



CFGS ASIX/DAM/DAW
Mòdul 2: Bases de dades
Professor: Jordi Quesada Balaguer

INS Joan d'Àustria



**Consorci d'Educació
de Barcelona**
Generalitat de Catalunya
Ajuntament de Barcelona

DDL



- DDL = Data Definition Language
- Ens permetrà treballar amb:
 - Bases de dades
 - Taules
 - Index
 - Vistes



2.1. BASES DE DADES

DDL: DATABASES



- Els SGBD organitzen la informació de diferents maneres. Les més habituals són:
 - Schema: Conjunt de taules que pertany a un usuari
 - Base de dades: Conjunt de schemas.



- A MySQL no hi ha diferència entre aquests dos nivells i els tracta com a sinònims.

DDL: DATABASES



- Podem veure les bases de dades ja creades amb la comanda

show databases;

```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql        |
| performance_schema |
| sys          |
+-----+
4 rows in set (0.04 sec)

mysql> _
```

DDL: DATABASES



- Per crear una nova base de dades farem servir la comanda

`CREATE DATABASE <<nom>>;`

- Per als noms mysql és case sensitive a Linux. A windows no.



Crea una base de dades anomenada prova.
Comprova que s'ha creat correctament

DDL: DATABASES



- Les bases de dades ens permeten guardar estructures més petites anomenades taules.
- El primer que haurem de fer és indicar en quina base de dades volem treballar. Això es fa amb la comanda

`use <<nom>>;`

- Per exemple:

```
use prova;
```

```
mysql> use prova;  
Database changed  
mysql> _
```

DDL: DATABASES



- Si volem treballar amb una altra base de dades simplement tornem a fer la comanda use.
- Podem saber en quina base de dades estem amb la comanda:

```
select database();
```




2.2. TAULES

DDL: TAULES



- Les taules seran la manera de guardar les relacions del model relacional.
- La sintaxi bàsica és:

```
CREATE TABLE [IF NOT EXISTS] <<nom>> (  
    definició_columna1,  
    definició_columna2,  
  
    definició_columnaN,  
    restriccions  
);
```

DDL: TAULES



- If not exists fa una comprovació. Si la taula ja existia no la crea
- Cada columna representa un atribut de la relació.
- Però ara hem d'indicar més coses.
- Definició columna és:

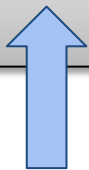
nom tipus [valor per defecte] [restricció_columna] [opcions]



2.2. TAULES

2.2.1. Tipus de dades

nom tipus [valor per defecte] [restricció_columna] [opcions]



DDL: TAULES: Tipus dades



- És a dir, per crear una columna haurem d'indicar el tipus de dades que volem guardar.
- El tipus de dades escollit ha de ser apropiat per poder guardar tots els valors possibles però de la forma més econòmica possible.
- Els tipus més bàsics que veurem són els de tipus numèric, text i dates

DDL: TAULES: Tipus dades



- Tipus numèrics:
 - Cal tenir present el rang i l'espai que ocupen.
 - Enters:
 - Accepten atribut UNSIGNED
 - Podem definir el tipus BOOLEAN que en realitat serà un TinyInt (1 = true i 0 = fals)

TIPUS DE DADES	ESPAI
TINYINT	1 byte
SMALLINT	2 bytes
MEDIUMINT	3 bytes
INT	4 bytes
BIGINT	8 bytes

DDL: TAULES: Tipus dades



- Tipus numèrics:
 - Reals:
 - Accepten atribut UNSIGNED
 - Els tipus són float (4 bytes) i double (8 bytes).
 - Podem indicar precissió i escala.
 - La precissió indica el número de dígit total
 - La escala indica el número de dígit de la part decimal
 - Per exemple Float(7,4) representa aproximadament entre el -999.9999 fins al 999.9999
 - Si no indiquem precissió i escala seran els màxims permesos per hardware

DDL: TAULES: Tipus dades



- Tipus text:
 - char(tamany):
 - Longitud fixa. Amb espais en blanc si és necessari
 - Tamany pot ser fins a 255
 - varchar(tamany):
 - Longitud variable.
 - text: Es divideix en:
 - TINYTEXT (màxima longitud 255 bytes)
 - TEXT (65535 bytes)
 - MEDIUMTEXT (16777215 bytes)
 - LONGTEXT ($4 \cdot 10^9$ bytes).

