Contenu

FONCTION DE BASE

MySQL -u root -p	/* cnx base
show databases;	/* affiche listes des bases
create database exemple;	/* création de base
use nombase;	/* utlisation base
describe nombase	/* affiche la table avec champs

VARIABLES types données

Numérique

- TINYINT [M] [UNSIGNED]
  Occupe 1 octet. Ce type peut stocker des nombres entiers de -128 à 127 s'il ne porte pas l'attribut UNSIGNED, dans le cas contraire il peut stocker des entiers de 0 à 255.

- SMALLINT [M] [UNSIGNED]
  Occupe 2 octets. Ce type de données peut stocker des nombres entiers de -32 768 à 32 767 si il ne porte pas l'attribut UNSIGNED, dans le cas contraire il peut stocker des entiers de 0 à 65 535.

- MEDIUMINT [M] [UNSIGNED]
  Occupe 3 octets. Ce type de données peut stocker des nombres entiers de -8 388 608 à 8 388 607 si il ne porte porte pas l'attribut UNSIGNED, dans le cas contraire il peut stocker des entiers de 0 à 16 777 215.

- INT [M] [UNSIGNED]
  Occupe 4 octets. Ce type de données peut stocker des nombres entiers de -2 147 483 648 à 2 147 483 647 si il ne porte pas l'attribut UNSIGNED, dans le cas contraire il peut stocker des entiers de 0 à 4 294 967 295.

- INTEGER [M] [UNSIGNED]
  Même chose que le type INT.

- BIGINT [M] [UNSIGNED]
  Occupe 8 octets. Ce type de données stocke les nombres entiers allant de -9 223 372 036 854 775 808 à 9 223 372 036 854 775 807 sans l'attribut UNSIGNED, et de 0 à 18 446 744 073 709 551 615 avec.

- FLOAT (précision simple de 0 à 24 et précision double de 25 à 53) [UNSIGNED]
  Occupe 4 octets si la précision est inférieure à 24 ou 8 au-delà.
  Stocke un nombre de type flottant.

- FLOAT[(M,D)] [UNSIGNED]
  Occupe 4 octets. M est le nombre de chiffres et D est le nombre de décimales.
  Ce type de données permet de stocker des nombres flottants à précision simple. Va de -1.175494351E-38 à 3.402823466E+38. Si UNSIGNED est activé, les nombres négatifs sont retirés mais ne permettent pas d'avoir des nombres positifs plus grands.

- DOUBLE PRECISION[(M,D)]
  Occupe 8 octets. Même chose que le type DOUBLE

- DOUBLE [(M,D)]
  Occupe 8 octets. Stocke des nombres flottants à double précision de -1.7976931348623157E+308 à -2.2250738585072014E-308, 0, et de 2.2250738585072014E-308 à 1.7976931348623157E+308.
  Si UNSIGNED est activé, les nombres négatifs sont retirés mais ne permettent pas d'avoir des nombres positifs plus grands.

- REAL[(M,D)]
  Occupe 8 octets. Même chose que le type DOUBLE

- DECIMAL ou NUMERIC [(M,D)]
  Occupe M+2 octets si D > 0, M+1 octets si D = 0
  Contient des nombres flottants stockés comme des chaînes de caractères.

Date

- DATE
  Occupe 3 octets. Stocke une date au format 'AAAA-MM-JJ' allant de '1000-01-01' à '9999-12-31'

- DATETIME
  Occupe 8 octets. Stocke une date et une heure au format 'AAAA-MM-JJ HH:MM:SS' allant de '1000-01-01 00:00:00' à '9999-12-31 23:59:59'

- TIMESTAMP [M]
  Occupe 4 octets. Stocke une date sous forme numérique allant de '1970-01-01 00:00:00' à l'année 2037. L'affichage dépend des valeurs de M : AAAAMMJJHHMMSS, AAMMJJHHMMSS, AAAAMMJJ, ou AAMMJJ pour M égal respectivement à 14, 12, 8, et 6

- TIME
  Occupe 3 octets. Stocke l'heure au format 'HH:MM:SS', allant de '-838:59:59' à '838:59:59'

- YEAR
  Occupe 1 octet. Année à 2 ou 4 chiffres allant de 1901 à 2155 (4 chiffres) et de 1970-2069 (2 chiffres).

