

Lenguajes de definición y modificación de datos SQL

Relación	Conjunto de filas y columnas determinando entidades
Tuplas	Conjunto de valores para cada atributo de la relación
Columnas	Atributos o propiedades de la relación
Grado	Nº de columnas de la relación
Cardinalidad	Nº de filas de la relación
Dominio	Conjunto de valores permitidos para un atributo
Nivel físico	Modo de almacenamiento físico de los datos
Nivel Lógico o conceptual	Descripción a nivel lógico de los datos mediante la especificación de las entidades, atributos, relaciones, y restricciones
Nivel externo o vistas	Subconjunto de los datos de la base de datos requeridas por el usuario
Definición o descripción [DDL]	Especificar elementos de datos que la integran, su estructura y relaciones entre ellas, reglas de integridad y confidencialidad
Manipulación [DML]	Inserción de nuevos datos y modificación y borrado o eliminación de datos existentes
Control [DCL]	Utilidades de gestión de usuarios y permisos y de administración del sistema (copias de seguridad...)

Lenguajes de definición y modificación de datos SQL

DDL

CREATE	Creación de los diferentes objetos
ALTER	Modificación de objetos creados anteriormente
DROP	Eliminación de objetos creados anteriormente

DML

SELECT	Selección de vista de la base de datos
INSERT	Inserción de tuplas en la base de datos
UPDATE	Modificación de tuplas en la base de datos
DELETE	Eliminación de tuplas en la base de datos

Lenguajes de definición y modificación de datos SQL

Crear base de datos/esquema

```
CREATE {DATABASE|SCHEMA}  
[IF NOT EXISTS] nombre_BD  
[DEFAULT] CHARA  
CTER SET nombre_caracteres  
[DEFAULT] COLLATE nombre_cotejamiento
```

Create database pedidos
Character set latin1
Collate latin1_spanish_ci;

Crear relación / tabla

```
CREATE TABLA  
[IF NOT EXISTS] nombre_tabla  
([columna1 definicion_columna1] [restricciones_columna1],  
[columna2 definicion_columna2] [restricciones_columna2],  
...  
[columnan definicion_columnan] [restricciones_columnan],  
[restriccion_tabla1], [restricción_tabla2],...[restricción_tablan]  
[opciones_tabla]);
```

Use pedidos;
Create table pedido
(RefPed char(5) primary key,
FecPed date not null);

Use pedidos;
Create table articulo
(CodArt char(5) primary key,
DesArt varchar(30) not null,
PvpArt float(6,2) unsigned not
null);

Use pedidos;
Create table LineaPedido
(RefPed char(5),
CodArt char(5),
CantArt int(4) unsigned not null default 1,
Index (RefPed),
Foreign key (RefPed) references pedido(RefPed) on update cascade,
Index (CodArt),
Foreign key (CodArt) references articulo(CodArt) on update cascade,
Primary key (RefPed, CodArt));

Lenguajes de definición y modificación de datos SQL

Insertar datos

```
INSERT [INTO] nombre_tabla [atributo1,...atributon]  
VALUES ([DEFAULT|valor11],...,[DEFAULT|valor1n]),  
([DEFAULT|valor21],...,[DEFAULT|valor2n]),...  
([DEFAULT|valorn1],...,[DEFAULT|valornn]);
```

```
INSERT INTO PEDIDO VALUES ('P0001','2014-02-16'),  
('P0002','2014-02-18'), ('P0003','2014-02-23'),  
('P0004','2014-02-25');  
INSERT INTO ARTICULO VALUES ('A0043','Bolígrafo  
azul fino',0.78),  
('A0078','Bolígrafo rojo normal',1.05), ('A0075','Lápiz  
2B',0.55),  
('A0012','Goma de borrar',0.15),  
('A0089','Sacapuntas',0.25);  
INSERT INTO LINEAPEDIDO VALUES  
('P0001','A0043',10),  
('P0001','A0078',12), ('P0002','A0043',5),  
('P0003','A0075',20),  
('P0004','A0012',15), ('P0004','A0043',5),  
('P0004','A0089',50);
```

Seleccionar datos (DML)

```
SELECT expression1, expresion2, expression3...  
FROM tabla1, table2, tabla3  
[WHERE criterio de seleccion]  
[GROUP BY expresion1, expresion2,...]  
[HAVING criterio de agrupamiento]  
[ORDER BY expresion1 [asc|desc], expresion2 [asc|desc],...]
```

```
Select A.CodArt, DesArt, count(RefPed),sum(Cantart)  
From LineaPedido L, Articulo A  
Where L.CodArt=A.CodArt  
Group By A.CodArt, DesArt  
Having Count (RefPed)>1;
```

