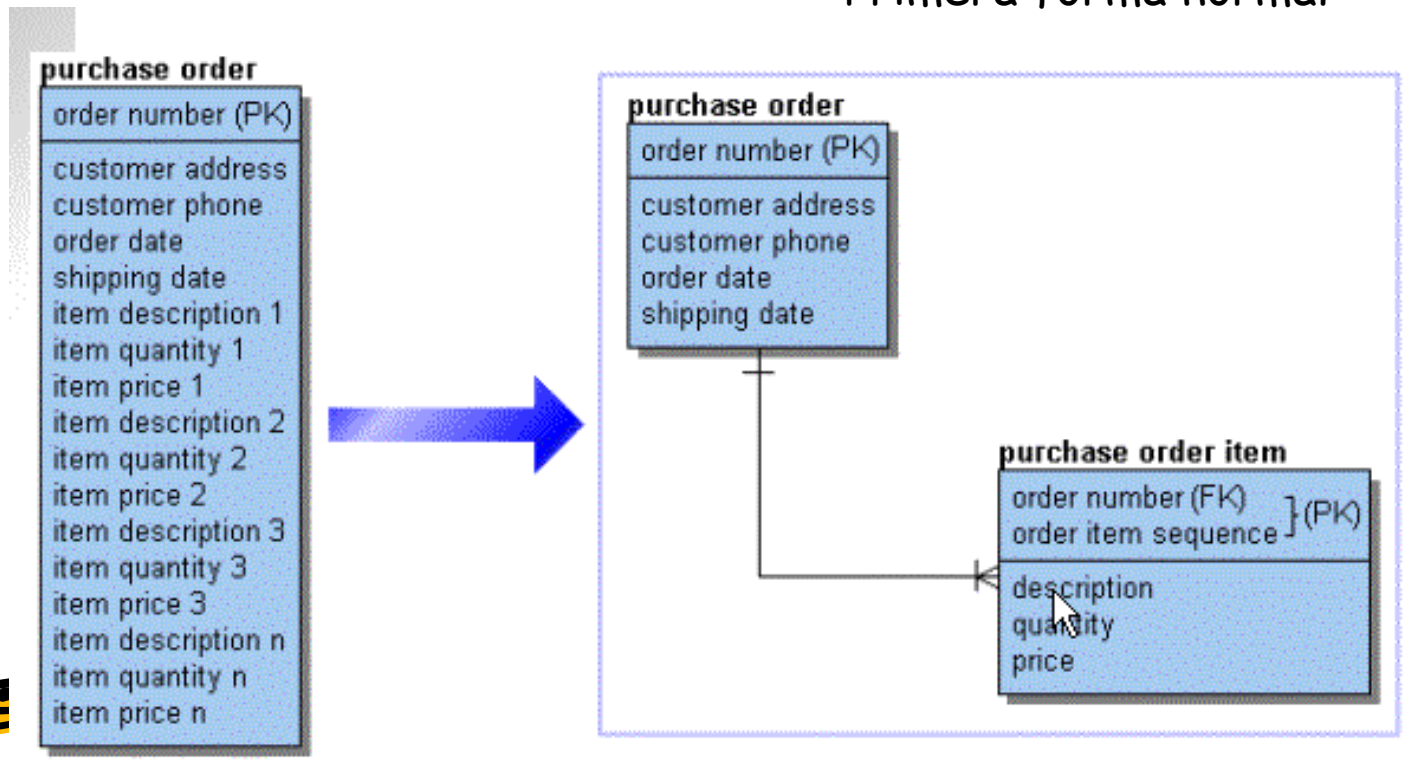
Restricciones

- Restricciones que dependen de la semántica del dominio de los elementos (ej. Nadie tiene 10m de estatura, Nadie de 25 años de edad tiene 35 años de experiencia profesional)
- Restricciones relacionadas a DEPENDENCIAS FUNCIONALES, las cuales son declaradas por el diseñador de la BD que pretenden garantizar su integridad



Normalización de Bases de Datos

Primera forma normal



Conversión a 1NF

Primer Paso: Eliminación de grupos repetitivos

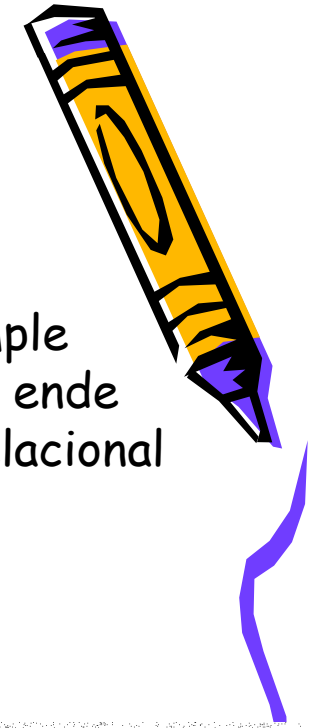


Table name: RPT_FORMAT — Database name: Ch05_ConstructCo

PROJ_NUM	PROJ_NAME	EMP_NUM	EMP_NAME	JOB_CLASS	CHG_HOUR	HOURS
15	Evergreen	103	June E. Arbough	Elect. Engineer	$84.50	23.8
		101	John G. News	Database Designer	$105.00	19.4
		105	Alice K. Johnson *	Database Designer	$105.00	35.7
		106	William Smithfield	Programmer	$35.75	12.6
		102	David H. Senior	Systems Analyst	$96.75	23.8
18	Amber Wave	114	Annelise Jones	Applications Designer	$48.10	24.6
		118	James J. Frommer	General Support	$18.36	45.3
		104	Anne K. Ramoras *	Systems Analyst	$96.75	32.4
		112	Darlene M. Smithson	DSS Analyst	$45.95	44.0
22	Rolling Tide	105	Alice K. Johnson	Database Designer	$105.00	64.7
		104	Anne K. Ramoras	Systems Analyst	$96.75	48.4
		113	Delbert K. Joenbrood *	Applications Designer	$48.10	23.6
		111	Geoff B. Wabash	Clerical Support	$26.87	22.0
		106	William Smithfield	Programmer	$35.75	12.8
25	Starflight	107	Maria D. Alonzo	Programmer	$35.75	24.6
		115	Travis B. Bawangi	Systems Analyst	$96.75	45.8
		101	John G. News *	Database Designer	$	
		114	Annelise Jones	Applications Designer		
		108	Ralph B. Washington	Systems Analyst		
		118	James J. Frommer	General Support		
		112	Darlene M. Smithson	DSS Analyst		

Este tipo de tablas no cumple ni siquiera como 1NF y por ende no puede llamarse tabla relacional

Asegurarse que cada campo con nulo tenga el valor que corresponde a su "grupo"

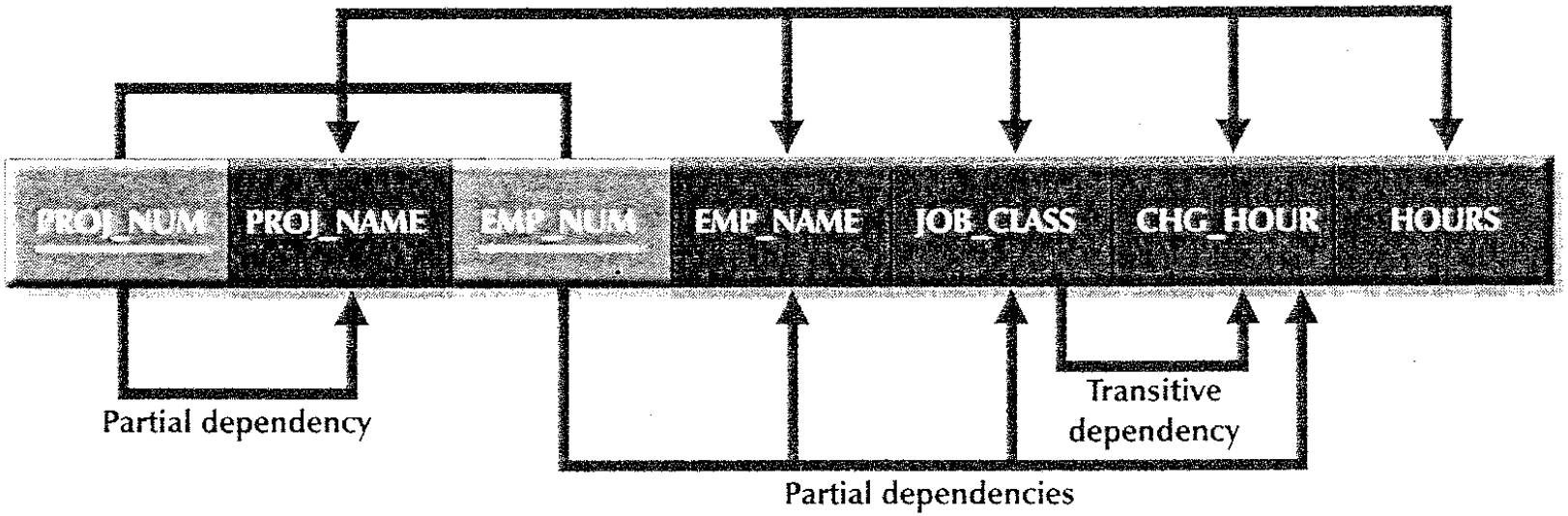
Table name: DATA_ORG_1NF — Database name: Ch05_ConstructCo

PROJ_NUM	PROJ_NAME	EMP_NUM	EMP_NAME	JOB_CLASS	CHG_HOUR	HOURS
15	Evergreen	103	June E. Arbough	Elect. Engineer	$84.50	23.8
15	Evergreen	101	John G. News	Database Designer	$105.00	19.4
15	Evergreen	105	Alice K. Johnson *	Database Designer	$105.00	35.7
15	Evergreen	106	William Smithfield	Programmer	$35.75	12.6
15	Evergreen	102	David H. Senior	Systems Analyst	$96.75	23.8
18	Amber Wave	114	Annelise Jones	Applications Designer	$48.10	24.6
18	Amber Wave	118	James J. Frommer	General Support	$18.36	45.3
18	Amber Wave	104	Anne K. Ramoras *	Systems Analyst	$96.75	32.4
18	Amber Wave	112	Darlene M. Smithson	DSS Analyst	$45.95	44.0
22	Rolling Tide	105	Alice K. Johnson	Database Designer	$105.00	64.7
22	Rolling Tide	104	Anne K. Ramoras	Systems Analyst	$96.75	48.4
22	Rolling Tide	113	Delbert K. Joenbrood *	Applications Designer	$48.10	23.6
22	Rolling Tide	111	Geoff B. Wabash	Clerical Support	$26.87	22.0
22	Rolling Tide	106	William Smithfield	Programmer	$35.75	12.8
25	Starflight	107	Maria D. Alonzo	Programmer	$35.75	24.6
25	Starflight	115	Travis B. Bawangi	Systems Analyst	$96.75	45.8
25	Starflight	101	John G. News *	Database Designer	$105.00	56.3
25	Starflight	114	Annelise Jones	Applications Designer	$48.10	33.1
25	Starflight	108	Ralph B. Washington	Systems Analyst	$96.75	23.6
25	Starflight	118	James J. Frommer	General Support	$18.36	30.5
25	Starflight	112	Darlene M. Smithson	DSS Analyst	$45.95	41.4

Conversion a la 1NF

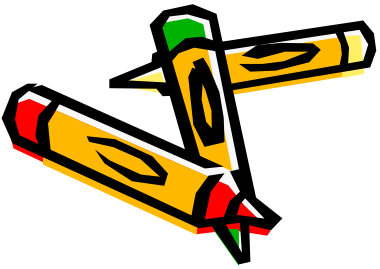
- Segundo paso, identificar la llave primaria, en este caso PROJ_NUM, EMP_NUM
- Tercer paso. Identificar las dependencias. Decir que el atributo B es funcionalmente dependiente del atributo A es equivalente a decir que **A identifica a B**

$$A \rightarrow B$$



La primera forma normal implica

- Todas las claves están definidas
- No hay grupos repetitivos en una tabla
- Todos los atributos dependen de la clave primaria



Conversión a la 2NF

- **Primer paso**: Identificar todas las llaves y todas sus combinaciones, cada llave y cada combinación de llaves constituye la llave primaria de una nueva tabla.

Ej.

PROJ_NUM

EMP_NUM

PROJ_NUM EMP_NUM

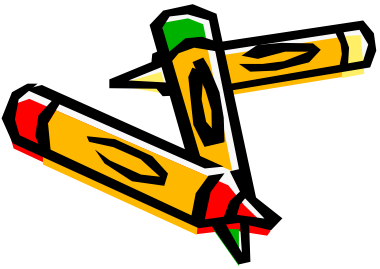
- **Segundo paso**: Identificar los atributos que dependen de cada llave o combinación de llaves y generar las tablas nuevas

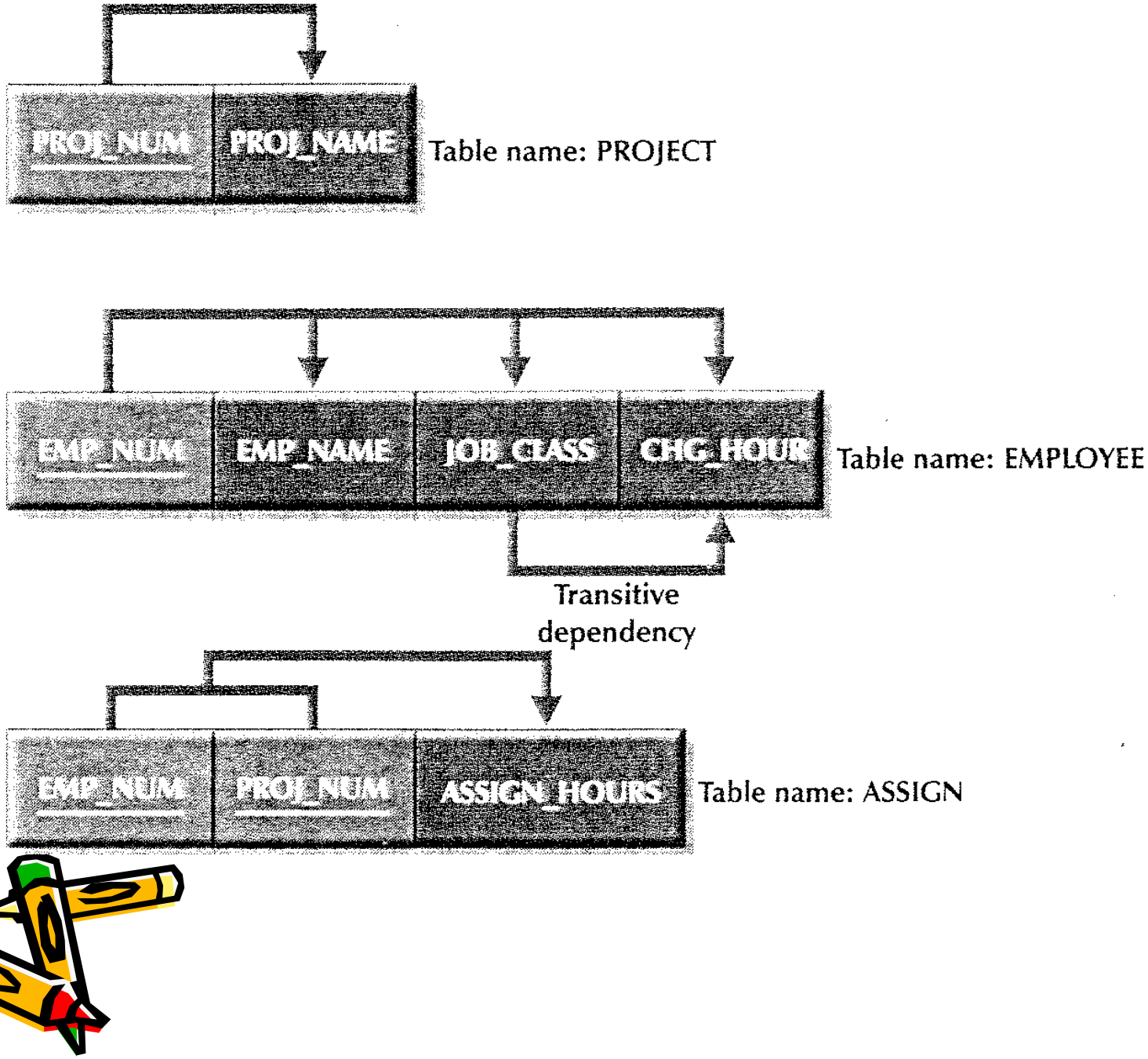
Ej.