DDL: TAULES: Tipus dades



- Tipus text:
 - enum: Permet indicar una llista de valors possibles. El valor sempre serà un element d'aquests. Internament el valor es guardarà com un enter, però quan fem consultes es mostrarà el seu valor textual, per això permeten compactar informació.
 - Ens pot representar problemes quan ordenem.
 - Una alternativa serà utilitzar restriccions
 - Sintaxi: enum('Element1', 'Element2'...)

DDL: TAULES: Tipus dades



- Tipus text:
 - set: Permet indicar una llista de valors possibles. El valor sempre serà un o més element d'aquests.
 - Sintaxi: set('Element1', 'Element2'...)

DDL: TAULES: Tipus dades



- Tipus Dates:
 - Date: Representa una data en format YYYY-MM-DD
 - Time: Representa hores en format HH:MM:SS
 - DateTime: Combinació de les dos anteriors
 - Year: Permet guardar només l'any en dos formats: 2 dígitos o 4 dígitos: Year(2) o Year(4). Amb 2 dígitos podem guardar desde 1970 fins a 2069. Amb 4 dígitos podem guardar desde 1901 fins a 2155

DDL: TAULES



Crea una taula anomenada t1.

Volem que tingui els següents atributs:

- DNI
- Nom complet
- Edat
- Sexe
- Sou
- email
- Telèfon
- Es_estudiant



Crea una taula anomenada t2 que tingui:

- num-habitacio
- tipus amb valors possibles doble, triple o suite.



2.2. TAULES

2.2.2. Valor per defecte

nom tipus [valor per defecte] [restricció_columna] [opcions]



DDL: TAULES: Default



- Com veurem més endavant, en el moment que introduïm informació a la base de dades, no sempre s'assignen tots els valors.
- Quan això passa la base de dades assignarà un valor especial anomenat NULL.
- El valor NULL indica desconegut.

DDL: TAULES: Default



- Quan definim una columna podem indicar el valor per defecte que volem que tingui si no indiquem un altre valor.
- Això es fa amb la forma: DEFAULT <valor>
- El valor ha de ser del tipus de la columna

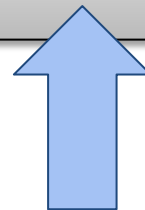
```
CREATE TABLE producte (  
    codi INT,  
    nom varchar(20) DEFAULT 'Desconegut',  
    preu double(10,2) DEFAULT 0.00  
);
```



2.2. TAULES

2.2.3. Restriccions de columna

nom tipus [valor per defecte] [restricció_columna] [opcions]



DDL: TAULES: Restriccions



- Quan definim les columnes també podem indicar una sèrie de normes que s'han de complir
- Aquestes normes s'anomenen restriccions

DDL: TAULES: Restriccions



- Quan definim les columnes també podem indicar una sèrie de normes que s'han de complir
- Aquestes normes s'anomenen restriccions



DDL: TAULES: Restriccions



- Les restriccions disponibles són:
 - NULL | NOT NULL
 - PRIMARY KEY = KEY
 - UNIQUE = UNIQUE KEY
 - CHECK: Des de la versió 8.0.16
- Una taula només pot tenir una clau primària
- La clau primària no accepta duplicats ni valors NULL
- Unique en canvi si accepta valors NULL.
Podem tenir varies columnes UNIQUE

DDL: TAULES: Restriccions



Crea una taula anomenada t3 que tingui:

- DNI com a clau primària
- nom obligatori
- email obligatori i amb control de repetits
- sou obligatori i com a valor mínim 400

Crea una taula anomenada t4 que tingui:

- num-habitacio clau primària
- tipus amb valors possibles doble, triple o suite. Utilitza check



2.2. TAULES

2.2.4. Opcions de columna

nom tipus [valor per defecte] [restricció_columna] [opcions]



DDL: TAULES: Opcions



- Les opcions possibles quan definim una columna són:
 - **AUTO_INCREMENT**: MySQL assignarà automàticament el valor de manera incremental. Evidentment el camp ha de ser de tipus enter i a més ha de ser clau primària. Només es permet un **AUTO_INCREMENT** per taula.
 - **COMMENT**: Permet indicar un comentari

DDL: TAULES



Crea una taula anomenada t5 que tingui:


- codi clau primària autoincremental
- model obligatori
- marca obligatori
- matrícula clau alternativa, obligatori
- bastidor clau alternativa, no obligatori
- color no obligatori



2.2. TAULES

2.2.5. Restriccions de taula

```
CREATE TABLE [IF NOT EXISTS] <<nom>> (  
    definició_columna1,  
    definició_columna2,  
  
    definició_columnaN,  
    restriccions_taula  
);
```



DDL: TAULES: Restriccions



- Intenta fer el següent exercici:



Crea una taula anomenada t6 que tingui:

- nom-hotel clau primària
- num-habitacio clau primària
- tipus

DDL: TAULES: Restriccions



- Sovint ens trobem que hi ha restriccions que s'han d'aplicar a més d'una columna
- A nivell de taula tenim més restriccions que a nivell de columna (excepte NOT NULL)
- Mantenir d'una banda la definició de les columnes i d'altra les restriccions que hi ha facilita la lectura del codi
- Així, normalment, totes les restriccions (excepte NOT NULL) s'apliquen a nivell de taula i no de columna

DDL: TAULES: Restriccions



- Les restriccions disponibles a nivell de taula seguiran el format:

[CONSTRAINT <<nom>>] tipus (columnes)

- és molt recomanable posar nom a les restriccions

DDL: TAULES: Restriccions



- Les restriccions disponibles són:
 - PRIMARY KEY
 - UNIQUE
 - CHECK
 - FOREIGN KEY

Crea una taula anomenada t6 que tingui:

- nom-hotel clau primària
- num-habitacio clau primària
- tipus

DDL: TAULES: Restriccions



- La restricció de clau forana és la que ens permetrà relacionar taules.
- En concret implica una o més columnes d'una taula “filla” que apuntaran a una o més columnes d'una taula “pare”
- La sintaxis és:

```
[CONSTRAINT nom] FOREIGN KEY (col_name, ...)
REFERENCES TAULA (col_name, ...)
[ON DELETE POLÍTICA]
[ON UPDATE POLÍTICA]
```

DDL: TAULES: Restriccions



- Per exemple:

```
CREATE TABLE PERSONA (  
    DNI CHAR(9),  
    Nom varchar(20),  
    Constraint PK_PERSONA PRIMARY KEY(DNI)  
);  
  
CREATE TABLE COCHE(  
    matricula CHAR(7),  
    marca varchar(20),  
    dni_propietari char(9),  
    Constraint PK_COCHE PRIMARY KEY(matricula),  
    Constraint FK_COCHE_PERS FOREIGN KEY  
(dni_propietari) REFERENCES PERSONA (DNI)  
);
```

DDL: TAULES: Restriccions



- Podem indicar les accions a realitzar en el cas de borrrat de files i de modificació de valors

```
[CONSTRAINT nom] FOREIGN KEY (col_name, ...)
REFERENCES TAULA (col_name, ...)
[ON DELETE POLÍTICA]
[ON UPDATE POLÍTICA]
```

- La *POLÍTICA* pot ser: RESTRICT, CASCADE o SET NULL
- Si no indiquem s'aplica RESTRICT