Lenguajes de definición y modificación de datos SQL

Crear vistas

CREATE [OR REPLACE] VIEW

nombre_vista

[campo₁,...,campo₂,...,campo_n],

AS sentencia_select

```
Create view ArticulosPedidos as  
Select A.CodArt, DesArt, count(RefPed), sum(CantArt)  
From LineaPedido L, Articulo A  
Where L.CodArt=A.CodArt  
Group By A.CodArt, DesArt  
Having count(RefPed)> 1;
```

Lenguajes de definición y modificación de datos SQL

Eliminar datos

DELETE FROM nombre_vista
[**WHERE** condicion],
[**ORDER BY** criterio]

```
Delete from articulo where PvpArt > 2;  
Delete from pedido where refped not in  
(select RefPed from lineapedido);  
Delete from pedido;
```

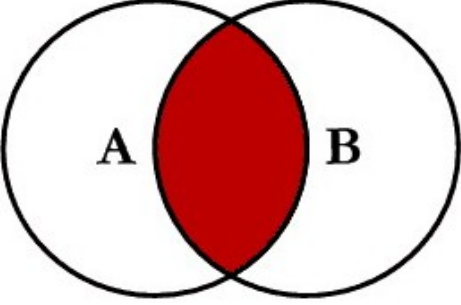
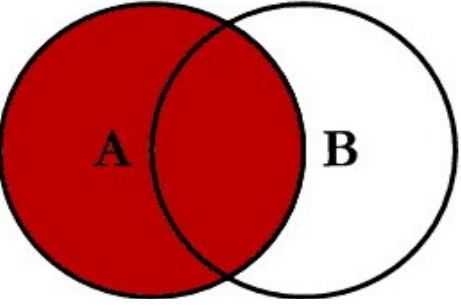
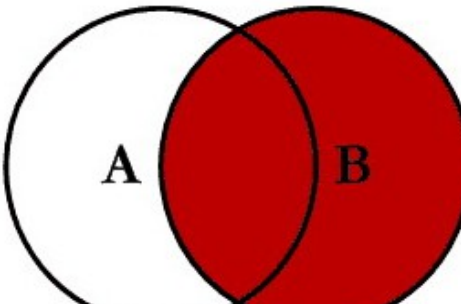
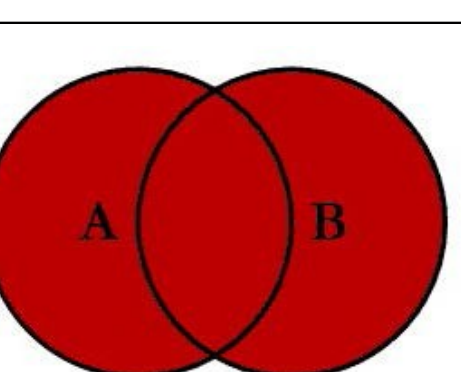
Eliminar tablas

DROP TABLE [IF EXISTS] nombre_tabla₁,...
TRUNCATE TABLE nombre_tabla

```
Drop table articulo, pedido;  
Truncate table lineapedido;
```

Lenguajes de definición y modificación de datos SQL

Agrupación de tablas

	<pre>SELECT <columnas> FROM tablaA A, tablaB B WHERE A.Key=B.Key SELECT <columnas> FROM tablaA A INNER JOIN tablaB B ON A,Key=B,Key</pre>
	<pre>SELECT <columnas> FROM tablaA A LEFT JOIN tablaB B ON A,Key=B,Key</pre>
	<pre>SELECT <columnas> FROM tablaA A RIGHT JOIN tablaB B ON A,Key=B,Key</pre>
	<pre>SELECT <columnas> FROM tablaA UNION [ALL] SELECT <columnas> FROM tablaB</pre>

Lenguajes de definición y modificación de datos SQL

ALTER TABLE (MODIFICAR TABLA)

Una vez que se crea la tabla en la base de datos, hay muchas ocasiones donde uno puede desear cambiar la estructura de la tabla. Los casos típicos incluyen los siguientes:

- Agregar una columna
- Eliminar una columna
- Cambiar el nombre de una columna
- Cambiar el tipo de datos para una columna
-