PROJECT(**PROJ_NUM**, PROJ_NAME)

EMPLOYEE(**EMP_NUM**, EMP_NAME, JOB_CLASS, CHG_HOUR)

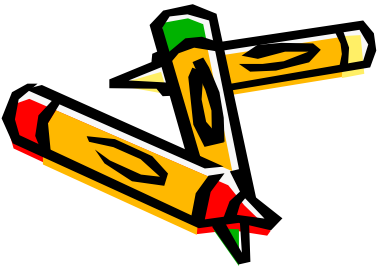
ASSIGN(**PROJ_NUM, EMP_NUM**, ASSIGN_HOURS)




PROJ_NUM
PROJ_NAME
Table name: PROJECT
EMP_NUM
EMP_NAME
JOB_CLASS
CHG_HOUR
Table name: EMPLOYEE
Transitive dependency
EMP_NUM
PROJ_NUM
ASSIGN_HOURS
Table name: ASSIGN

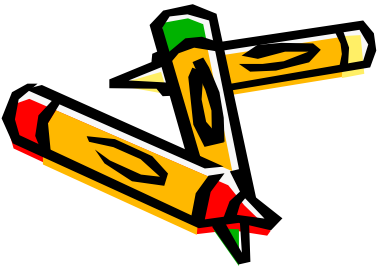
La segunda forma normal

- La 2NF implica que no hay dependencias parciales, es decir, ningún atributo depende solamente de una parte de la llave primaria
- Si la llave primaria no es llave compuesta, la tabla automáticamente está en la 2NF



Una tabla se encuentra en la 3NF si:

- Cumple con la 2NF
- Cada atributo depende directamente de la llave primaria (no de manera transitiva)



Tercera Forma Normal (3NF)

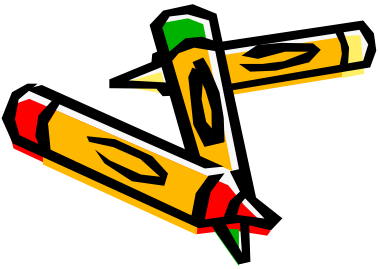
- Primer paso: Por cada dependencia transitiva, use al determinante como llave primaria de una tabla nueva

JOB_CLASS

- Segundo paso: Identificar los atributos dependientes de cada determinante identificado en el primer paso

JOB_CLASS -> CHG_HOUR

- Elimine los atributos dependientes de las dependencias transitivas



3NF

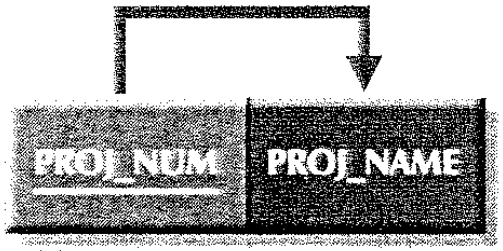


Table name: PROJECT

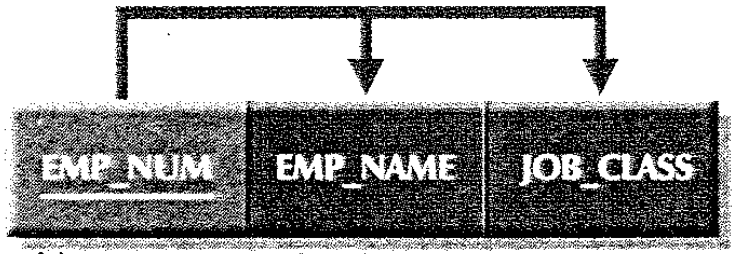


Table name: EMPLOYEE

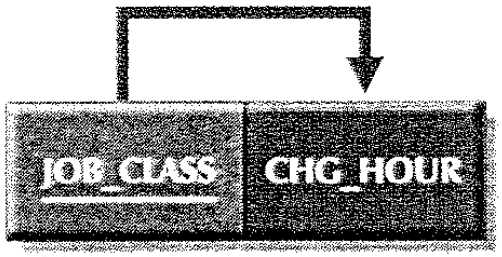


Table name: JOB

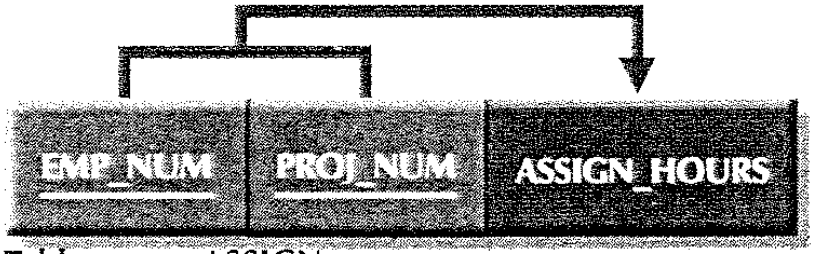
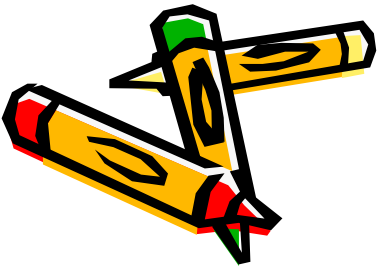


Table name: ASSIGN

La 3NF presenta problemas potenciales cuando:

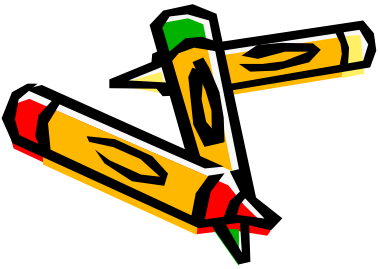
- La relación tiene varias llaves candidatas
- Las llaves candidatas son compuestas
- Las llaves candidatas se traslapan (tienen al menos un atributo en común)

Recordar que una llave candidata es una llave que tiene todo lo necesario para ser llave primaria excepto que no lo es

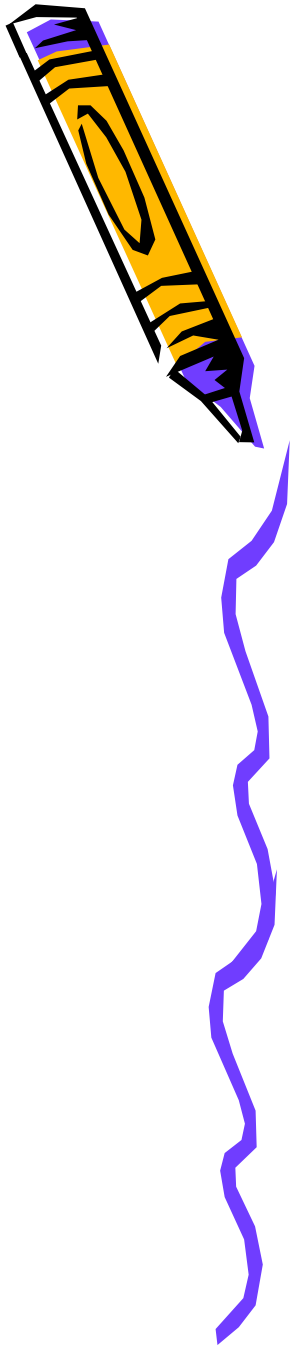


Forma normal de Boyce-Codd (BCNF)

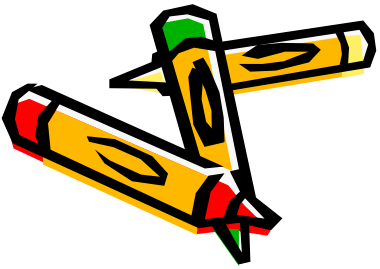
- Una relación está en BCNF si y solo si cada determinante es una llave candidata. Esto implica que en un diagrama de dependencias funcionales las únicas flechas que existen son aquellas que salen de atributos que forman parte de llaves candidatas (no solo de las llaves primarias)
- Como consecuencia de lo anterior, las llaves candidatas no tienen traslape (no comparten atributos)



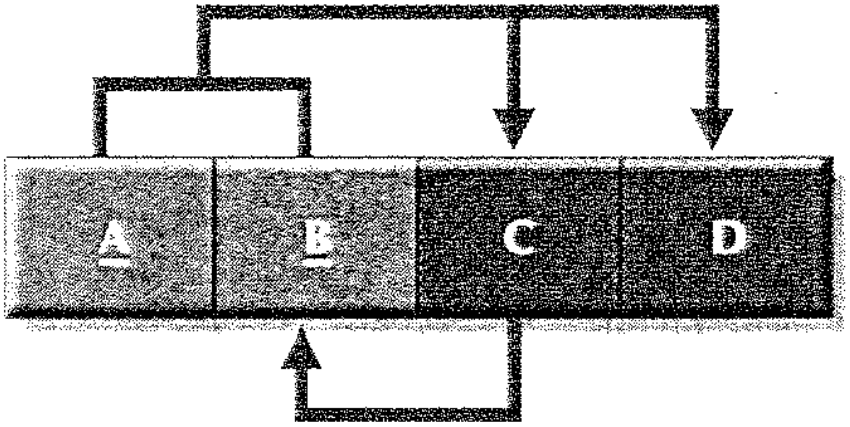
Acerca de la BCNF



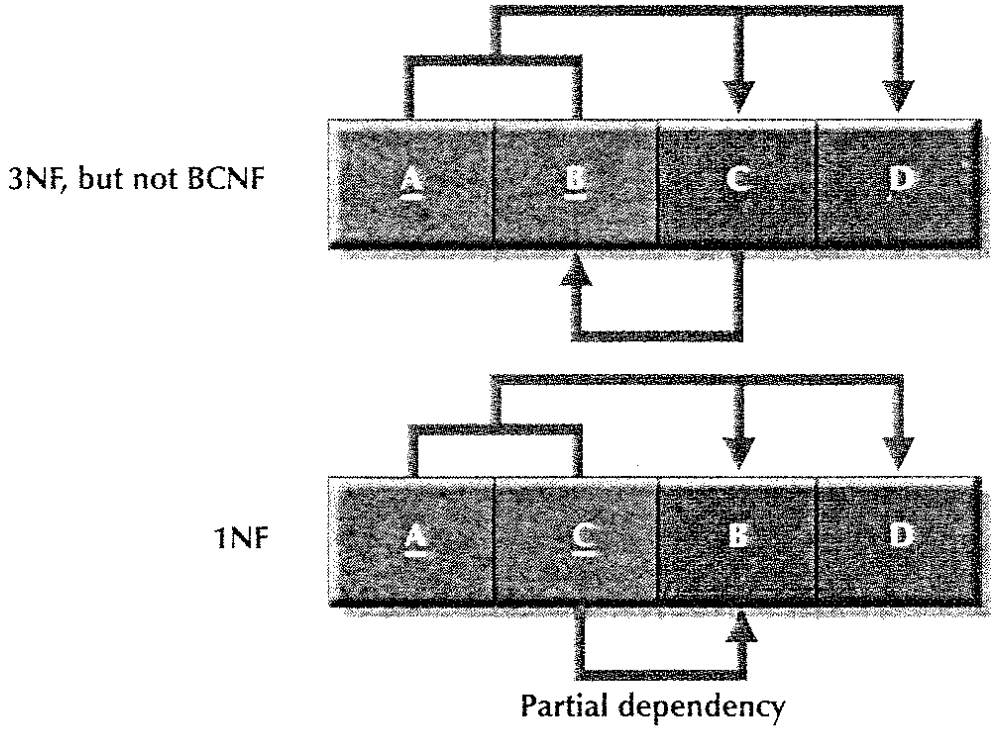
- Si una relación tiene solo una llave candidata, la 3NF y la BCNF son equivalentes
- Las llaves candidatas traslapadas no siempre ocasionan problemas
- BCNF es mas simple que la 3NF en el sentido de que su definición no hace referencia a la 2NF ni a la 1NF

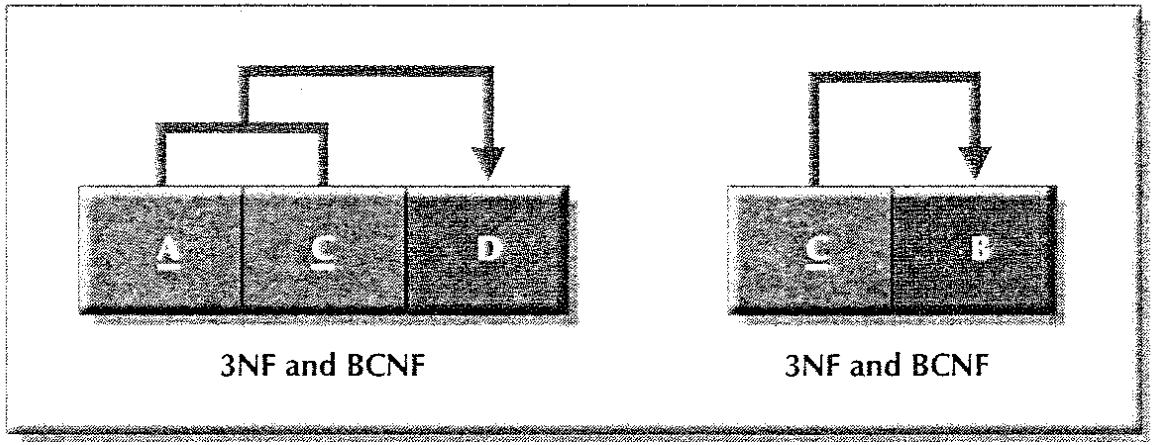


Ejemplo de relación que cumple 3NF pero no BCNF



Para convertir esta relación a la BCNF, se debe elegir A+C como Llave primaria, luego convertir a 2NF y luego a 3NF


3NF, but not BCNF
A
B
C
D
1NF
A
C
B
D
Partial dependency


A
C
D
3NF and BCNF
C
B
3NF and BCNF

Para cada estudiante, una materia se la imparte un solo profesor.

