CREACIÓ DE TAULES

Escollir el tipus de dades correcte: [Working with Data in MySQL]

Tipus de dades	

En Mysql els tipus de dades no són els mateixos que en Oracle, a la dreta es mostra la relació d'equivalència entre els dos DBMS.

En els següents links podeu consultar més informació sobre això :

[Oracle and MySQL Compared]

MySQL Data Type	Oracle Data Type
BIGINT	NUMBER(19, 0)
CHAR	CHAR
DATE	DATE
DATETIME	DATE
DECIMAL	FLOAT (24)
DOUBLE	FLOAT (24)
DOUBLE PRECISION	FLOAT (24)
ENUM	VARCHAR2
FLOAT	FLOAT
INT	NUMBER(10, 0)
INTEGER	NUMBER(10, 0)
MEDIUMINT	NUMBER(7, 0)
NUMERIC	NUMBER
REAL	FLOAT (24)
SET	VARCHAR2
SMALLINT	NUMBER(5, 0)
TIME	DATE
TIMESTAMP	DATE
TINYINT	NUMBER(3, 0)
TINYTEXT	VARCHAR2
VARCHAR	VARCHAR2, CLOB
YEAR	NUMBER

MySQL Data Type	Oracle Data Type
BIT	RAW
BLOB	BLOB, RAW
LONGBLOB	BLOB, RAW
LONGTEXT	CLOB, RAW
MEDIUMBLOB	BLOB, RAW
MEDIUMTEXT5	CLOB, RAW
TEXT	VARCHAR2, CLOB
TINYBLOB	RAW

CREATE TABLE
En els següents links podeu consultar més informació sobre les opcions per crear taules en mysql: MySQL Create Table - javatpoint MySQL :: MySQL 8.0 Reference Manual :: 13.1.20 CREATE TABLE Statement

Exemples	

Creació d'una taula MySQL
 	
`CREATE TABLE IF NOT EXISTS task (`
`  task_id INT AUTO_INCREMENT PRIMARY_KEY,`
`  title VARCHAR(255) NOT NULL,`
`  start_date DATE,`
`  due_date DATE,`
`  status TINYINT NOT NULL,`
`  priority TINYINT NOT NULL,`
`  description TEXT,`
`  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP`
`);`
	The tasks table has the following columns:
	
	• The task_id is an **auto-increment** column. If you use the INSERT statement to insert a new row into the table without specifying a value for the task_id column, MySQL will automatically generate a sequential integer for the task_id starting from 1.
	
	• The title column is a variable character string column whose maximum length is 255. It means that you cannot insert a string whose length is greater than 255 into this column. The NOT NULL constraint indicates that the column does not accept NULL. In other words, you have to provide a non-NULL value when you insert or update this column.
	
	• The start_date and due_date are DATE columns. Because these columns do not have the NOT NULL constraint, they can store NULL. The start_date column has a default value of the current date. In other words, if you don’t provide a value for the start_date column when you insert a new row, the start_date column will take the current date of the database server.
	
	• The status and priority are the TINYINT columns which do not allow NULL.

Exemples	
	• The description column is a TEXT column that accepts NULL. • The created_at is a TIMESTAMP column that accepts the current time as the default value. • The task_id is the primary key column of the tasks table. It means that the values in the task_id column will uniquely identify rows in the table.
Modificació d'una taula per afegir o treure **COLUMNES** MySQL: ALTER TABLE	

`ALTER TABLE contacts`
`  ADD last_name varchar(40) NOT NULL`
`  AFTER contact_id;`
	• Aquest exemple de MySQL ALTER TABLE afegirà una columna anomenada *last_name* a la taula de *contacts*.
	• Es crearà com a columna NOT NULL i apareixerà després del camp *contact_id* de la taula.

`ALTER TABLE contacts` `  NOIDIFY last_name varchar(50) NULL;`	• Aquest exemple d'ALTER TABLE modificarà la columna anomenada *last_name* perquè sigui un tipus de dades de varchar(50) i obligarà la columna a permetre valors NULL.
`ALTER TABLE contacts` `  DROP COLUMN contact_type;`	• Aquest exemple d'ALTER TABLE eliminarà la columna anomenada *contact_type* de la taula anomenada *contacts*.

`ALTER TABLE contacts`
`  change COLUMN contact_type ctype`
`  varchar(20) NOT NULL;`
	• Aquest exemple de MySQL ALTER TABLE canviarà el nom de la columna anomenada *contact_type* a *ctype*. La columna es definirà com una columna varchar(20) NOT NULL.