La sintaxis SQL para **ALTER TABLE** es

ALTER TABLE "nombre_tabla"
[modificar especificación];

[modificar especificación] depende del tipo de modificación que deseamos realizar. Para los usos mencionados anteriormente, las instrucciones [modificar especificación] son:

- Agregar una columna:
 - **ADD** "columna 1" "tipos de datos para columna 1"
- Eliminar una columna:
 - **DROP** "columna 1"
- Cambiar el nombre de una columna:
 - **CHANGE** "nombre antiguo de la columna" "nuevo nombre de la columna" "tipos de datos para la nueva columna".
- Cambiar el tipo de datos para una columna:
 - **MODIFY** "columna 1" "nuevo tipo de datos"

ALTER TABLE [nombre tabla] MODIFY [nombre_campo][tipo_datos];

Lenguajes de definición y modificación de datos SQL

Funciones de agregación	
AVG	Media de los valores del campo Select avg(precio) from articulos
COUNT	Devuelve el numero de tuplas existentes en la tabla. Select count(*) from empleados
SUM	Suma de los valores del campo select sum(importe) from ventas
MIN	Valor mínimo existente en la tabla. Select min(salario) from empleados
MAX	Valor máximo existente en la tabla. Select max(edad) from empleados

Funciones de control de flujo	
IF(expr1,expr2,expr3)	Si expr1 es true, entonces devuelve expr2, si no devuelve expr3 SELECT IF(1>2,2,3); -> 3 SELECT IF(1<2,'yes','no'); -> 'yes' SELECT IF(STRCMP('test','test1'),'no','yes'); -> 'no'
CASE value WHEN [compare value] THEN result	SELECT CASE 1 WHEN 1 THEN 'one' WHEN 2 THEN 'two' ELSE 'more' END; -> 1
IFNULL (expr1,expr2)	Devuelve expr1 si no es nulo, si lo es devuelve expr2 SELECT IFNULL(1,0); -> 1 SELECT IFNULL(NULL,10); -> 10 SELECT IFNULL(1/0,10); -> 10
NULLIF (expr1,expr2)	Devuelve null si expr1 es igual a expr2, si no devuelve expr1 SELECT NULLIF(1,1); -> NULL SELECT NULLIF(1,2); -> 1

Lenguajes de definición y modificación de datos SQL

Tipo Texto (Char(x), Varchar(x), Text, TinyText, MediumText, LongText)	
Char(x)	Tipo de datos que admite caracteres alfanuméricos. La longitud de este campo varía entre 1-255 y está delimitado a la longitud especificada entre paréntesis (x) en el momento de la creación del campo de la tabla. Al introducir datos en este campo siempre se solicitará el número de caracteres especificados. Si creamos un campo con Char(5) deberemos introducir cinco caracteres cada vez que incluyamos un dato en ese campo. Si incluimos menos, MySQL rellenará los caracteres que faltan hasta el número indicado con espacios.
Varchar(x)	Tipo de datos que admite caracteres alfanuméricos. Su uso es similar a Char(x). A la hora de definir un campo de datos Varchar deberemos especificar el número máximo de caracteres que podrá aceptar en la entrada de datos, donde x es un número entre 1-255. A diferencia de Char, este tipo de datos es variable en su longitud, admitiendo entradas inferiores a la establecida.
Text, TinyText, MediumText, LongText	Mediante la declaración de este tipo de datos se admiten la inclusión de cadenas alfanuméricas "case-insensitive" de longitudes variables. TinyText admite un máximo de 255 caracteres, Text admite 65.535, MediumText permite introducir textos de hasta 16.777.215 caracteres, LongText nos ofrece la posibilidad de incluir un máximo de 4.294.967.295 caracteres. Estos campos no necesitan de especificaciones de longitud a la hora de ser declarados.

Tipo Binario (Blob, TinyBlob, MediumBlob, LongBlob)	
Blob	Un tipo de datos Blob es un objeto binario que puede almacenar cualquier tipo de datos o información, desde un archivo de texto con todo su formato (se diferencia en esto de el tipo Text) hasta imágenes, archivos de sonido o video, etc. Al igual que el tipo Text, Blob admite hasta 65.535 caracteres.
TinyBlob, MediumBlob, LongBlob	Son datos del mismo tipo que el anterior pero que varían en cuanto a su tamaño, así TinyBlob admite hasta 255 caracteres máximo, MediumBlob acepta tamaños de hasta 16.777.215 de caracteres y LongBlob 4.294.967.295 caracteres (como vemos estos tamaños se corresponden con los de TinyText, MediumText y LongText).