Un profesor solo imparte una materia pero una materia la pueden
impartir diferentes profesores

La relación SJT(S,J,T) tiene las dependencias funcionales:
(S,J)→T y T→J
Y las llaves candidatas traslapadas (S,J) y (S,T)
Por lo tanto cumple 3NF pero no cumple BCNF

S	J	T
Smith	Math	Prof White
Smith	Physics	Prof Green
Jones	Math	Prof White
Jones	Physics	Prof Brown

Si reemplazamos la relación SJT(S,J,T) por las dos relaciones ST(S,T) y TJ(T,J) ambas estarán en BCNF

S	T
Smith	Prof White
Smith	Prof Green
Jones	Prof White
Jones	Prof Brown

T	J
Prof White	Math
Prof Green	Physics
Prof Brown	Physics

EL problema ahora es que las 2 relaciones no se pueden actualizar de manera independiente

Dependencias multivaluadas

Si el atributo A de una relación R multi-determina a un atributo B de la misma relación R, escribimos R.A->>R.B

Curso ->> Profesor
Curso ->> Texto

O bien: Curso ->> Profesor|Texto

debido a que dado un curso, podemos Listar al conjunto de profesores que Lo pueden impartir y conjunto de Textos que se usan en ese curso

Curso	Profesor	Texto
Física	Green	Basic Mechanics
Física	Green	Principles of Optics
Física	Brown	Basic Mechanics
Física	Brown	Principles of Optics
Mate	Green	Basic Mechanics
Mate	Green	Vector analysis
Mate	Green	Trigonometry

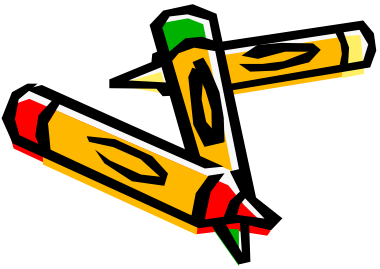
Toda dependencia funcional es una dependencia mutivaluada pero no al reves

Dos definiciones para 4NF

- Una relación R esta en la 4NF si para cualquier dependencia multivaluada A->>B, todos los atributos de R son funcionalmente dependientes de A
- De otra manera, Una relación R esta en la 4NF si está en la BCNF y todas sus dependencias multivaluadas son en realidad dependencias funcionales

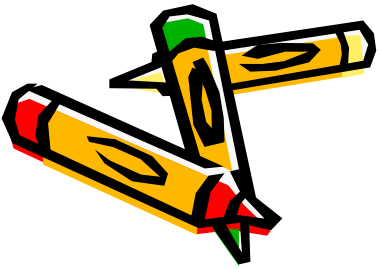
Teorema de Fagin:
Una relacion R(A,B, C) se puede descomponer sin pérdidas en las Relaciones R1(A,B) y R2(A,C) si y solo si A->>B|C

La relación R(Curso,Profesor,Texto) no cumple 4NF
Pero las dos relaciones R1(Curso,Profesor) y R2(Curso,Texto)
Si lo estám



Descomponibilidad

- No todas las relaciones se pueden descomponer sin perdidas en 2 relaciones
- Decimos que una relación es n-descomponible si se puede descomponer sin pérdida en n de sus proyecciones pero no en un número menor de sus proyecciones (Aho, Beery & Ullman)



S	P	J
S1	P1	J2
S1	P2	J1
S2	P1	J1
S1	P1	J1

SPJ se 3-descompone en:

S	P
S1	P1
S1	P2
S2	P1

P	J
P1	J2
P2	J1
P1	J1

J	S
J2	S1
J1	S1
J1	S2

Junta de SP Y PJ

S	P	J
S1	P1	J2
S1	P1	J1
S1	P2	J1
S2	P1	J2
S2	P1	J1

Espurio →

Junta de SP,PJ y JS

S	P	J
S1	P1	J2
S1	P2	J1
S2	P1	J1
S1	P1	J1

Restricción cíclica 3D

- La función de la relación JS es la de eliminar los tuples espurios de la junta de la relación SP con PJ, esto se puede expresar en términos de una restricción cíclica:

If (S1,P1) aparece en la relación SP
AND (P1,J1) aparece en la relación PJ
AND (J1,S1) aparece en la relación JS
THEN (S1,P1,J1) aparece en la relación SPJ

Una relación R es n-descomponible si y solo si satisface una relación cíclica similar en donde participen n proyecciones de R

Se conoce como restricción 3D por que representa una restricción del "mundo real", por ej.

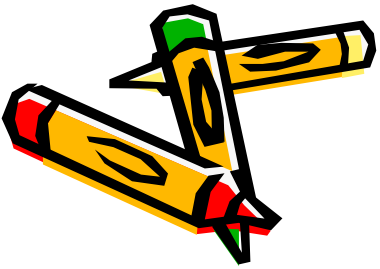
a) "Arrendadora Gómez" renta "gruas" y
b) "Gruas" son usadas para el proyecto "Planta nuclear"
c) "Arrendadora Gómez" es proveedor del proyecto "Planta nuclear"

Se concluye que:

d) Arrendadora Gomez provee gruas para el proyecto planta nuclear

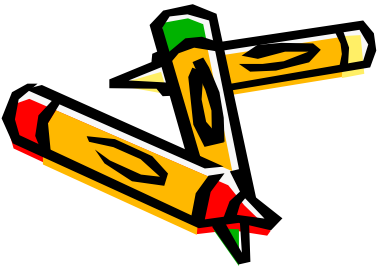
Dependencia conjunta

- La relación R satisface la dependencia conjunta *(X,Y,…,Z) si y solo si R es igual a la junta natural de sus proyecciones X,Y,…,Z los cuales son subconjuntos de atributos de R. Por ejemplo la relación SPJ cumple la dependencia conjunta *(SP, PJ,JS)



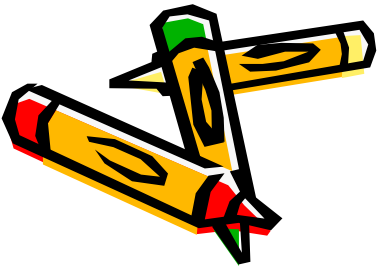
Dependencia conjunta como generalización de dependencia multivaluada

- Teorema de Fagin: R(A,B,C) se puede descomponer en R1(A,B) y R2(A,C) si y solo si A->>B|C
- Teorema d Fagin: R(A,B,C) satisface la dependencia conjunta *(AB,AC) si y solo si R(A,B,C) se puede descomponer sin pérdidas en R1(A,B) y R2(A,C)
- Entonces podemos ver a la dependencia multivaluada como un caso particular de la dependencia conjunta



5NF ó PJ/NF (Projection-Join Normal Form)

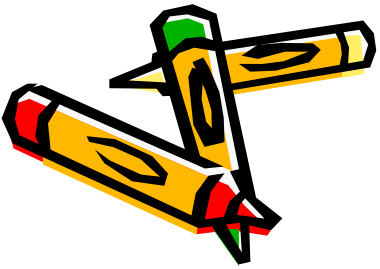
- La relación R esta en la 5NF si y solo si cada dependencia conjunta de R es consecuencia de las llaves candidatas de R
- Ej. La Rel (S#,Sname,Status,City) tiene la dependencia conjunta *((S#,Sname,Status), (S#,City)) la cual está implicada por el hecho de que S# es una llave candidata
- Fagin proporciona un algoritmo para verificar si una dependencia conjunta está implicada por una determinada llave candidata



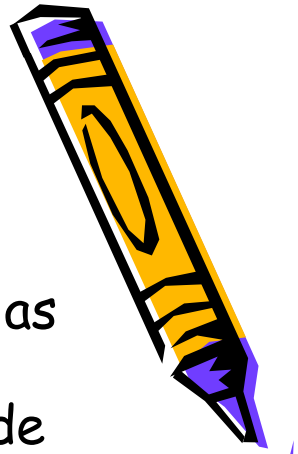
La 5NF es en realidad de interés meramente académico

La Normalización por si sola no garantiza un buen diseño de la BD

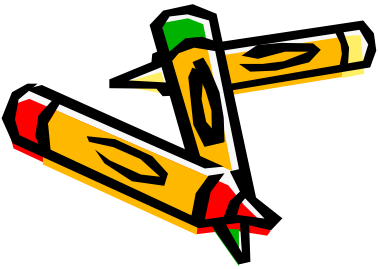
- Asignación de llave primaria sintética
- Nombres de atributos convencionales (Ej. Agregar como prefijo el nombre de la tabla)
- Atributos atómicos (Ej en lugar de nombre_completo es mejor paterno, materno y nombres)
- Añadir atributos
- Facilitar precisión histórica
- Uso de atributos derivados puede ahorrar tiempo al generar reportes



Denormalización

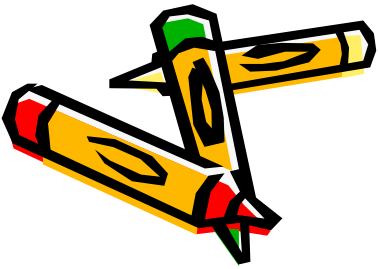


- Al aumentar el nivel de normalización, el número de tablas aumenta
- Recuperar la información de tablas conjuntas requiere de mas operaciones de disco
- Cierto grado de Denormalización puede entonces incrementar la velocidad de procesamiento de datos
- La normalización previene anomalías de inserción, borrado y actualización de datos
- Encontrar el equilibrio entre velocidad y anomalías
- Algunas anomalías son solo de interés académico, por ejemplo, ¿Es realmente preocupante que el atributo codigo_postal sea determinante del atributo ciudad en una tabla de Clientes donde la llave primaria es num_cliente?. ¿Realmente necesitamos la tabla Cod(cod_postal,ciudad) para eliminar una dependencia transitiva?

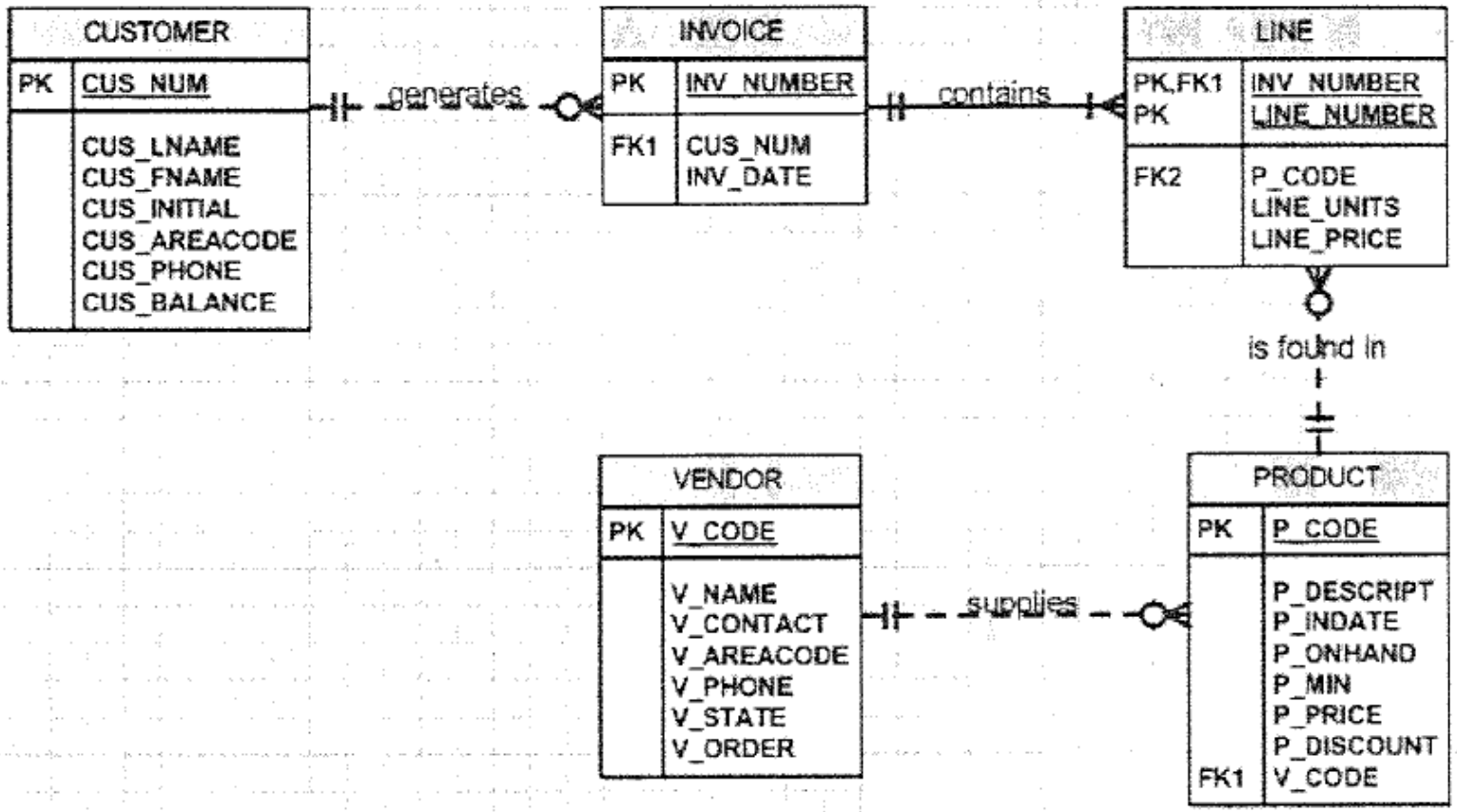


Denormalización (Cont)

- Un alto grado de normalización es difícil de sostener
- En aplicaciones como Data warehousing es común utilizar la 2NF
- De cualquier forma las tablas no normalizadas deben justificarse en cada caso

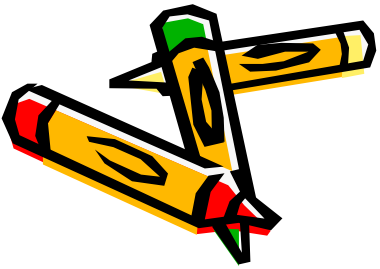


Esquema para ejemplos de SQL



Para crear el esquema

- Create schema authorization <usuario>;
- Este comando rara vez se usa ya que al crear un usuario, el DBMS automáticamente le crea un esquema
- Por lo tanto nos concentramos en la creación y manipulación de las tablas
- Para crear el esquema de la lámina anterior correr el script CH06DBINIT.SQL



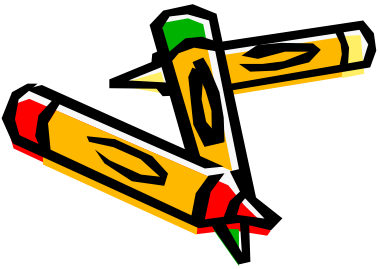
Restricciones (constraints)

Con create table se pueden poner restricciones como parte de La definición de una columna o en la cláusula ***constraint***

- Not null
- Check
- Unique
- Primary key
- References
- Default
- Las restricciones se pueden deshabilitar o habilitar con

Alter table …. enable/disable <nombre_de_restriccion>

Las restricciones a las que no les pongamos nombre, reciben un nombre que se puede averiguar en user_constraints



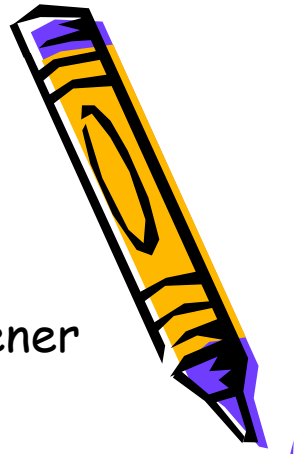
Primary key

- Si se usa esta restricción, entonces Not null y Unique son redundantes. Para llaves primarias compuestas usar la clausula Primary key al final de la sentencia create table, el orden en que se escriben los atributos ahí impacta el performance de la BD

```
CREATE TABLE LINE (
INV_NUMBER              NUMBER NOT NULL,
LINE_NUMBER             NUMBER(2,0) NOT NULL,
P_CODE                  VARCHAR(10) NOT NULL,
LINE_UNITS              NUMBER(9,2) DEFAULT 0.00 NOT NULL,
LINE_PRICE              NUMBER(9,2) DEFAULT 0.00 NOT NULL,
PRIMARY KEY (INV_NUMBER,LINE_NUMBER),
FOREIGN KEY (INV_NUMBER) REFERENCES INVOICE ON DELETE CASCADE,
FOREIGN KEY (P_CODE) REFERENCES PRODUCT(P_CODE),
CONSTRAINT LINE_UI1 UNIQUE(INV_NUMBER, P_CODE));
```

```
CREATE TABLE INVOICE (
INV_NUMBER              NUMBER PRIMARY KEY,
CUS_CODE  NUMBER NOT NULL REFERENCES CUSTOMER(CUS_CODE),
INV_DATE  DATE DEFAULT SYSDATE NOT NULL,
CONSTRAINT INV_CK1 CHECK (INV_DATE > TO_DATE('01-JAN-2002','DD-MON-
YYYY')));
```