Exemples	
Modificació d'una taula MySQL, afegir o treure PRIMARY KEY: ALTER TABLE	
Modificació d'una taula MySQL per afegir o treure UNIQUE constraints: ALTER TABLE • La restricció UNIQUE garanteix que tots els valors d'una columna siguin diferents. • Tant les restriccions UNIQUE com PRIMARY KEY ofereixen una garantia d'unicitat per a una columna o conjunt de columnes. • Una restricció PRIMARY KEY té automàticament una restricció UNIQUE. • Tanmateix, podeu tenir moltes restriccions UNIQUE per taula, però només una restricció PRIMARY KEY per taula. • Alguns dels camps que formen part de la restricció UNIQUE poden contenir valors nuls sempre que la combinació de valors sigui única.	

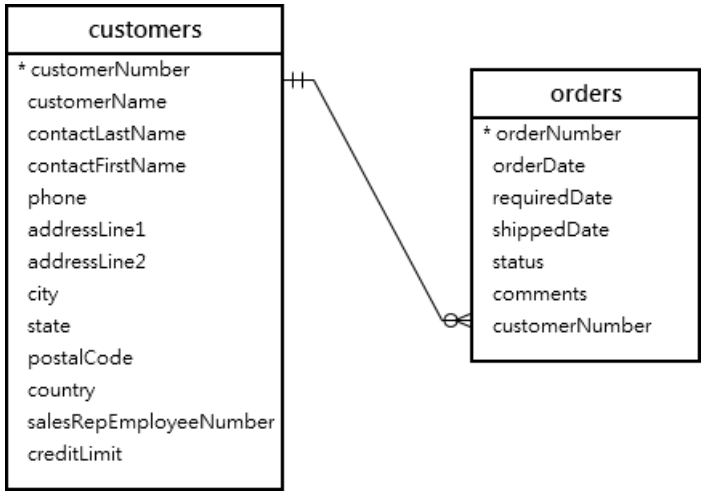
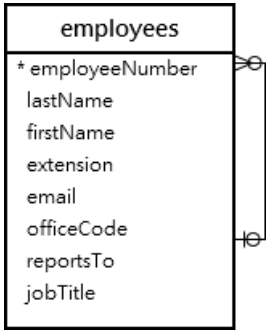
`ALTER TABLE contacts`
`  ADD CONSTRAINT reference_unique`
`  UNIQUE (referenc_number);`
 	
`ALTER TABLE contacts`
`  ADD CONSTRAINT contact_name_unique`
`  UNIQUE (last_name, first_name);`
	• En aquest exemple, hem creat una restricció única a la taula de *contacts* existent anomenada reference_unique. Consisteix en el camp anomenat *reference_number*.
	• També podríem crear una restricció única amb més d'un camp com es mostra al segon exemple.

`ALTER TABLE contacts` `  DROP CONSTRAINT reference_unique;`	• En aquest exemple, estem eliminant una restricció única a la taula de *contacts* anomenada *reference_unique*.

TAULES - Restriccions

Restriccions (constraints)
Table constraints 1. NOT NULL: Ensures that the value of the column must not be null 2. CHECK: Before inserting data in the table, it evaluates the condition specified in the CHECK constraint. If the condition fails, then the insert statement fails 3. DEFAULT: Default values of the column. If you do not specify the value of the column in the insert statement, the query inserts the value specified in the DEFAULT constraint **Primary and Foreign keys,** once columns are defined, you can create primary key and foreign keys using the following keywords 1. PRIMARY KEY: It's a unique index and must be defined as NOT NULL. A table can have only one primary key. The PRIMARY KEY is placed first in the create table statement 2. FOREIGN KEY: MySQL supports the foreign keys. A table can have more than one foreign key that references the primary key of different tables
Exemples de CHECK constraints: https://www.w3schools.com/mysql/mysql_check.asp

TAULES - Foreign Keys

Exemples de relacions - Claus foranes	
	En aquest diagrama: • cada client (*customers*) pot tenir zero o moltes comandes i • cada comanda (*orders*) pertany a un client. La relació entre la taula *customers* i la taula *orders* és d'un a molts, aquesta relació s'estableix definint una FOREIGN KEY a la taula *orders*: • a la taula *customers* hi ha una columna *customerNumber*, que es la seva PRIMARY KEY.. • a la taula *orders* hi ha també la columna *customerNumber* que s'utilitzarà per definir la FK cap a la taula *customers*. La taula de *customers* s'anomena taula pare, i la taula de *orders* es coneix com a taula filla
	De vegades, quan hi ha una relació reflexiva, la taula pare i la taula filla són la mateixa. En aquest cas, la clau forana fa referència a la clau primària dins de la mateixa taula. La columna *reportTo* és una FOREIGN KEY que fa referència a la columna *employeeNumber*, que és la PRIMARY KEY. de la taula *employees*. Aquesta FK permet que la taula *employees* emmagatzemi la relació de reporting entre empleats i directius. Cada empleat reporta a zero o un empleat(directiu) i un empleat(directiu) pot tenir zero o molts subordinats.

