TABLAS:

Maestro clave numérica:

La típica tabla de SQL de toda la vida,

Se identifica por un ID único como clave y suele ser AI

ID Nom Ape Edad

1 Vic Mrtz 19

2 Victor Gnzl 20

Maestro clave Arbolada:

Clave Alfabética Mayus

Desde la clave Arbolada podemos crear Subregistros que pertenecen a esas claves:

TABLA SISTEMAS OPERATIVOS

ID NAME

**W** Windows

ID NAME

**W**indows Vista Windows Vista

**W**indows 98 Windows 98

**W**indows 7 Windows 7

**L** Linux

ID NAME

**L**inux GNU Linux GNU

**L**inux Debian Linux Debian

**L**inux Ubuntu Linux Ubuntu

**I** IOS

ID NAME

**I**OS Mobile IOS Mobile

Tablas historicas :

No tienen Clave/ID

Se usan para unir tablas por relaciones:

TABLA DTO

CLT FAM POR\_DTO

ID\_CLT ID\_FAM 10%

José Windows 20%

Luisa Linux 40%

Esto quiere decir que tanto Luisa tiene el 40% de descuento en la FAMILIA linux, en este caso la tabla de DTO, nos ha servido para unir las dos tablas y de esta forma poder manipular tanto los REGISTROS de FAMILIAS, como los De CLIENTES.

Tablas submaestras:

Es un subregistro, es decir, no puede existir sin depender de otras tablas.

Depende de otras tablas para existir:

TABLA LINEAS:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| PEDIDOS | ID | ARTICULOS | CNT | PRE | IMP |
| ID\_PEDIDO | ID\_LINEA | ID\_ART | CNT\_ART | PRE\_ART | Calculo() |
| 10 | 1 | 45 | 2 | 2€ | 30€ |
| 31 | 2 | 2134 | 34 | 5€ | 200€ |
| 76 | 3 | 42 | 12 | 1000€ | 2000€ |

Entendamos la tabla:

Esta tabla sub maestra depende de PEDIDOS y ARTICULOS para existir, porque si no hay PEDIDO, no podemos rellenar los datos y si no hay ARTICULOS, NO podemos rellenar los registros.

Entendiendo esto pongamos un ejemplo de la tabla de Arriba

ID\_PEDIDO ID\_LINEA NOM\_ARTICULO CNT\_PEDIDO PRECIO IMP

206 1 Cochecitos 2000 0.50€ 4000€

En este caso podemos ver que han pedido 2000 cochecitos y el registro ha dependido totalmente de la tabla PEDIDOS y ARTICULOS para el registro.

TABLAS de EXTENSION:

CASOS ESPECIFICOS,

Mejor, no usarlas.

EJEMPLO REAL DE TABLAS:

Diagrama

Descripción generada automáticamente

Para entender esta BBDD, vamos a empezar con la tabla CLIENTES,

TABLAS:

CLIENTES:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | NAME | TLF | CIF | EMAIL | TOT\_PED |
| 1 | Víctor | 961553902 | 54426616M | victorbg189@gmail.com | 10 |
| 2 | Luisa | 123456789 | 12345678R | [emailejemplo@gmail.com](mailto:emailejemplo@gmail.com) | 50 |

Partiendo de estos campos: entendemos lo siguiente:

REGISTRO:

Víctor almacena un IDENTIFICADOR único con un NOMBRE, TELEFTONO, CIF, EMAIL y un total de PEDIDOS.

Índices:

Sirven para buscar registros, se pueden buscar por:

ID: 1

NOMBRE: Víctor

PALABRAS: Víctor

PARTES\_PALABRA: Vic -> Víctor

PUNTEROS:

Los punteros son SELECTS a un grupo/registro específicos:

ANT\_NOM: el puntero anterior al nombre es un puntero al REGISTRO ANTERIOR de su misma tabla.

SIG\_NOM: lo mismo que arriba pero para el REGISTRO SIGUIENTE.

PROBLEMA/OBJETIVO:

Queremos navegar entre los registros en un FRM:

BBDD SOL: Necesitamos dos punteros virtuales que apunten al siguiente/anterior registro,

APP SOL: FRM : cuando haga click sobre X btn, active un proceso o ejecute un comando para hacer referencia a ese CLIENTE.

ENLACES:

Son las relaciones:

N <-------------------> 1

N <-------------------> N