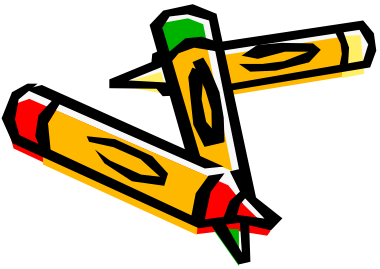
Foreign key



- Primero crear la tabla referida
- Si se usa ***on update cascade***, cualquier cambio en el atributo automáticamente se actualiza el atributo referido para mantener la "integridad referencial" (oracle10g no soporta esto)
- Si se usa ***on delete cascade***, el borrado del registro provoca borrado del registro referido
- Si la llave foránea es compuesta, en lugar de usar la restricción ***references*** se usa la restricción ***foreign key*** en la clausula ***constraint***

```
CREATE TABLE INVOICE (
INV_NUMBER              NUMBER PRIMARY KEY,
CUS_CODE  NUMBER NOT NULL REFERENCES CUSTOMER(CUS_CODE),
INV_DATE  DATE DEFAULT SYSDATE NOT NULL,
CONSTRAINT INV_CK1 CHECK (INV_DATE > TO_DATE('01-JAN-2002','DD-MON-YYYY')));
```

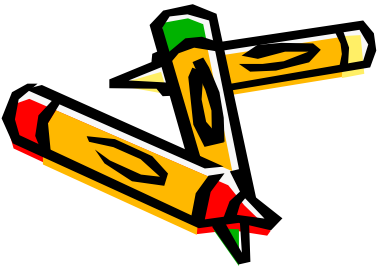
```
CREATE TABLE LINE (
INV_NUMBER              NUMBER NOT NULL,
LINE_NUMBER             NUMBER(2,0) NOT NULL,
P_CODE                  VARCHAR(10) NOT NULL,
LINE_UNITS              NUMBER(9,2) DEFAULT 0.00 NOT NULL,
LINE_PRICE              NUMBER(9,2) DEFAULT 0.00 NOT NULL,
PRIMARY KEY (INV_NUMBER,LINE_NUMBER),
FOREIGN KEY (INV_NUMBER) REFERENCES INVOICE ON DELETE CASCADE,
FOREIGN KEY (P_CODE) REFERENCES PRODUCT(P_CODE),
CONSTRAINT LINE_UI1 UNIQUE(INV_NUMBER, P_CODE));
```



check

```
CREATE TABLE INVOICE (
INV_NUMBER              NUMBER PRIMARY KEY,
CUS_CODE  NUMBER NOT NULL REFERENCES CUSTOMER(CUS_CODE),
INV_DATE  DATE DEFAULT SYSDATE NOT NULL,
CONSTRAINT INV_CK1 CHECK (INV_DATE > TO_DATE('01-JAN-2002','DD-MON-YYYY')));
```

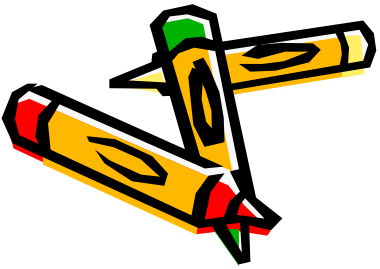
```
CREATE TABLE CUSTOMER (
CUS_CODE  NUMBER PRIMARY KEY,
CUS_LNAME               VARCHAR(15) NOT NULL,
CUS_FNAME               VARCHAR(15) NOT NULL,
CUS_INITIAL             CHAR(1),
CUS_AREACODE            CHAR(3) DEFAULT '615' NOT NULL CHECK(CUS_AREACODE IN ('615','713','931')),
CUS_PHONE               CHAR(8) NOT NULL,
CUS_BALANCE             NUMBER(9,2) DEFAULT 0.00,
CONSTRAINT CUS_UI1 UNIQUE(CUS_LNAME,CUS_FNAME));
```



default

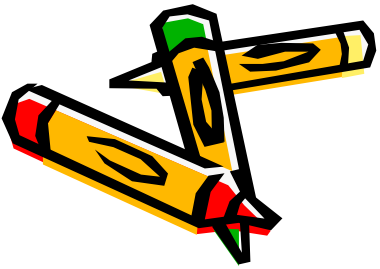
```
create table t1
(
   id$ integer not null,
   charcol char default 'Y',
   datecol date default sysdate,
   strcol varchar2(30) default user,
   intcol integer default 12
);
insert into t1 (id$) values (1);
select * from t1;
      ID$ C DATECOL   STRCOL                         INTCOL
--------- - --------- ------------------------------ ----------
        1 Y 28-MAY-04 SCOTT                          12
```

```
create table t2(charcol char default 'Y',datecol date default sysdate);
insert into t2 (charcol) values (default);
select * from t2;

C DATECOL
- ---------
Y 28-MAY-04
```

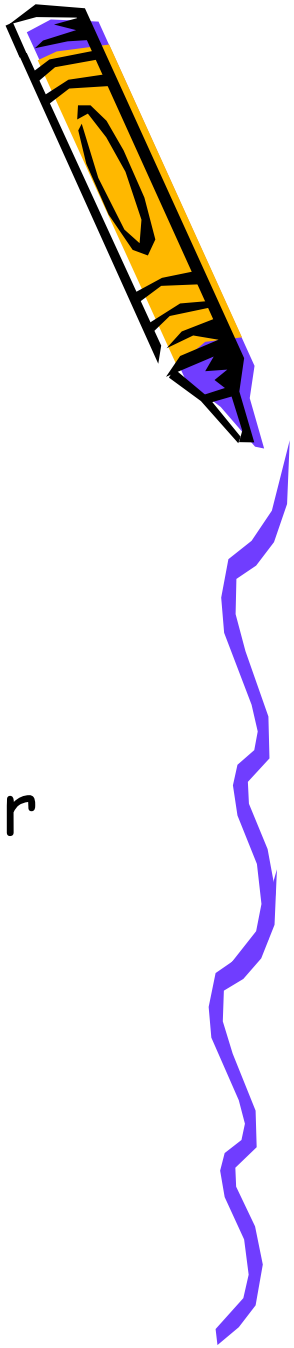


Indices

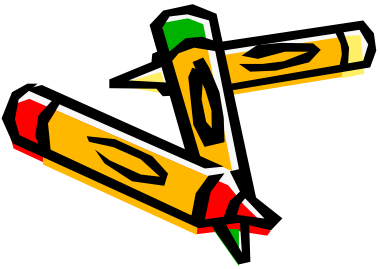
- Se pueden crear índices con característica de unicidad o no
- Create [unique] index <nombre_indice> on <nombre_tabla> (nom_col1,nom_col2,..) [tablespace <nombre_tabla_de_espacio> [storage [initial <int>] [next <int>] [pctincrease <int>] [minextents <int>] [maxextents <int>]]]
- Un índice funcionará para consultas que recuperen a menos de 20% de la tabla de lo contrario resulta contraproducente
- Alter index …
- Drop index



Comandos de manipulación de datos

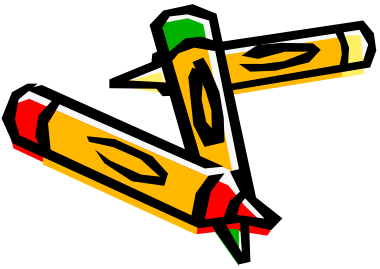


- Insert into <nombre_tabla> values (,,)
- Insert into <nombre_tabla> select …
- Insert into <nombre_tabla> values (..,null,..,);
- Para tablas con atributos con valores por defecto
- Insert into <nombre_tabla> (campo_oblig1,campo_oblig2) values (,)



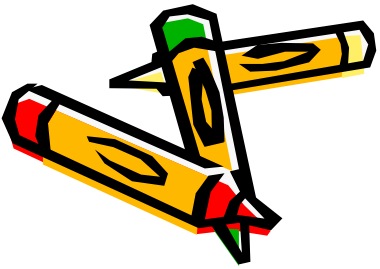
Delete

- Delete from <tabla> [where <condición>]



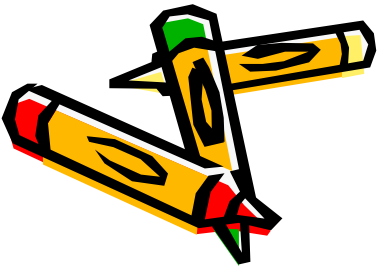
Consultas

- (Proyección) Select <col1,col2,…> from <tabla>
- (Selección) Select * from tabla where <condición>
- Operadores de comparación >,< >=,<=,<> != (Funcionan no solo con campos numéricos)
- Operadores aritméticos +,-,*,/,^ (Para cálculo de columnas o en expresiones para la cláusula WHERE
- Operadores lógicos AND OR NOT



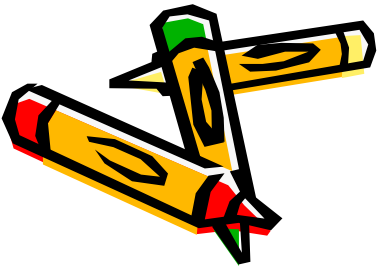
Consultas

- Cláusula order by <col1 [ASC| DESC],col2,…>
- Cláusula distinct
- Select distinct v_code from product;
- Alias para columnas



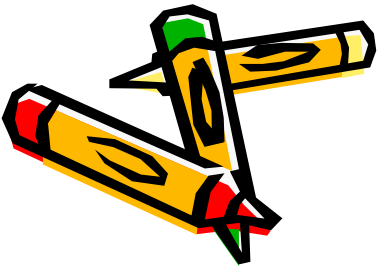
Operadores especiales

- BETWEEN
- IN
- LIKE
- IS NULL
- EXISTS



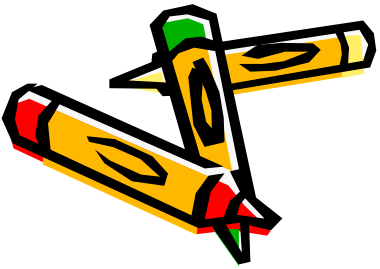
Ejemplo con IN

- Listar las claves y los nombres de los proveedores que nos han vendido algun producto
- Select v_code,v_name from vendor where v_code in (select v_code from product);



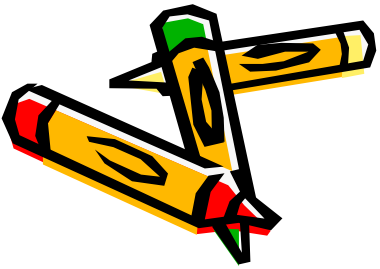
Ejemplo con exists

- Lista los nombres de todos los vendedores si ocurre que necesitamos algún producto
- Select v_name from vendor where exists (select * from product where p_onhand <= p_min);



Funciones agregadas de SQL

- Count
- Min
- Max
- Sum
- avg

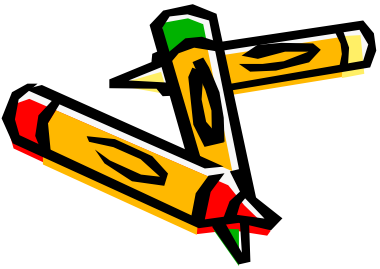


Agrupamiento de datos

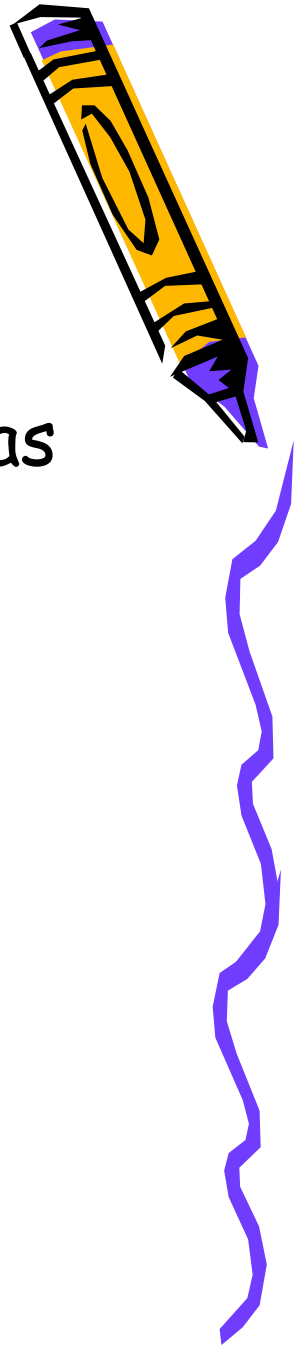
- SELECT <lista_columnas>
  FROM <lista_tablas>
  [WHERE <condición>]
  [GROUP BY <lista_columnas>]
  [HAVING <condición>]
  [ORDER BY <col1> ASC|DESC,
  <col2> ASC|DESC,…]

Select v_code,min(p_price) from product group by v_code;

- Select v_code,count(distinct (p_code)), avg(p_price) from product group by v_code;
- Select v_code,count(distinct (p_code)), avg(p_price) from product group by v_code having avg(p_price) < 10;



Ej Estima la inversión importante



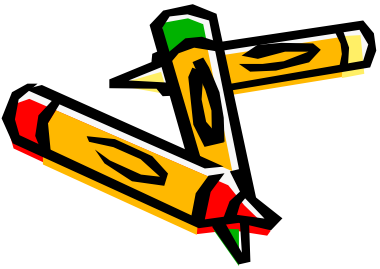
- Select v_code,sum(p_onhand*p_price) as costo_total

  from product

  group by v_code

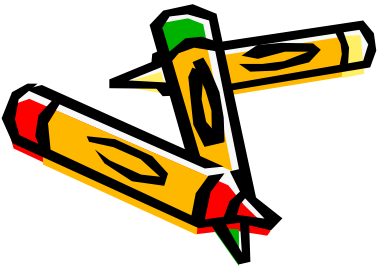
  having sum(p_onhand*p_price)>2000

  order by costo_total desc

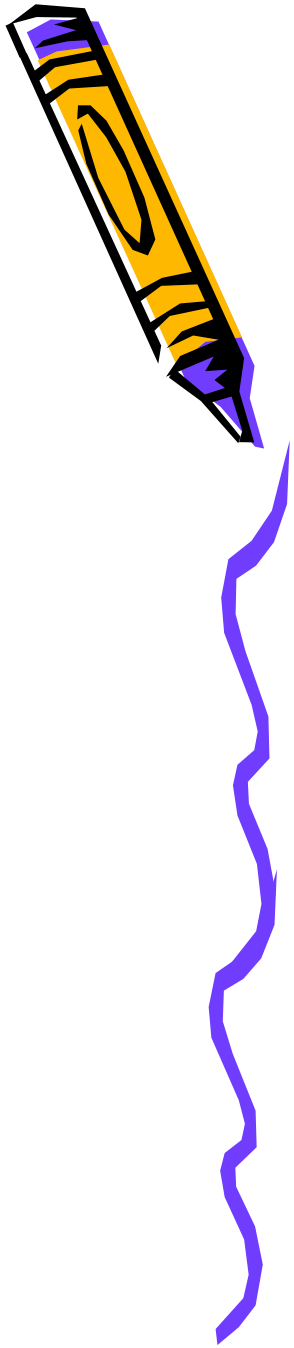


Equi-junta

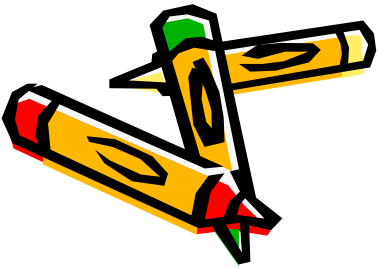
- Select p.p_descript,p.p_price,v.v_name, v.v_contact,v.v_areacode,v.v_phone
  from product p,vendor v
  where p.v_code=v.v_code
  and p_indate > '15-Ene-2004'
  order by p_price;