Definició de claus foranes a una taula MySQL: CONSTRAINT FOREIGN KEY	

Sintaxi de la definició de constraints en una sentència CREATE TABLE :
 	
`[CONSTRAINT constraint_name]`
`FOREIGN KEY [foreign_key_name] (column_name, ...)`
`REFERENCES parent_table(colunm_name,...)`
`[ON DELETE reference_option]`
`[ON UPDATE reference_option]`
	First, specify the name of foreign key constraint that you want to create after the CONSTRAINT keyword. If you omit the constraint name, MySQL automatically generates a name for the foreign key constraint.
	
	Second, specify a list of comma-separated foreign key columns after the FOREIGN KEY keywords. The foreign key name is also optional and is generated automatically if you skip it.
	
	Third, specify the parent table followed by a list of comma-separated columns to which the foreign key columns reference.
	
	Finally, specify how foreign key maintains the referential integrity between the child and parent tables by using the ON DELETE and ON UPDATE clauses. The reference_option determines action which MySQL will take when values in the parent key columns are deleted (ON DELETE) or updated (ON UPDATE).
	
	MySQL has five reference options: CASCADE, SET NULL, NO ACTION, RESTRICT, and SET DEFAULT.
	
	•CASCADE: if a row from the parent table is deleted or updated, the values of the matching rows in the child table automatically deleted or updated.
	•SET NULL: if a row from the parent table is deleted or updated, the values of the foreign key column (or columns) in the child table are set to NULL.
	•RESTRICT: if a row from the parent table has a matching row in the child table, MySQL rejects deleting or updating rows in the parent table.
	•NO ACTION: is the same as RESTRICT.
	•SET DEFAULT: is recognized by the MySQL parser. However, this action is rejected by both InnoDB and NDB tables.
	
	In fact, MySQL fully supports three actions: RESTRICT, CASCADE and SET NULL.
	
	If you don't specify the ON DELETE and ON UPDATE clause, the default action is RESTRICT.

Definició de claus foranes a una taula MySQL: CONSTRAINT FOREIGN KEY

```
CREATE TABLE categories(
  categoryId INT AUTO_INCREMENT PRIMARY KEY,
  categoryName VARCHAR(100) NOT NULL
) ENGINE = INNODB;


CREATE TABLE products(
  productId INT AUTO_INCREMENT PRIMARY KEY,
  productName VARCHAR(100) NOT NULL,
  categoryId INT,
  CONSTRAINT fk_category FOREIGN KEY (categoryId)
                REFERENCES categories(categoryId)
) ENGINE = INNODB;
```

- El *categoryId* de la taula de *products* és la columna de FOREIGN KEY que fa referència a la columna *categoryId* de la taula de *categories*.
- Com que no especifiquem cap clàusula ON UPDATE i ON DELETE, l'acció predeterminada és RESTRICT tant per a l'operació d'UPDATE com per a la DELETE.
- **NO** podrem esborrar una fila de la taula pare *categories* si té fills a *products*.
- **NO** podrem modificar la clau primária *categoryId* de la taula pare *categories* si té fills.
  - `Error Code: 1451. Cannot delete or update a parent row: a foreign key constraint fails (`fkdemo`.`products`, CONSTRAINT `fk_category` FOREIGN KEY (`categoryId`) REFERENCES `categories` (`categoryId`) ON DELETE RESTRICT ON UPDATE RESTRICT)`
- **NO** podrem inserir un fill a *products* amb un *categoryId* que no estigui a la taula pare *categories*.
  - `Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails (`fkdemo`.`products`, CONSTRAINT `fk_category` FOREIGN KEY (`categoryId`) REFERENCES `categories` (`categoryId`) ON DELETE RESTRICT ON UPDATE RESTRICT)`

Definició de claus foranes a una taula MySQL: CONSTRAINT FOREIGN KEY

```
CREATE TABLE products(
  productId INT AUTO_INCREMENT PRIMARY KEY,
  productName VARCHAR(100) NOT NULL,
  categoryId INT,
  CONSTRAINT fk_category FOREIGN KEY (categoryId)
                REFERENCES categories(categoryId)
                ON UPDATE CASCADE
                ON DELETE CASCADE
) ENGINE = INNODB;
```

