**Practica 8 MYSQL**

**1.-** Crea la base de datos **P8** y en ella la tabla **Usuarios**:

CREATE TABLE usuarios(

id Smallint (5) unsigned not NULL AUTO\_INCREMENT,

Nombre varchar (30) NOT NULL,

curso Smallint (5) UNSIGNED DEFAULT NULL,

PRIMARY KEY (id)

) ENGINE = InnoDB;

Introduce los datos:

|  |  |  |
| --- | --- | --- |
| **id** | **nombre** | **curso** |
| 1 | Paula | 1 |
| 2 | Silvia | 1 |
| 3 | Carlos | 2 |
| 4 | Ruth | 5 |
| 5 | José Antonio | (NULL) |

**2.-** Crea la tabla **Cursos**:

CREATE TABLE cursos (

id smallint(5) unsigned NOT NULL AUTO\_INCREMENT,

nombre varchar(50) NOT NULL,

PRIMARY KEY (id)

) ENGINE=InnoDB;

Introduce los datos:

|  |  |
| --- | --- |
| **id** | **nombre** |
| 1 | HTML5 |
| 2 | CSS3 |
| 3 | JavaScript |
| 4 | PHP |
| 5 | MySQL |

**3.-** Añadimos una clave ajena para relacionar los usuarios con los curos que realizan:

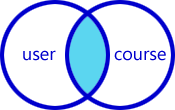
ALTER TABLE usuarios

ADD FOREIGN KEY (curso) REFERENCES cursos (id)

ON UPDATE CASCADE;

Comprueba que todo está correcto con **show create table**.

INNER JOIN

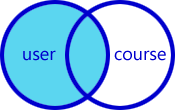
La cláusula se utiliza con más frecuencia es INNER JOIN. Esto produce un conjunto de registros que coinciden tanto en las tablas de usuario y cursos, es decir, todos los usuarios que están inscritos en un curso:

**4.-** Prueba y escribe el resultado:

SELECT usuarios.nombre, cursos.nombre

FROM usuarios INNER JOIN cursos on usuarios.curso = cursos.id;

LEFT JOIN

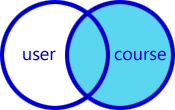
¿Qué pasa si se requiere una lista de todos los estudiantes y sus cursos, incluso si no están inscritos en uno? A LEFT JOIN produce un conjunto de registros que coincide con cada entrada en la tabla de la izquierda (por el usuario), independientemente de cualquier entrada coincidente en la tabla a la derecha.

**5.-** Prueba y escribe el resultado:

SELECT usuarios.nombre, cursos.nombre

FROM usuarios LEFT JOIN cursos on usuarios.curso = cursos.id;

RIGHT JOIN

Tal vez necesitamos una lista de todos los cursos y los estudiantes, incluso si nadie se ha inscrito? A RIGHT JOIN produce un conjunto de registros que coincide con cada entrada en la tabla de la derecha (por supuesto), independientemente de cualquier entrada coincidente en la tabla de la izquierda.

**6.-** Prueba y escribe el resultado:

SELECT usuarios.nombre, cursos.nombre

FROM usuarios RIGHT JOIN cursos on usuarios.curso = cursos.id;

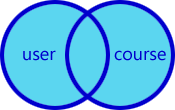
RIGHT JOIN rara vez se utilizan ya que se puede expresar el mismo resultado utilizando un LEFT JOIN. Esto puede ser más eficiente y más rápido para la base de datos para analizar:

SELECT usuarios.nombre, cursos.nombre

FROM cursos LEFT JOIN usuarios on on usuarios.curso = cursos.id;

**7.-** Contar el número de alumnos matriculados en cada curso, usando LEFT JOIN.:

OUTER JOIN (o FULL OUTER JOIN)

Nuestra última opción es la OUTER JOIN que devuelve todos los registros en ambas tablas, independientemente de cualquier partido. Cuando no exista una coincidencia, el lado que falta contendrá NULL.

OUTER JOIN es menos útil que INNER, LEFT o RIGHT y no se implementó en MySQL. Sin embargo, puede evitar esta limitación mediante la unión de un LEFT y RIGHT JOIN, por ejemplo,

SELECT usuarios.nombre, cursos.nombre

FROM usuarios

LEFT JOIN cursos on usuarios.curso = cursos.id;

UNION

SELECT usuarios.nombre, cursos.nombre

FROM usuarios

RIGHT JOIN cursos on usuarios.curso = cursos.id;

**8.-** Haz una copia de seguridad de la base de datos **P8**. Luego Borra la base de datos y restaura la copia efectuada.

mysqldump –u root -p p8 > copia\_p8.sql