Listar apellido, número de pedido, fecha del pedido y descripciones de los productos del cliente 10014

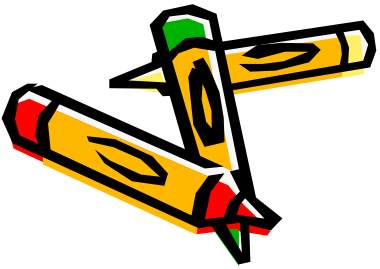


- Select cus_lname, i.inv_number, inv_date, p_descript
- from customer c, invoice i, line, product p
- where c.cus_code = i.cus_code and
- line.p_code=p.p_code and
- c.cus_code=10014 and
- i.inv_number=line.inv_number
- order by i.inv_number;



Ejemplos de subconsultas

- Select p_code,p_descript,p_price from product where p_price= (select max(p_price) from product);
- Select p_descript, p_onhand, p_price, v_code from product where p_price > (select avg(p_price) from product) order by p_price desc;



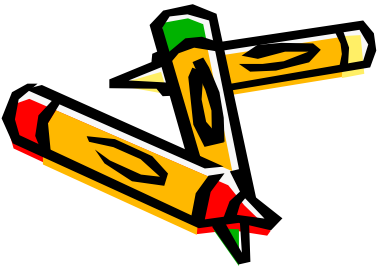
Junta de una tabla consigo misma

- Select e.emp_num,e.emp_lname, m.emp_num, m.emp_lname

  from emp e,emp m

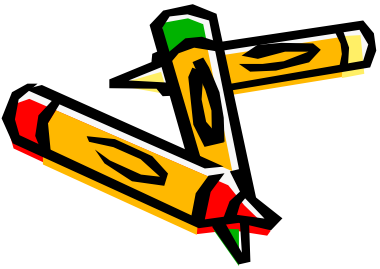
  where e.emp_mgr=m.emp_num

  order by e.emp_mgr;



Outer joins

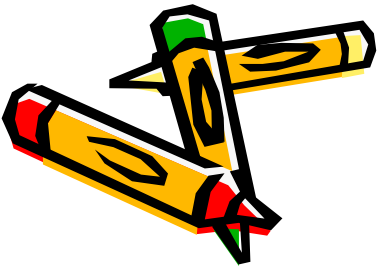
- La equi-junta de las tablas product y vendor produce solo 14 lineas, sin embargo la tabla product tiene 16 registros, el problema es que dos registros tienen NULL en el campo v_code, para incluir estos se necesita hacer un "outer join"



Left outer join mostrará todos los vendedores junto con los productos que vende encontrados en la tabla product

- Select p_code,vendor.v_code,v_name
  from vendor
  left join product
  on vendor.v_code=product.v_code;

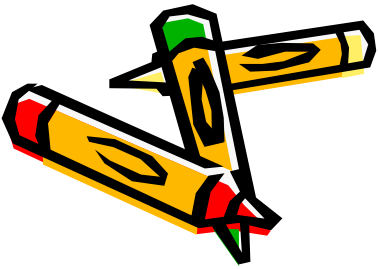
- Select p_code,vendor.v_code,v_name
  from vendor ,product
  where vendor.v_code=product.v_code (+);



Right outer join mostrará todos los Productos junto con los vendedores que los proveen encontrados en la tabla vendor

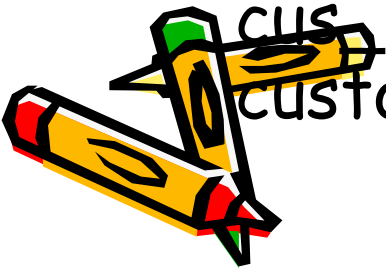
- Select p_code,vendor.v_code,v_name
  from vendor
  right join product
  on vendor.v_code=product.v_code;

- Select p_code,vendor.v_code,v_name
  from vendor ,product
  where vendor.v_code (+)=product.v_code;



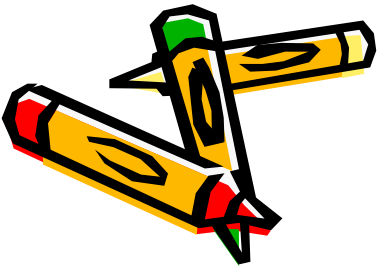
Union y union all

- Select <lista_columnas> from <tabla> union
  Select <lista_columnas> from <tabla> union
  Select <lista_columnas> from <tabla> union …
- Select cus_lname, cus_fname, cus_initial, cus_areacode, cus_phone from customer UNION Select cus_lname, cus_fname, cus_initial, cus_areacode, cus_phone from customer_2;
- Select cus_lname, cus_fname, cus_initial, cus_areacode, cus_phone from customer UNION ALL Select cus_lname, cus_fname, cus_initial, cus_areacode, cus_phone from customer_2;



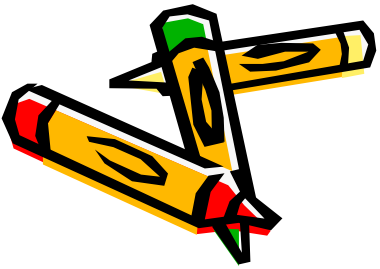
Intersect.

- Ej Detecta los duplicados
- Select cus_lname, cus_fname, cus_initial, cus_areacode, cus_phone from customer INTERSECT Select cus_lname, cus_fname, cus_initial, cus_areacode, cus_phone from customer_2;
- Ej Clientes con código de área 615 que han hecho alguna compra
- Select cus_code from customer where cus_areacode='615' intersect select distinct cus_code from invoice;



Minus

- No es conmutativo
- Select cus_lname, cus_fname, cus_initial, cus_areacode, cus_phone from customer MINUS Select cus_lname, cus_fname, cus_initial, cus_areacode, cus_phone from customer_2;
- Clientes que no han hecho compras
- Select cus_code from customer where cus_areacode='615' minus select distinct cus_code from invoice;

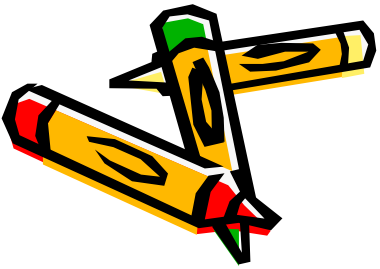


Juntas

Select * from T1,T2	Producto cartesiano
Select * from T1 cross join T2	Producto cartesiano
Select * from T1,T2 where T1.C1=T2.C1	Equi-junta
Select * from T1 natural join T2	Junta natural
Select * from T1 join T2 using (C1)	Junta natural restringida a ciertas columnas
Select * from T1 join T2 on T1.C1=T2.C1	Equi-junta
Select * from T1 left outer join T2 on T1.C1=T2.C1	Incluye todos los registros de la tabla izquierda (T1)
Select * from T1 right outer join T2 on T1.C1=T2.C1	Incluye todos los registros de la tabla derecha (T2)
Select * from T1 full outer join T2 on T1.C1=T2.C1	Incluye todos los registros de ambas tablas

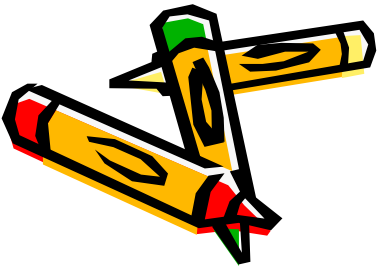
ANY y ALL

- El operador IN implica comparaciones con =, para comparaciones con < o con > podemos usar ANY o ALL
- Select p_code, p_onhand*p_price from product where p_onhand*p_price > ALL (Select p_onhand*p_price from product where v_code in (select v_code from vendor where v_state='FL'));



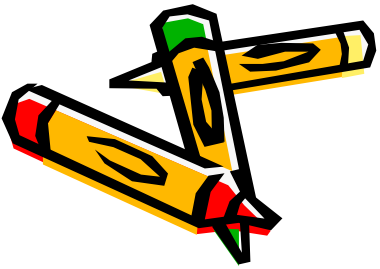
La salida de una subconsulta es una tabla virtual

- Listar los nombres de los clientes que han comprado ambos productos con código 13-Q2/P2 y 23109-HB
- Select distinct customer.cus_code, customer. Cus_lname from customer, (select invoice.cus_code from invoice natural join line where p_code='13-Q2/P2') CP1, (Select invoice.cus_code from invoice natural join line where p_code='23109-HB') CP2 where customer.cus_code=CP1.cus_code and cp1.cus_code=cp2.cus_code;



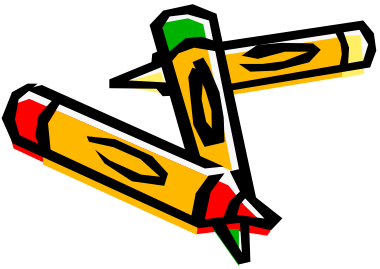
Subconsultas "inline"

- La subconsulta debe regresar un solo valor para poderse usar en la lista de columnas
- Ej. Listar para cada producto su precio y la diferencia con el promedio de precios
- Select p_code, p_price, (select avg(p_price) from product) as avgprice, p_price - (select avg(p_price) from product) as diff from product



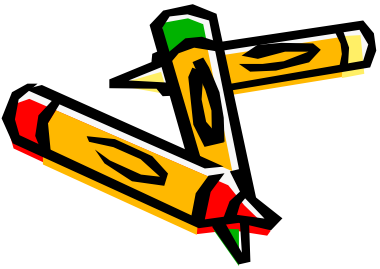
Consultas correlacionadas

- Cuando se usan subconsultas primero se ejecutan las consultas mas internas hasta llegar a la principal
- En cambio, en las consultas correlacionadas, se inicia primero la consulta externa y por cada registro que regresa esta, se ejecuta la consulta interna recibiendo como parámetro el registro de la externa

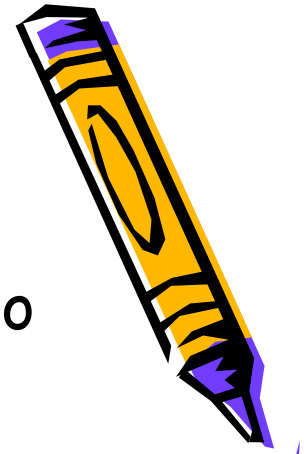


Una consulta correlacionada

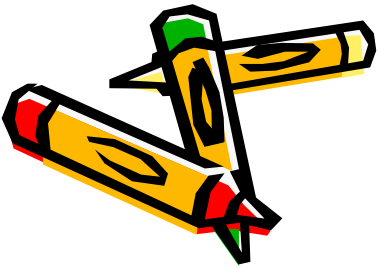
- Lista todos las ventas en las que las unidades vendidas fueron mas que el promedio de unidades vendidas para ese producto.
- Select inv_number, p_code, line_units from line ls where ls.line_units > (select avg(line_units) from line la where la.p_code=ls.p_code);
- Select inv_number, p_code, line_units, (select avg(line_units) from line lx where lx.p_code=ls.p_code) as avg from line ls where ls.line_units > (select avg(line_units) from line la where la.p_code=ls.p_code);



Consulta correlacionada y EXISTS

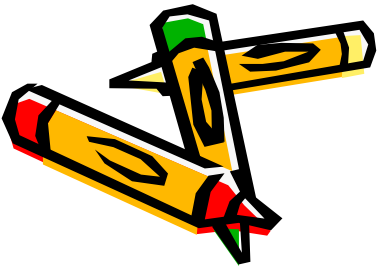


- Ej. Lista los clientes que nos han comprado algo
- Select cus_code, cus_lname, cus_fname from customer where exists (select * from invoice where invoice.cus_code = customer.cus_code);
- Ej Lista los proveedores que nos han vendido algún artículo de los que tenemos menos del doble del mínimo
- Select v_code, v_name from vendor where exists (select * from product where p_onhand < p_min*2 and vendor.v_code = product.v_code);



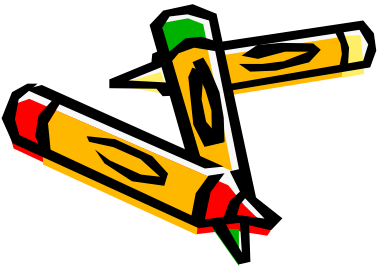
PL/SQL

- Incluye aspectos procedurales como declaración de variables, funciones, sentencias de control de flujo (condicionales y ciclos)
- Permite la ejecución de procedimiento almacenados
- Permite la implementación de disparadores
- Se puede combinar con otros lenguajes procedurales como C, Java, etc



Estructura de un programa en PL/SQL

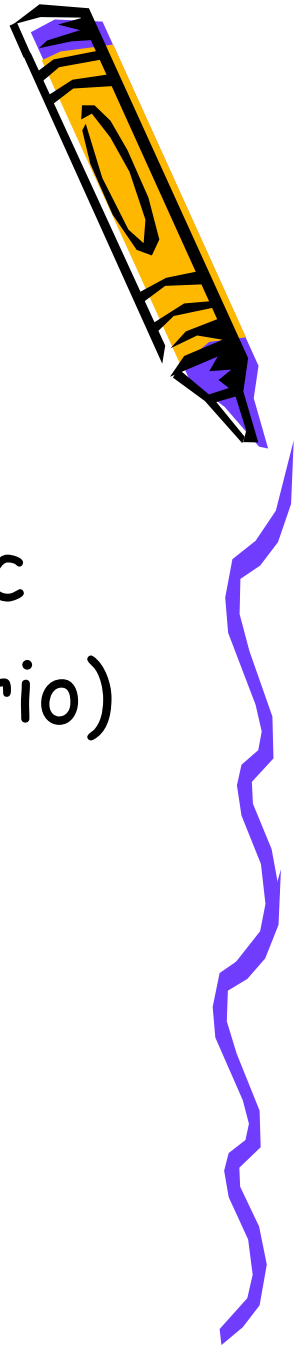
```
DECLARE
/* declaracion de variables, cursores y
   excepciones de usuario */
BEGIN
   /* Sentencias ejecutables */
EXCEPTION
   -- Control de errores
END;
```



Ejemplo

```
set serveroutput on
declare
   w_p1 number(3):=0;
   w_p2 number(3):=10;
   w_num number(2):=0;
begin
    while w_p2 < 300 loop
        select count(p_code) into w_num from product where
            p_price between w_p1 and w_p2;
        dbms_output.put_line('Hay '||w_num||' productos con
               precio entre '||w_p1||' y '||w_p2);
        w_p1:=w_p2+1;
       w_p2:=w_p2+50;
   end loop;
end;
```