Integritat referencial – Opció **CASCADE**

- UPDATE categories SET categoryId = 100 WHERE categoryId = 1;
  - Si hi ha files amb el valor 1 a la columna *categoryId* a la taula de *products* s'han actualitzat automàticament a 100 a causa de l'acció ON UPDATE CASCADE.
- DELETE FROM categories WHERE categoryId = 2;
  - Si hi ha files amb el *categoryId* 2 a la taula de *products* s'han esborrat automàticament a causa de l'acció ON DELETE CASCADE.

```
CREATE TABLE products(
  productId INT AUTO_INCREMENT PRIMARY KEY,
  productName VARCHAR(100) NOT NULL,
  categoryId INT,
  CONSTRAINT fk_category FOREIGN KEY (categoryId)
                REFERENCES categories(categoryId)
                ON UPDATE SET NULL
                ON DELETE SET NULL
) ENGINE = INNODB;
```

Integritat referencial – Opció **SET NULL**

- UPDATE categories SET categoryId = 100 WHERE categoryId = 1;
  - Si hi ha files amb el valor 1 a la columna *categoryId* a la taula de *products* es canviaran automàticament a NULL a causa de l'acció ON UPDATE SET NULL.
- DELETE FROM categories WHERE categoryId = 2;
  - Si hi ha files amb el *categoryId* 2 a la taula de *products* es canviaran automàticament a NULL a causa de l'acció ON DELETE SET NULL..

<table>
<tr><th colspan="2">Definició de claus foranes a una taula MySQL: CONSTRAINT FOREIGN KEY</th></tr>
<tr><td>

```
ALTER TABLE table_name
  ADD CONSTRAINT constraint name
    FOREIGN KEY foreign_key_name (column_name, ...)
    REFERENCES parent_table(colunm_name,...)
[ON DELETE reference_option]
[ON UPDATE reference_option]
```

</td><td>
Altres exemples de definició de FK amb un ALTER TABLE:

creem una columna nova a una taula:
<pre>ALTER TABLE exam ADD COLUMN student_id INT;</pre>
creem la relació d'integritat per aquesta nova columna:
<pre>ALTER TABLE exam ADD CONSTRAINT fk_student_id
FOREIGN KEY(student_id) REFERENCES student(student_id);</pre>
no s'ha indicat cap acció per DELETE o UPDATE així que s'aplicarà la de defecte que es RESTRICT

</td></tr>
<tr><td>

```
ALTER TABLE table_name
DROP FOREIGN KEY constraint_name;
```

</td><td>
Per eliminar una constraint es fa servir DROP:
<pre>ALTER TABLE exam DROP FOREIGN KEY fk_student_id;</pre>
</td></tr>
</table>

INFORMACIÓ DE LA BASE DE DADES

Diccionari de dades	Exemples
INFORMATION_SCHEMA és una BD d'informació de MYSQL INFORMATION_SCHEMA és la base de dades d'informació, que emmagatzema informació sobre totes les altres bases de dades que manté el servidor MySQL . Dins l'INFORMATION_SCHEMA hi ha diverses taules de només lectura. Cada usuari MySQL té dret a accedir a aquestes taules, però només als registres que corresponen als objectes als quals té permís d'accés.	Les següents comandes són equivalents: `SELECT SCHEMA_NAME AS `Database`   FROM INFORMATION_SCHEMA.SCHEMATA   [WHERE SCHEMA_NAME LIKE 'wild']` `SHOW DATABASES   [LIKE 'wild']` `SELECT table_name FROM INFORMATION_SCHEMA.TABLES   [WHERE table_schema = 'db_name']   [WHERE|AND table_name LIKE 'wild']` `SHOW TABLES   [FROM db_name]   [LIKE 'wild']` `SELECT COLUMN_NAME, DATA_TYPE, IS_NULLABLE, COLUMN_DEFAULT   FROM INFORMATION_SCHEMA.COLUMNS   WHERE table_name = 'tbl_name'   [AND table_schema = 'db_name']   [AND column_name LIKE 'wild']` `SHOW COLUMNS   FROM tbl_name   [FROM db_name]   [LIKE wild]`

Webgrafia

Enllaços web	
MySQL :: MySQL 8.0 Reference Manual :: 13.1.20.5 FOREIGN KEY Constraints	FOREIGN KEY Constraints
MySQL :: MySQL 8.0 Reference Manual :: 26 INFORMATION_SCHEMA Tables	INFORMATION_SCHEMA Tables
Capítulo 22. La base de datos de información INFORMATION_SCHEMA (guebs.com)	La base de datos de información INFORMATION_SCHEMA