Caractères

- CHAR(M) [BINARY]
  Occupe M octets, M allant jusqu'à 255
  Chaîne de 255 caractères maximum remplie d'espaces à la fin. L'option BINARY est utilisée pour tenir compte de la casse.

- BIT
  Occupe 1 octet. Même chose que CHAR(1)

- BOOL
  Occupe 1 octet. Même chose que CHAR(1)

- CHAR (M)
  Occupe M octets. Stocke des caractères. Si vous stockez un caractère et que M vaut 255, la donnée prendra 255 octets. Autant donc employer ce type de données pour des mots de longueur identique.

- VARCHAR (M) [BINARY]
  Occupe L+1 octets (ou L représente la longueur de la chaîne).
  Ce type de données stocke des chaînes de 255 caractères maximum. L'option BINARY permet de tenir compte de la casse.

- TINYBLOB (L représente la longueur de la chaîne)
  Occupe L+1 octets.
  Stocke des chaînes de 255 caractères maximum. Ce champ est sensible à la casse.

- TINYTEXT
  Occupe L+1 octet.
  Stocke des chaînes de 255 caractères maximum. Ce champ est insensible à la casse.

- BLOB
  Occupe L+1 octet.
  Stocke des Chaînes de 65535 caractères maximum. Ce champ est sensible à la casse.

- TEXT
  Occupe L+2 octets.
  Stocke des chaînes de 65535 caractères maximum. Ce champ est insensible à la casse.

- MEDIUMBLOB
  Occupe L+3 octets.
  Stocke des chaînes de 16777215 caractères maximum.

- MEDIUMTEXT
  Occupe L+3 octets.
  Chaîne de 16 777 215 caractères maximum. Ce champ est insensible à la casse.

- LONGBLOB
  Occupe L+4 octets.

Stocke des chaînes de 4 294 967 295 caractères maximum. Ce champ est sensible à la casse.

- LONGTEXT
  Occupe L+4 octets.
  Stocke des chaînes de 4 294 967 295 caractères maximum.

- ENUM('valeur_possible1','valeur_possible2','valeur_possible3',...)
  Occupe 1 ou 2 octets (la place occupée est fonction du nombre de solutions possibles : 65 535 valeurs maximum.

- SET('valeur_possible1','valeur_possible2',...)
  Occupe 1, 2, 3, 4 ou 8 octets, selon de nombre de solutions possibles (de 0 à 64 valeurs maximum)

CREATION

```
create table nombase(                    /* création de la table avec champs
-> nom varchar(25),
-> ...
);
```

OPTIONS CREATION CHAMP

```
primary key
auto_increment
not null
default '...'
```

COPY

```
create table nombase1 like nombase2;                    /* copy de table sans data
create table nombase1 as select * from nombase2;        /* copy table avec data
```

AJOUT DE CHAMPS

```
alter table nombase add tel varchar(24);                /* ajoutr d'un champ
alter table nombase modify tel varchar(24);             /* modifie un champ
alter tables nombase change tel mobile varchar(24);     /* modifie le nom d'un champ
alter table nombase modify auto_increment=Nbr;          /* modifie l'incrémentation
```

VIDE OU SUPPRIME LA TABLE

```
truncate table nombase;                 /* vide la table
drop database namebase, namebase2...;   /* supp de la base
```

RENOMME LA TABLE

```
rename table test00 to testbis                  /* renomme la table
rename table test00 to autrebase.toto;          /* renomme la table et la déplace
```

INSERTION

insert into nombase values (0,'rolland','didier'); /* insert une ligne

insert into nombase (champ,...) values ('valeur1',...);

insert into stagiaires (nom,prenom,email) values ('rolland','noah','noah@gmail.com'),

-> ('rolland','didier','didier@gmail.com'),

-> ('rolland','lily','lily@gmail.com'),

LES FONCTIONS D'AGREGATIONS

Fonction	Description	Compatibilité
AVG(champ)	retourne la moyenne de "champ" sur l'ensemble des enregistrements	MySQL PostgreSQL
BIT_AND(champ)	retourne le résultat d'un "et logique" appliqué sur l'ensemble des enregistrements	MySQL
BIT_OR(champ)	retourne le résultat d'un "ou logique" appliqué sur l'ensemble des enregistrements	MySQL
COUNT(champ)	retourne le nombre d'enregistrements	MySQL PostgreSQL
COUNT(DISTINCT champ)	retourne le nombre d'enregistrements distincts	MySQL
MAX(champ)	retourne la plus grande valeur de "champ" parmi la liste des enregistrements sélectionnés (s'applique également à des champs alphanumériques)	MySQL PostgreSQL
MIN(champ)	retourne la plus petite valeur parmi la liste des enregistrements sélectionnés (s'applique également à des champs alphanumériques)	MySQL PostgreSQL
STD(champ)	retourne l'ecart-type de "champ" sur l'ensemble des enregistrements	MySQL
STDDEV(champ)	retourne l'écart-type de "champ" sur l'ensemble des	MySQL