Tipos de variables

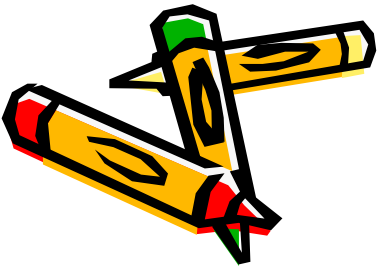


- Escalares

Varchar2, date,lob, long, number, etc

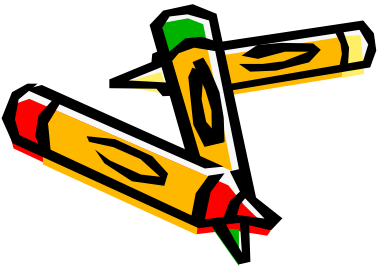
- Compuestos (definidos por el usuario)

Registros, tablas y matrices



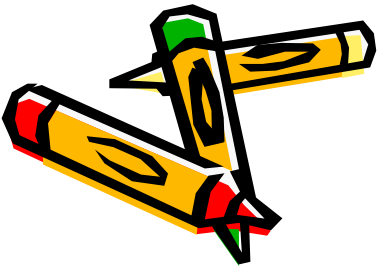
Declaración

- <nom_variable> <tipo> [CONSTANT] [NOT NULL] [:=<valor>];
- <tipo> puede ser un tipo basico de SQL, tabla.campo%TYPE ó tabla %ROWTYPE



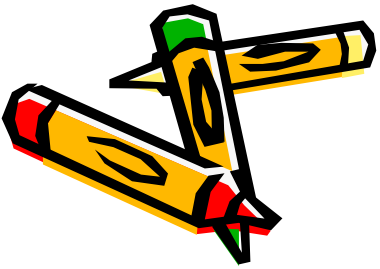
Ejemplos de declaraciones

- precio NUMBER:=300;
- factor CONSTANT NUMBER(3,2):=0.1;
- rcliente clientes%ROWTYPE;
- mat alumnos.matricula%TYPE;



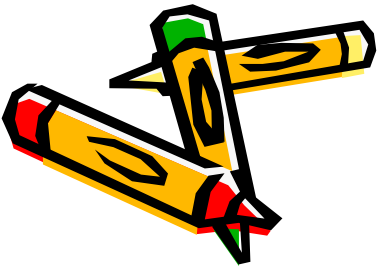
Condicionales

- IF <condición>
  THEN <sentencia>;
  [ELSEIF <condición> THEN <sentencia>]
  [ELSEIF <condición> THEN <sentencia>]
  :
  [ELSE <sentencia>]
  END IF;

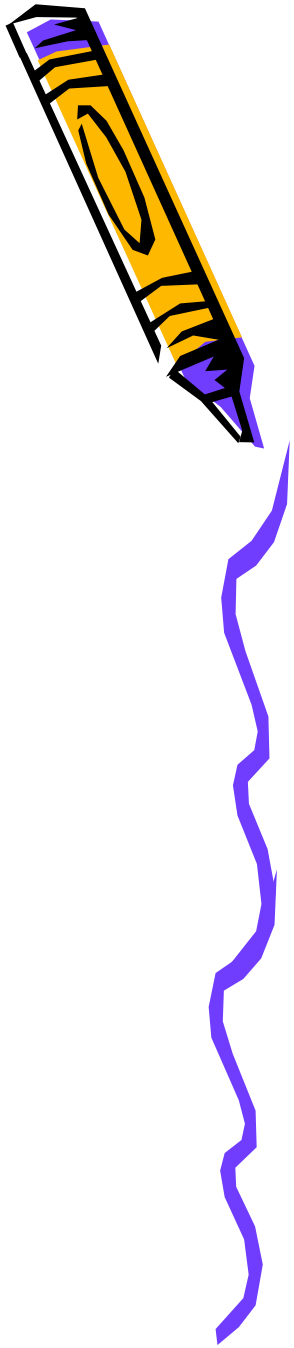


Saltos

- Goto <etiqueta>
- EXIT (Salir de un ciclo)



Ciclos infinitos



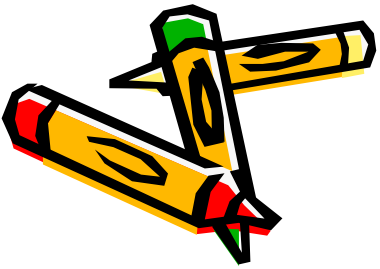
- [etiq] LOOP

  <sentencia>;

  <sentencia>;

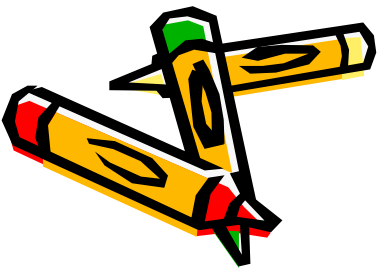
  :

  END LOOP;



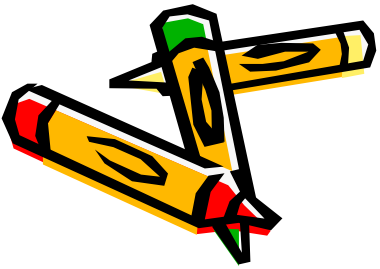
Ejemplo ciclo infinito

```
declare
contador number:=1;
begin
  loop
     insert into numeros values (contador);
     contador:=contador + 1;
     exit when contador > 50;
  end loop;
end;
```



[etiq] WHILE <condición> LOOP
<sentencia
END LOOP;

```
declare
contador number:=1;
begin
  while contador <= 50 loop
     insert into numeros values (contador);
     contador:=contador + 1;
  end loop;
end;
```

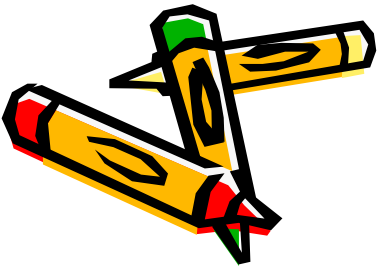


for <indice> **in** **[reverse]** <ini>..<fin> **loop**

<sentencia>

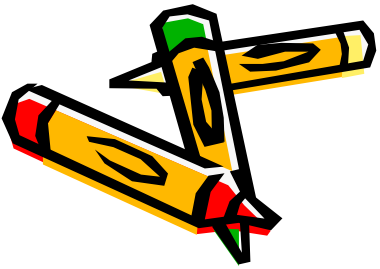
end loop;

```
declare
contador number:=1;
begin
  for contador in 1.. 50 loop
     insert into numeros values (contador);
  end loop;
end;
```



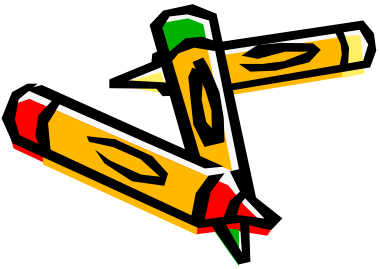
Cursores

- En la parte de declaraciones se define el cursor
- OPEN: ejecuta la consulta, se recuperan las filas y se sitúa el "cursor" en la primera de las filas
- FETCH: copia los valores de la fila donde se ubica el cursor en variables locales
- CLOSE: se cierra el cursor



Mostrar el num de seguro social de los empleados cuyo salario es mayor que el de su supervisor

```
Declare
   salario number;
   sal_super number;
   nss varchar2(9);
   nss_super varchar2(9);
   cursor cursor_salario is
      select nss, salario, nss_supervisor from
      empleado;
```

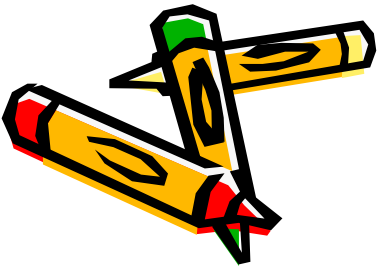


Continuación …

```
Begin
   open cursor_salario;
   loop
        fetch cursor_salario into nss,  salario, nss_super;
        exit when cursor_salario%notfound;
        if nss_super is not null then
             select salario into sal_super from empleado
                  where  nss=nss_super;
            if salario > sal_super then
                 dbms_output.put_line(nss);
           end if;
       end if;
   end loop;
   if cursor_salario%isopen then close cursor_salario;
```

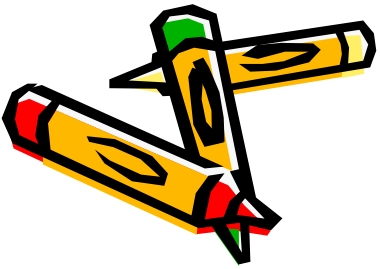
Continuacion(2)…

```
exception
    when no_data_found then
        dbms_output.put_line('Datos no
        encontrados '||nss);
   if cursor_salario%isopen then
      close cursor_salario;
end;
```

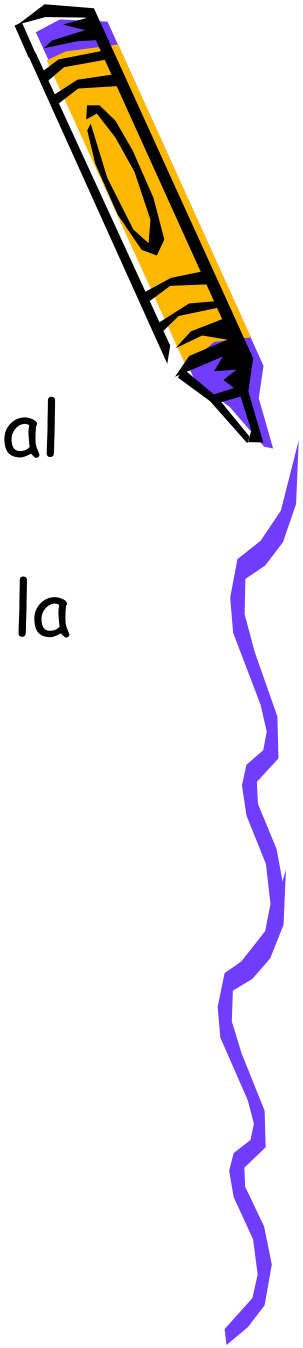


```
for <var_reg> in <consulta> loop
      <sentencia>
end loop;
```

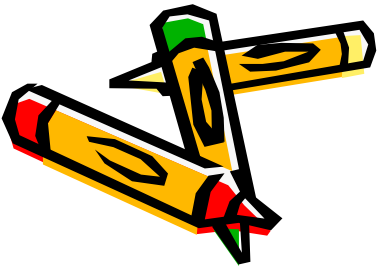
```
begin
    for registro in (select nombre, domicilio from
                          cliente)
         loop
             insert into temporal values
                 (registro.nombre, registro.domicilio);
         end loop;
end;
```



disparadores



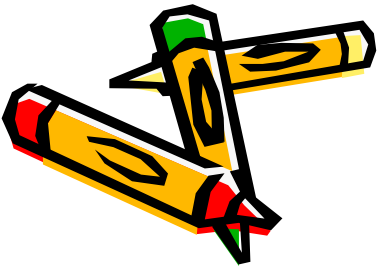
- Definen acciones que se deben realizar al ocurrir eventos
- Se usan para mantener la integridad de la base de datos
- Tambien se usan para auditar modificaciones de la base de datos
- Se almacenan en la base de datos
- Se compilan al crearlos o modificarlos



Sintaxis para crear disparador

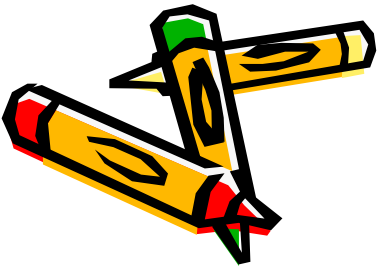
- Create or replace trigger <nombre_disparador> {before|after}

  {delete|insert|update [of <col1>,<col2>,…]} on <tabla> [for each row] <bloque PL/SQL>



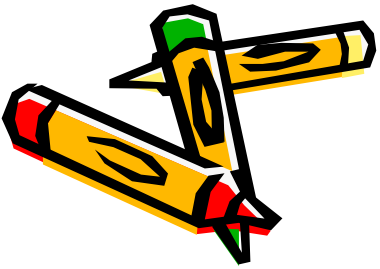
Ejemplo: disparador que avisa cuando se debe comprar algun producto

```
create or replace trigger trg_prod
   after insert or update of p_onhand
   on product
   begin
      update product set p_reorder=1
      where p_onhand <= p_min;
   end;
```



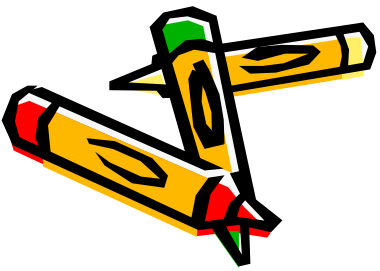
Si en lugar de reducir p_onhand se aumenta p_min?

```
create or replace trigger trg_prod
   after insert or update of p_onhand,
   p_min on product
   begin
      update product set p_reorder=1
      where p_onhand <= p_min;
   end;
```



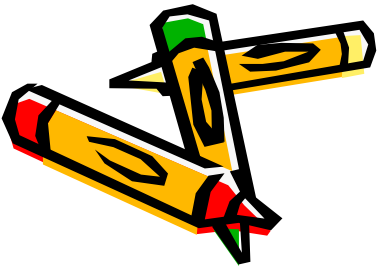
Si se incrementa p_onhand?

```
create or replace trigger trg_prod before insert or
   update of p_onhand on product for each row
  begin
    if :new.p_onhand <= :new.p_min then
        :new.p_reorder:=1;
    else
        :new.p_reorder:=0;
    end if;
end;
```



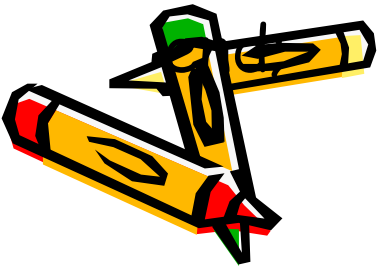
Actualizar la existencia tras una venta

```
create or replace trigger trg_line_prod
   after insert on line for each row
begin
     update product set
        p_onhand = p_onhand - :new.line_units
       where product.p_code = :new.p_code;
end;
```



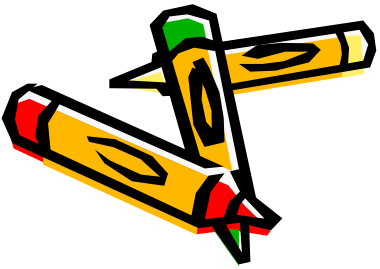
Actualiza el saldo del cliente

```
create or replace trigger trg_line_cus after insert
   on line for each row
  declare
   w_cus char(5);
   w_tot number:=0;
begin
   select cus_code into w_cus from invoice where
      invoice.inv_number=:new.inv_number;
   w_tot:=:new.line_price*:new.line_units;
   update customer set cus_balance = cus_balance +
    w_tot where cus_code=w_cus;
    dbms_output.put_line('Saldo modificado'||w_cus);
```



Procedimientos almacenados

```
create or replace procedure
   prc_prod_discount as
begin
    update product set
    p_discount=p_discount+0.05 where
   p_onhand >= p_min*2;
   dbms_output.put_line('Listo');
end;
```



Para ejecutarlo:
SQL> EXEC[UTE] prc_prod_discount;

Segunda versión (parametro de entrada)