Lenguajes de definición y modificación de datos SQL

Tipo numérico (TinyInt, SmallInt, MediumInt, Int, BigInt, Float, Double, Decimal)	
int	Este es un tipo de datos numéricos de tipo entero. Este tipo de datos guarda valores enteros (no decimales) entre -2.147.483.648 y 2.147.483.647.
Tinyint, Smallint, Mediumint, Bigint	Son tipos de datos numéricos enteros (no decimal). Tinyint agrupa un rango de números entre -128 y 127. Smallint alcanza desde -32.768 hasta 32.767. Mediumint tiene un rango comprendido entre -8.388.608 y 8.388.607. Finalmente el tipo de datos Bigint ocupa un rango numérico entre -9.223.372.036.854.775.808 hasta 9.223.372.036.854.775.807.
Float (M,D)	Número de coma flotante de precisión simple. El valor del argumento M nos indica el número de dígitos decimales que se van a utilizar para representar el número. Así, un valor de 5 nos permitirá representar números comprendidos entre -99 y 99 (Números expresados en binario con 5 dígitos y signo). El valor del argumento D nos indica el número de posiciones decimales que se van a utilizar en la representación del número. Así, una representación tipo Float (5,2) nos permitirá incluir números entre -99,99 y 99,99. El rango de los números de coma flotante de precisión simple es de -3.402823466E+38 a -1,175494351E-38, 0, y 1.175494351E-38 hasta 3,402823466E+38.
Double (M,D)	Número de coma flotante de precisión doble. Es un tipo de datos igual al anterior cuya única diferencia es el rango numérico que abarca, siendo este el comprendido entre 1,976931348623157E+308 hasta -2,2250738585072014E-308, 0, y 2,2250738585072014E-308 to 1,976931348623157E+308
Decimal (M,D)	Su uso es similar al de los anteriores, pero, en este caso, D puede tener valor 0. El rango de este número es el mismo que el de número con coma flotante de precisión doble.

Lenguajes de definición y modificación de datos SQL

Tipo Fecha-Hora (Date, DateTime, Timestamp, Time, Year)	
Date	Formato de Fecha. Su representación es en formato de fecha numérica del tipo 'YYYY-MM-DD' (Año con cuatro dígitos. Mes con dos dígitos, día con dos dígitos). Su rango es '1000-01-01' (1 de enero del año 1000. en el cual yo era aún muy pequeño) hasta '9999-12-31' (31 de diciembre del 9999, que ya veremos que pasa después de las uvas)
DateTime	Es una combinación de formato de fecha y hora conjuntamente. Su representación es 'YYYY-MM-DD HH:MM:SS' (Año con cuatro dígitos, Mes con dos dígitos, día con dos dígitos, hora con dos dígitos, minutos con dos dígitos, segundos con dos dígitos). El rango que soporta este formato es de '1000-01-01 00:00:00' (las 00 horas, 00 minutos, 00 segundos del 1 de enero del año 1000, que no se yo con que reloj podían medir esto) hasta '9999-12-31 23:59:59' (las 23 horas, 59 minutos. 59 segundos del 31 de diciembre del año 9999. es decir, justo antes de las campanadas y una vez que han acabado los cuartos).
TimeStamp(N)	Este es un tipo de datos muy particular. Necesita de un argumento N que puede ser uno de estos números; 14,12,10, 8, 6, 4, 2. N representa el número de dígitos que se utilizarán para representar un valor de fecha y hora comprendido desde el inicio del año 1970 hasta algún momento del año 2037. Así: Timestamp(14): YYYYMMDDHHMMSS (Año 4 dígitos - mes - día - hora - minutos - segundos 2 dígitos) Timestamp(12): YYMMDDHHMMSS (Año 2 dígitos - mes - día - hora - minutos - segundos 2 dígitos) Timestamp(10): YYMMDDHHMM (Año - mes - día - hora- minutos 2 dígitos) Timestamp(8): YYMMDDHH (Año - mes -día-hora 2 dígitos) Timestamp(6): YYMMDD (Año - mes - día 2 dígitos) Timestamp(4): YYMM (Año -mes 2 dígitos) Timestamp(2): YY (Año 2 dígitos)
Time	Tipo de datos con formato de Hora. mySQL muestra valores de hora con formato 'HH:MM:SS'
Year(D)	Tipo de datos con formato de año. Su representación puede ser 'YYYY' (año con formato de 4 dígitos) o 'YY' (año con formato de 2 dígitos) donde el valor del argumento D puede ser 4 o 2 respectivamente.