	enregistrements	
SUM(champ)	retourne la somme de "champ" sur l'ensemble des enregistrements	MySQL PostgreSQL

FONCTIONS TRAVAIL SUR BASE

select * from nombase; /* affiche toute la base

select database(); /* vois sur quelle base on ai

select now() as 'date to day :'; /* affiche la date du jour

select concat (nom, ' ', prenom) as texteaafficher from base; /* concatenise et affiche nom et prenom

select upper(nom) as le texte from table; /* colonne affiché en MAJ

select nom, (L ou R)pad(nom, 15,'.') from tablebase /* ajout des points avant ou aprés sur total 15 carac

% > remplace 0 a l'infinie

_ > remplace 1 caractère

distinct > retire les doublons

select @@lc_time_names; /* quelle langues

set lc_time_names = "fr_FR"; /* format francais

delete from table where id 6; /* supprime la ligne 6

update base set nom_col ="valeur" where condition /* modif sur ligne existantes

SELECT `id`, GROUP_CONCAT(`nom_colonne`) FROM `table` GROUP BY `id`

SELECT colonne1, colonne2 FROM table ORDER BY colonne1

SELECT * FROM table WHERE nom_colonne BETWEEN 'valeur1' AND 'valeur2'

SELECT nom_colonne FROM table WHERE nom_colonne IN (valeur1, valeur2, valeur3, ...)

FICHIER VIA VSCODE

source c:\dossier\nom.sql /* execute un fichier sql dans console

/* fichier dans vscode */

create table stagiaires(

id int primary key auto_increment,

nom varchar(32) not null,

prenom varchar(32) not null,

email varchar(64),

pseudo varchar(16),

couleur varchar(12) default 'Rose'

);

select * from base where penom="toto"; /* affiche que les prenoms toto

JOINTURE TABLES

- **INNER JOIN :** jointure interne pour retourner les enregistrements quand la condition est vrai dans les 2 tables. C'est l'une des jointures les plus communes.
- **CROSS JOIN :** jointure croisée permettant de faire le produit cartésien de 2 tables. En d'autres mots, permet de joindre chaque lignes d'une table avec chaque lignes d'une seconde table. Attention, le nombre de résultats est en général très élevé.
- **LEFT JOIN (ou LEFT OUTER JOIN) :** jointure externe pour retourner tous les enregistrements de la table de gauche (LEFT = gauche) même si la condition n'est pas vérifié dans l'autre table.
- **RIGHT JOIN (ou RIGHT OUTER JOIN) :** jointure externe pour retourner tous les enregistrements de la table de droite (RIGHT = droite) même si la condition n'est pas vérifié dans l'autre table.
- **FULL JOIN (ou FULL OUTER JOIN) :** jointure externe pour retourner les résultats quand la condition est vrai dans au moins une des 2 tables.
- **SELF JOIN :** permet d'effectuer une jointure d'une table avec elle-même comme si c'était une autre table.
- **NATURAL JOIN :** jointure naturelle entre 2 tables s'il y a au moins une colonne qui porte le même nom entre les 2 tables SQL
- **UNION JOIN :** jointure d'union

```
select * from table T inner join table2 T2 on T2.id_pers = T.id;
```

(T est alias)

```
select * from table left outer join table2 on table.id = table2.id_pers ;

select * from table right outer join table2 on table.id = table2.id_pers ;

select nom from table1 inner join table2 on table1.champ1 = table2.champ2 order by tables2.champ;

select nom, count(DISTINCT champ) from table1 inner join table2 on table1.champ1 = table2.champ2 group by table2 ;
```

select nom, count(DISTINCT champ) from table1 inner join table2 on table1.champ1 = table2.champ2 where table1.champ1 = 'xxx' and nom not like '%x%';