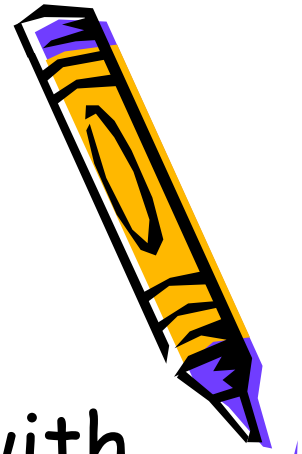
```
create or replace procedure prc_prod_discount(wpi in number)
    as
Begin
     if (wpi <=0) or (wpi >=1) then
          dbms_output.put_line('Error: No puede haber descuentos
    negativos ni de mas de 100%');
   else
       update product set
       p_discount=p_discount+wpi where p_onhand >= p_min*2;
      dbms_output.put_line('Listo');
   end if;
end;
```

Para ejecutarlo:
SQL> EXEC[UTE] prc_prod_discount(<num>);

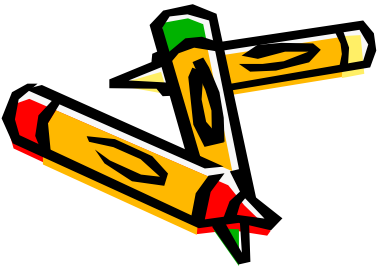
Función almacenada

```
create or replace function
                    calc_edad(fec_nac in number)
                                        return number is
begin
return trunc(
            abs(
                months_between(
                        to_date(fec_nac,'YYYYMMDD')
                            ,sysdate)/12
                )
            );
```

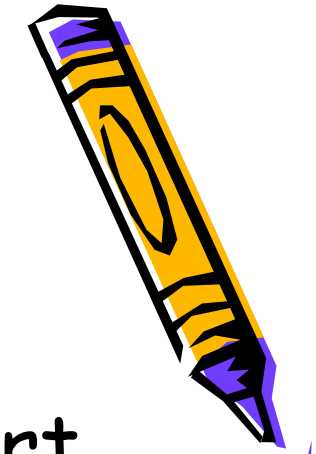
Secuencias



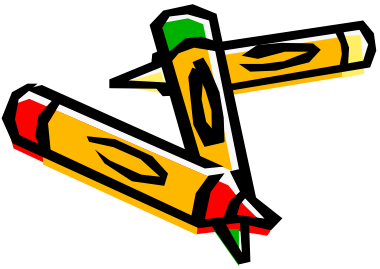
- Create sequence <nombre> [start with <num>] [increment by <num>] [maxvalue <num>] [minvalue <num>] [cycle|nocycle]
- Alter sequence <nombre> …
- Drop sequence <nombre>;



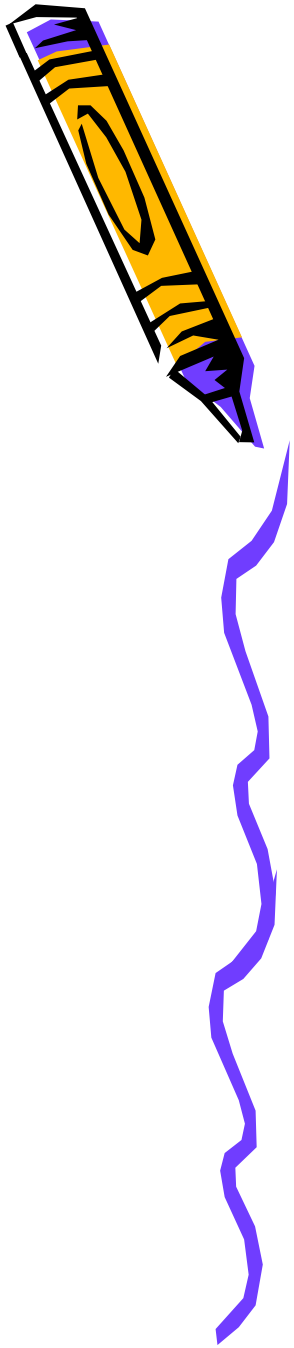
Ejemplo



- Create sequence mi_secuencia start with 100 minvalue 100 increment by 10 maxvalue 200 cycle;
- Select mi_Secuencia.nextval from val;
- Select mi_secuencia.currval from val;

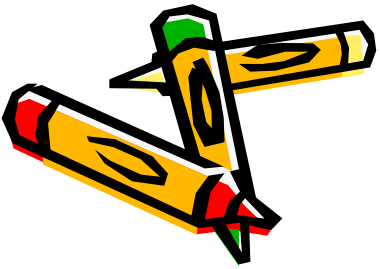


JDBC



- La variable de ambiente ORACLE_HOME por ejemplo: c:\oracleexe\app\oracle\product\10.2.0\server\
- Agregar a la variable de ambiente CLASSPATH la ruta %ORACLE_HOME%\jdbc\lib\ojdbc14.jar
- Verificar el archivo %ORACLE_HOME %\NETWORK\ADMIN\tnsnames.ora

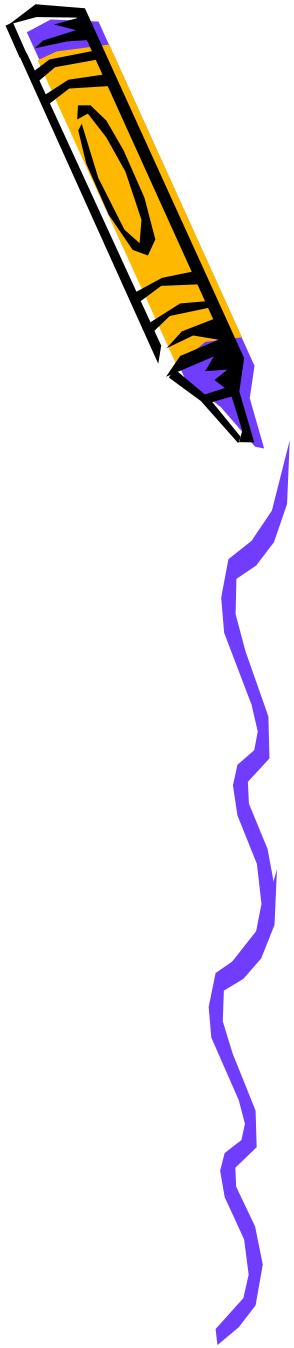
```
XE =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = D37LF9F1)(PORT =
     1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = XE)
    )
  )
```



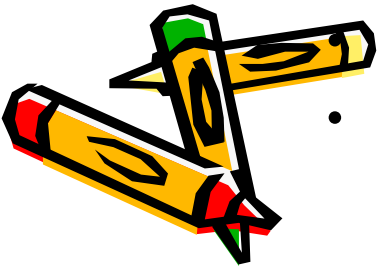
Ejemplo Consulta

```
import java.sql.*;
 class prueba_jdbc {
  public static void main (String args []) throws SQLException  {
    DriverManager.registerDriver (new
     oracle.jdbc.driver.OracleDriver());


    Connection conexion = DriverManager.getConnection
          ("jdbc:oracle:thin:@D37LF9F1:1521:XE", "antonio", "susa1412");
         // driver@machineName:port:SID          ,  userid,  password
    Statement sentencia = conexion.createStatement();
    ResultSet resultado =
             sentencia.executeQuery("select
     entidad,municipio,nombre_municipio from catalogo_municipios");
    while (resultado.next())
         System.out.println (resultado.getString(1)+"
     "+resultado.getString(2)+" "+resultado.getString(3));
    sentencia.close();
```

Métodos de la clase ResulSet

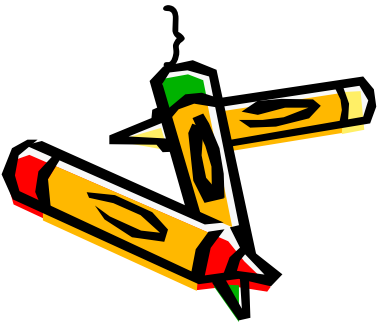


- boolean next()
- boolean previous()
- boolean first()
- boolean last()
- boolean isLast()
- boolean isFirst()
- String getString(int columnIndex)
- String getString(String columnName)
- int getInt(int columnIndex)
- int getInt(String columnName)
- float getFloat(int columnIndex)
- java.sql.Date getDate(int columnIndex)
- boolean wasNull()
- ResulSetMetaData getMetaData()



Ejem wasNull()

```
Statement sql=db.createStatement();
String cadena_sql="Select num_mgr from
   empleados";
ResultSet rs=sql.executeQuery(cadena_sql);
While (rs.next()) {
   int num=rs.getInt("num_mgr");
   if (rs.wasNull())
        System.out.println("No tiene jefe");
   else
        System.out.println("El jefe tiene num_emp
   "+num);
```



Ejemplo manipulación

```
import java.sql.*;
 class inserta_jdbc {
  public static void main (String args []) throws SQLException
    {
    DriverManager.registerDriver (new
    oracle.jdbc.driver.OracleDriver());

    Connection conexion = DriverManager.getConnection
         ("jdbc:oracle:thin:@D37LF9F1:1521:XE", "antonio",
    "susa1412");
        // driver@machineName:port:SID          ,  userid,
    password
    Statement sentencia = conexion.createStatement();
    sentencia.execute("insert into catalogo_municipios values
    (33,1,'Disneylandia')");
    sentencia.close();
    }
  }
```

```
import java.sql.*;
 class inserta_muchos_jdbc {
  connection conexion;


  public static void main (String args []) throws SQLException  {
    DriverManager.registerDriver (new oracle.jdbc.driver.OracleDriver());
    inserta_muchos_jdbc a=new inserta_muchos_jdbc();
    a.conecta();
    a.inserta();
  }


  public void conecta() throws SQLException {
  conexion = DriverManager.getConnection
         ("jdbc:oracle:thin:@D37LF9F1:1521:XE", "antonio", "susa1412");
   }


  public void inserta() throws SQLException {
       String municipios[]={"Disneylandia","Springfield","Shelbyville"};
       for (int i=0;i<municipios.length;i++) {
    Statement sentencia = conexion.createStatement();
    System.out.println("insert into catalogo_municipios values
      33,"+i+",'"+municipios[i]+"')");
    sentencia.execute("insert into catalogo_municipios values
      (33,"+i+",'"+municipios[i]+"')");
    sentencia.close();
       }
  }
}
```

SQL dinámico

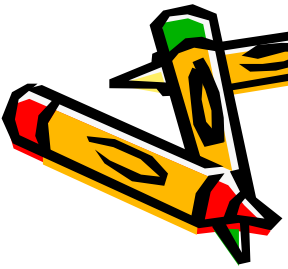
```
import java.sql.*;
 class inserta_muchos_ver2 {
  Connection conexion;

  public static void main (String args []) throws SQLException  {
    DriverManager.registerDriver (new oracle.jdbc.driver.OracleDriver());
    inserta_muchos_ver2 a=new inserta_muchos_ver2();
    a.conecta();
    a.inserta();
  }

  public void conecta() throws SQLException {
  conexion = DriverManager.getConnection
         ("jdbc:oracle:thin:@D37LF9F1:1521:XE", "antonio", "susa1412");
        // driver@machineName:port:SID          ,  userid,  password
  }

  public void inserta() throws SQLException {
       String municipios[]={"Disneylandia","Springfield","Chelbyvile"};
       String cadena_sql="insert into catalogo_municipios values (33,?,?)";
       PreparedStatement ps=conexion.prepareStatement(cadena_sql);
       for (int i=0;i<municipios.length;i++) {
              ps.setInt(1,i);
              ps.setString(2,municipios[i]);
               ps.execute();
        }
   }
}
```

Escribe Blob

```
import java.sql.*;
import java.io.*;
 class escribe_blob {
  Connection conexion;

  public static void main (String args []) throws SQLException,IOException  {
    DriverManager.registerDriver (new oracle.jdbc.driver.OracleDriver());
    escribe_blob a=new escribe_blob();
    a.conecta();
    a.escribe();
  }

  public void conecta() throws SQLException {
     conexion = DriverManager.getConnection
         ("jdbc:oracle:thin:@D37LF9F1:1521:XE", "antonio", "susa1412");
        // driver@machineName:port:SID          ,  userid,  password
     conexion.setAutoCommit(false);
  }

  public void escribe() throws SQLException,IOException {
       PreparedStatement stmt;
       File f=new File("imagen.jpg");
       FileInputStream fis = new FileInputStream(f);
       stmt = conexion.prepareStatement("INSERT INTO empleados(imagen)
      VALUES(?)");
       stmt.setBinaryStream(1, fis, (int) f.length());
       stmt.execute();
       conexion.commit();
       fis.close();
       conexion.close();
  }
```

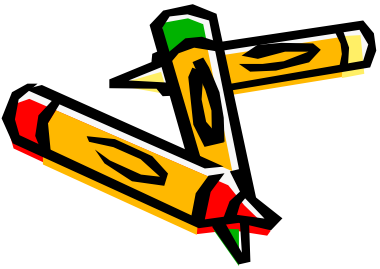
Leer un BLOB

```
import java.sql.*;
import java.io.*;
 class lee_blob {
  Connection conexion;
  public static void main (String args []) throws SQLException,IOException  {
    DriverManager.registerDriver (new oracle.jdbc.driver.OracleDriver());
    lee_blob a=new lee_blob();
    a.conecta();
    a.lee();
  }

  public void conecta() throws SQLException {
  conexion = DriverManager.getConnection
         ("jdbc:oracle:thin:@D37LF9F1:1521:XE", "antonio", "susa1412");
        // driver@machineName:port:SID          ,  userid,  password
  }

  public void lee() throws SQLException,IOException {
    Statement sentencia=conexion.createStatement();
    ResultSet rs=sentencia.executeQuery("Select imagen from empleados");
    if (rs.next()) {
         File file = new File("mi_archivo.jpg");
         FileOutputStream fos = new FileOutputStream(file);
         Blob bin = rs.getBlob("imagen");
         InputStream inStream = bin.getBinaryStream();
         int size = (int)bin.length();
         byte[] buffer = new byte[size];
         int length = -1;
         while ((length = inStream.read(buffer)) != -1) fos.write(buffer, 0, length);
        fos.close();
      }
      sentencia.close();
    }
  }
```



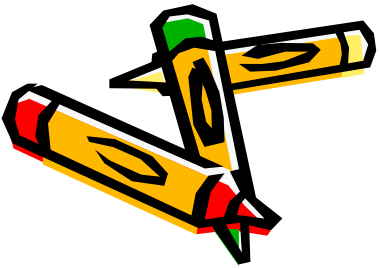
Consultas interactivas con SQL*Plus

- Pedir un valor al usuario con &
- Ampersand doble &&
- Comando Define
- Comando Accept



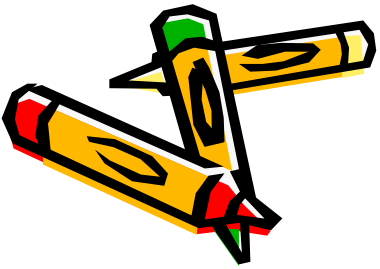
Pedir un valor al usuario con &

- Usar & como prefijo de una variable
- select entidad, municipio, nombre_municipio from catalogo_municipios where entidad=&edo and municipio=&mpio;
- Probar con SET VERIFY ON