Lenguajes de definición y modificación de datos SQL

Tipos de dato en una base de datos MySQL

Tipos de dato numéricos

INT (INTEGER): Ocupación de 4 bytes con valores entre -2147483648 y 2147483647 o entre 0 y 4294967295.

SMALLINT: Ocupación de 2 bytes con valores entre -32768 y 32767 o entre 0 y 65535.

TINYINT: Ocupación de 1 bytes con valores entre -128 y 127 o entre 0 y 255.

MEDIUMINT: Ocupación de 3 bytes con valores entre -8388608 y 8388607 o entre 0 y 16777215.

BIGINT: Ocupación de 8 bytes con valores entre -8388608 y 8388607 o entre 0 y 16777215.

DECIMAL (NUMERIC): Almacena los números de coma flotante como cadenas o string.

FLOAT (m,d): Almacena números de **coma flotante**, donde 'm' es el número de dígitos de la parte entera y 'd' el número de decimales.

DOUBLE (REAL): Almacena número de coma flotante con precisión doble. Igual que FLOAT, la diferencia es el rango de valores posibles.

BIT (BOOL, BOOLEAN): Número entero con valor 0 o 1.

Tipos de dato con formato fecha

DATE: Válido para almacenar una fecha con año, mes y día, su rango oscila entre '1000-01-01' y '9999-12-31'.

DATETIME: Almacena una fecha (año-mes-día) y una hora (horas-minutos-segundos), su rango oscila entre '1000-01-01 00:00:00' y '9999-12-31 23:59:59'.

TIME: Válido para almacenar una hora (horas-minutos-segundos). Su rango de horas oscila entre -838-59-59 y 838-59-59. El formato almacenado es 'HH:MM:SS'.

TIMESTAMP: Almacena una fecha y hora UTC. El rango de valores oscila entre '1970-01-01 00:00:01' y '2038-01-19 03:14:07'.

YEAR: Almacena un año dado con 2 o 4 dígitos de longitud, por defecto son 4. El rango de valores oscila entre 1901 y 2155 con 4 dígitos. Mientras que con 2 dígitos el rango es desde 1970 a 2069 (70-69).

Lenguajes de definición y modificación de datos SQL

Tipos de dato con formato string

CHAR: Ocupación fija cuya longitud comprende de 1 a 255 caracteres.

VARCHAR: Ocupación variable cuya longitud comprende de 1 a 255 caracteres.

TINYBLOB: Una longitud máxima de 255 caracteres. Válido para objetos binarios como son un fichero de texto, imágenes, ficheros de audio o vídeo. No distingue entre minúsculas y mayúsculas.

BLOB: Una longitud máxima de 65.535 caracteres. Válido para objetos binarios como son un fichero de texto, imágenes, ficheros de audio o vídeo. No distingue entre minúsculas y mayúsculas.

MEDIUMBLOB: Una longitud máxima de 16.777.215 caracteres. Válido para objetos binarios como son un fichero de texto, imágenes, ficheros de audio o vídeo. No distingue entre minúsculas y mayúsculas.

LOB: Una longitud máxima de 4.294.967.298 caracteres. Válido para objetos binarios como son un fichero de texto, imágenes, ficheros de audio o vídeo. No distingue entre minúsculas y mayúsculas.

SET: Almacena 0, uno o varios valores una lista con un máximo de 64 posibles valores.

ENUM: Igual que **SET** pero solo puede almacenar un valor.

TINYTEXT: Una longitud máxima de 255 caracteres. Sirve para almacenar texto plano sin formato. Distingue entre minúsculas y mayúsculas.

TEXT: Una longitud máxima de 65.535 caracteres. Sirve para almacenar texto plano sin formato. Distingue entre minúsculas y mayúsculas.

MEDIUMTEXT: Una longitud máxima de 16.777.215 caracteres. Sirve para almacenar texto plano sin formato. Distingue entre minúsculas y mayúsculas.

LONGTEXT: Una longitud máxima de 4.294.967.298 caracteres. Sirve para almacenar texto plano sin formato. Distingue entre minúsculas y mayúsculas.