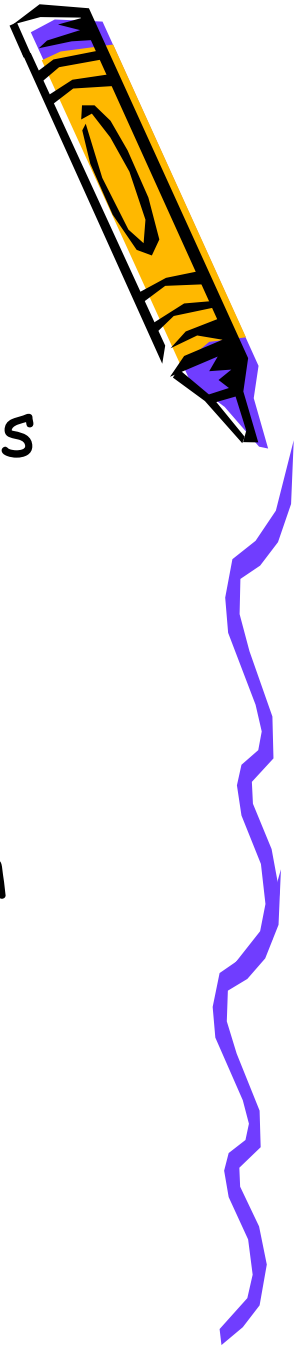


Para campos tipo cadena

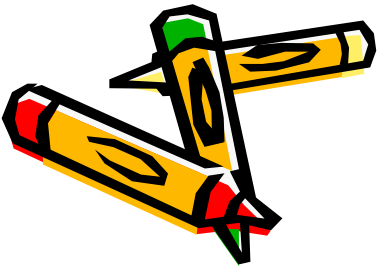
- select entidad, municipio, nombre_municipio from catalogo_municipios where nombre_municipio='&nombre_mpio'
- select entidad, municipio, nombre_municipio from catalogo_municipios where nombre_municipio = upper('&nombre_mpio')



Doble ampersand (&&)



- Usar doble && sirve para que no pida dos veces la misma variable de substitución
- Nota: Las variables de substitución se pueden usar en cualquier parte de la consulta
- select entidad,&&nombre_columna from catalogo_municipios where entidad=16 order by &nombre_columna

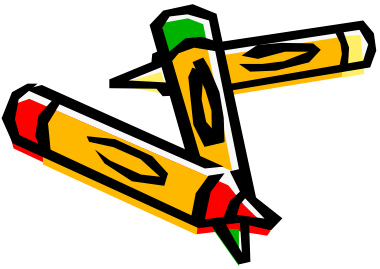


El comando Accept

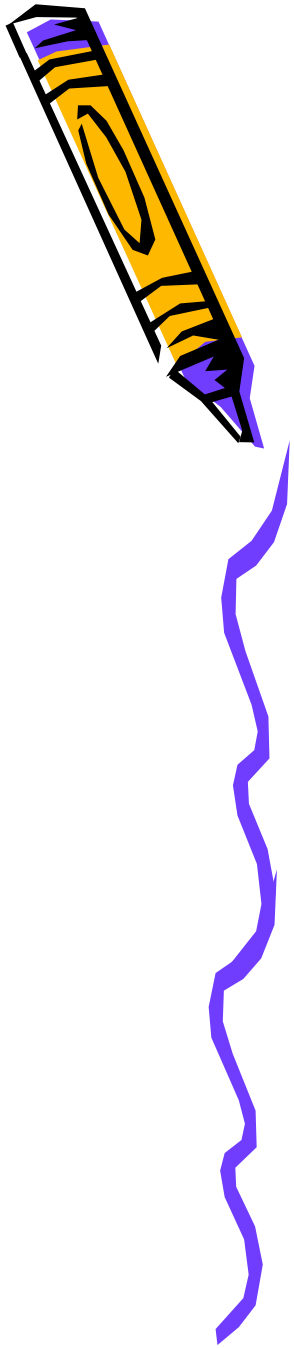
- Para pedir un dato al usuario de manera mas amigable

```
Accept &nombre_mpio prompt 'Dame el nombre del municipio
    que buscas:'
select entidad, municipio, nombre_municipio from
    catalogo_municipios where nombre_municipio =
   upper('&nombre_mpio');
```

- Accept <variable> prompt <texto> [hide]
- Usar 'hide' impide que se vea lo que el usuario escribe (útil para passwords)



Produciendo salida leíble con SQL*Plus



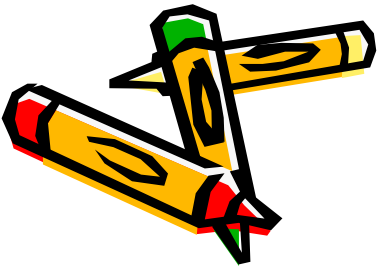
- Variables de ambiente de SQL*Plus
- SET <variable_ambiente> <valor>
- SHOW <variable_ambiente>
- Archivo login.sql
- Comandos de formato

-column

-ttitle

-btitle

-break

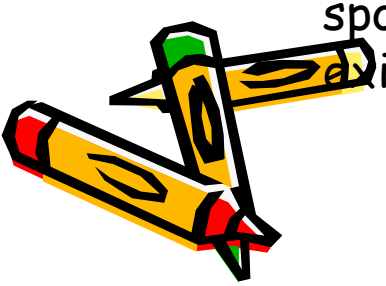


Ejemplo generación de reporte

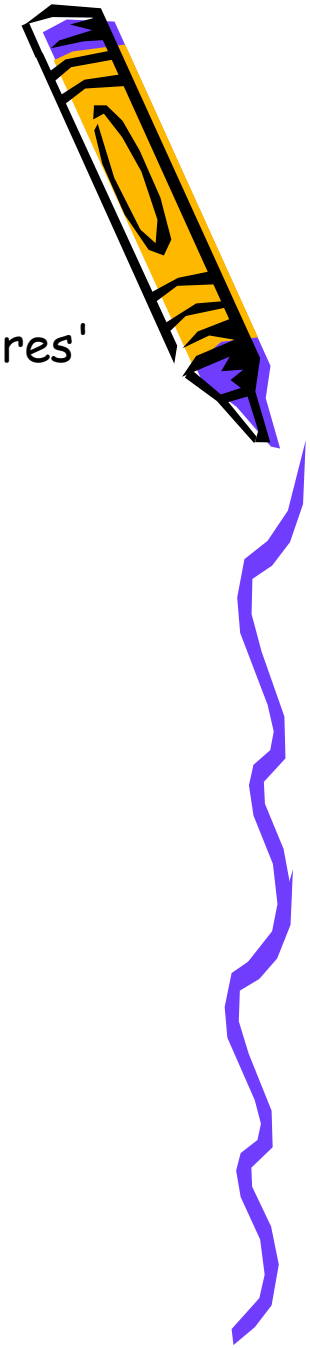
```
set pagesize 66
set linesize 80
set feed off
ttitle 'Instituto Federal Electoral|Listado Nominal de Electores'
btitle 'Centro Regional de Computo Morelia'
column clave_electoral heading 'CLAVE ELECTORAL' format a18
column nombre_completo heading 'APELLIDO PATERNO, MATERNO y NOMBRE'
    format a40
column dto format 99
column mpio format 999
column secc format 9999
spool listado
Select alfa_clave_electoral||lpad(fecha_nacimiento,6,0)||
    lpad(lugar_nacimiento,20)||digit_verificador||lpad(clave_homonimia,2,0)
    clave_electoral,
apellido_paterno||' '||apellido_materno||' '||nombre
nombre_completo,distrito dto,municipio mpio,
seccion secc,manzana mza
from padron order by 1;
spool off
exit
```

Usando break

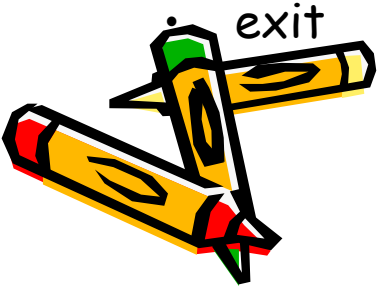
```
set pagesize 66
set linesize 80
set feed off
ttitle 'Instituto Federal Electoral|Listado Nominal de Electores'
btitle 'Centro Regional de Computo Morelia'
column clave_electoral heading 'CLAVE ELECTORAL' format a18
column nombre_completo heading 'APELLIDO PATERNO, MATERNO
    y NOMBRE' format a40
column manzana heading 'MANZANA'
break on manzana skip 2
compute count of clave_electoral on manzana
spool listado
select manzana, alfa_clave_electoral||lpad(fecha_nacimiento,6,0)||
    lpad(lugar_nacimiento,2,0)||digit_verificador||
    lpad(clave_homonimia,2,0) clave_electoral,
apellido_paterno||' '||apellido_materno||' '||nombre
    nombre_completo
from padron order by 1,2;
spool off
exit
```



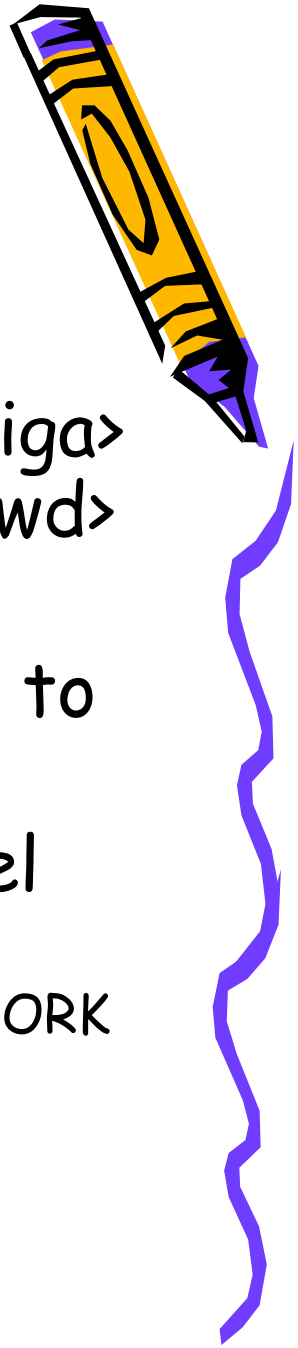
Un estadístico usando compute



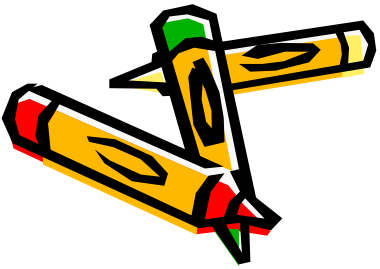
- set pagesize 66
- set linesize 80
- set feed off
- ttitle 'Instituto Federal Electoral|Listado Nominal de Electores'
- btitle 'Centro Regional de Computo Morelia'
- column manzana format a8
- column seccion format 9999
- column total format 9,999
- column ocupacion format 99
- break on seccion skip page on manzana skip 2
- compute sum of total on manzana
- compute sum of total on seccion
- spool estadistico
- select seccion,manzana,ocupacion,count(*) total
- from padron group by seccion,manzana,ocupacion
- order by 1,2,3;
- spool off
- exit



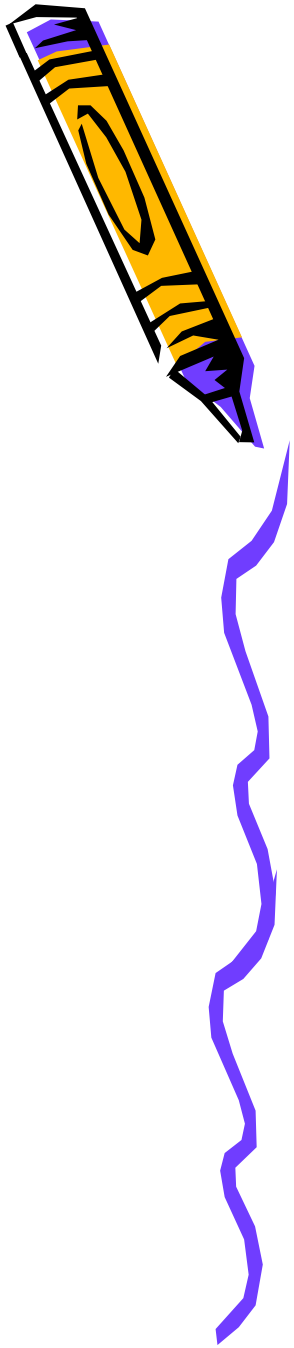
Acceso de bases de datos remotas



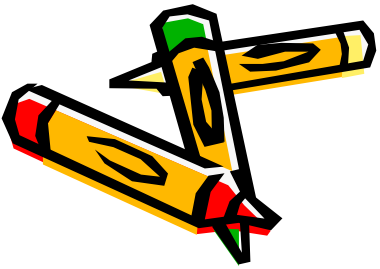
- Create [public] database link <nombre_liga> connect to <usuario> identified by <passwd> using '<cadena_de_coneccion>'
- Ej Create database link mi_liga connect to usu03 identified by xr24 using 'XE'
- En este ejemplo, XE es una entrada en el archivo $HOME/tnsnames.ora (en mi caso C:\oraclexe\app\oracle\product\10.2.0\server\NETWORK\ADMIN\tnsnames.ora)



Tnsnames.ora



```
XE =
 (DESCRIPTION =
  (ADDRESS = (PROTOCOL = TCP)(HOST = D37LF9F1)(PORT = 1521))
  (CONNECT_DATA =
   (SERVER = DEDICATED)
   (SERVICE_NAME = XE)
  )
 )

EXTPROC_CONNECTION_DATA =
 (DESCRIPTION =
  (ADDRESS_LIST =
   (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC_FOR_XE))
  )
  (CONNECT_DATA =
   (SID = PLSExtProc)
   (PRESENTATION = RO)
  )
 )

ORACLR_CONNECTION_DATA =
 (DESCRIPTION =
  (ADDRESS_LIST =
   (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC_FOR_XE))
  )
  (CONNECT_DATA =
   (SID = CLRExtProc)
   (PRESENTATION = RO)
  )
 )
```



Configuración

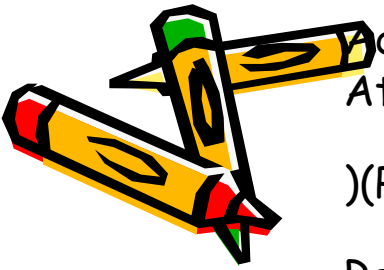
- Modificar el archivo tnsnames.ora de ambas máquinas para agregar la información de la base de datos remota de la otra maquina
- Tener comunicación a nivel sistema operativo, probar con ping
- Probar con tnsping, ej:

C:\Documents and Settings\Jose Antonio>tnsping xe

TNS Ping Utility for 32-bit Windows: Version 10.2.0.1.0 - Production on 07-MAY-2
010 10:15:24

Copyright (c) 1997, 2005, Oracle. All rights reserved.

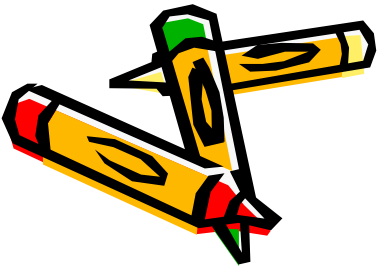
Archivos de parßmetros utilizados:
C:\oraclexe\app\oracle\product\10.2.0\server\network\admin\sqlnet.ora



Adaptador TNSNAMES utilizado para resolver el alias
Attempting to contact (DESCRIPTION = (ADDRESS = (PROTOCOL = TCP)(HOST = D37LF9F1
)(PORT = 1521)) (CONNECT_DATA = (SERVER = DEDICATED) (SERVICE_NAME = XE)))
Realizado correctamente (40 msec)

Para probar la liga

- Select * from prueba2@mi_liga;
- Con esto se accede a la tabla prueba2 del usuario usu03 en la máquina especificada por la entrada de tnsnames.ora denominada 'XE' (en este caso es la misma máquina
- Para correrlo realmente en una máquina remota el oracle Net debe estar corriendo en ambas máquinas



Borrar una liga

- Drop database link mi_liga