

MF0226_3 Programación de bases de datos relacionales

ENUNCIADO

Nuestra empresa lpartek dispone de una base datos ya diseñada y funcionando en explotación.

La base de datos contiene información sobre las formaciones que se imparten en lpartek, así como sus clientes, alumnos y formadores.

Necesitamos realizar una pasarela para que cualquier cliente (Java, PHP, JavaScript) puede consultar cierta información que nos interesa mostrar de forma pública. Para lo cual debemos de crear un medio de conexión entre la base datos y estos clientes usando un Web Service Rest.

Además tenemos que diseñar e implementar una nueva tabla para que los usuarios pueden escribir reseñas de los cursos.

La información que necesitamos retornar es la siguiente.

Todos los cursos realizados que muestre los siguientes campos:

- curso nombre
- curso identificador
- curso numero horas
- profesor del curso (nombre, apellido)

Detalle de un curso, junto con las reseñas de los usuarios

UF2175

Diagrama Entidad Relación de la BD actual

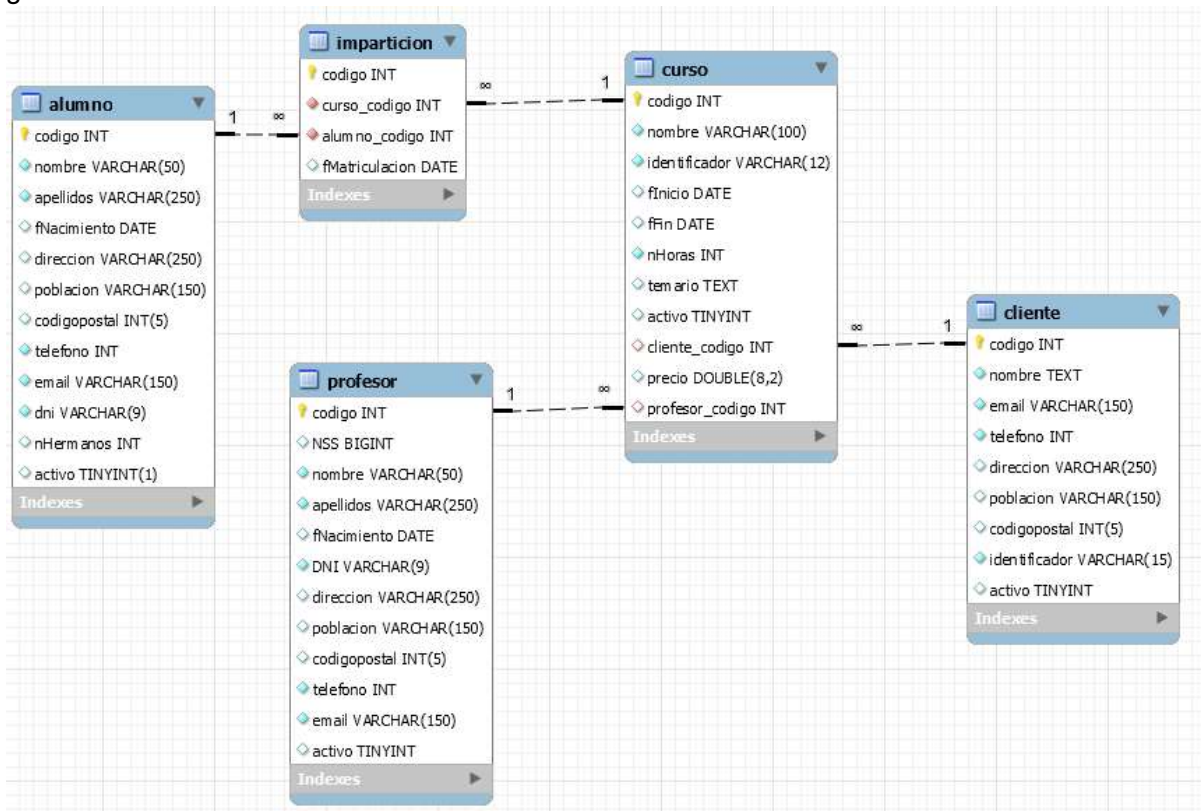
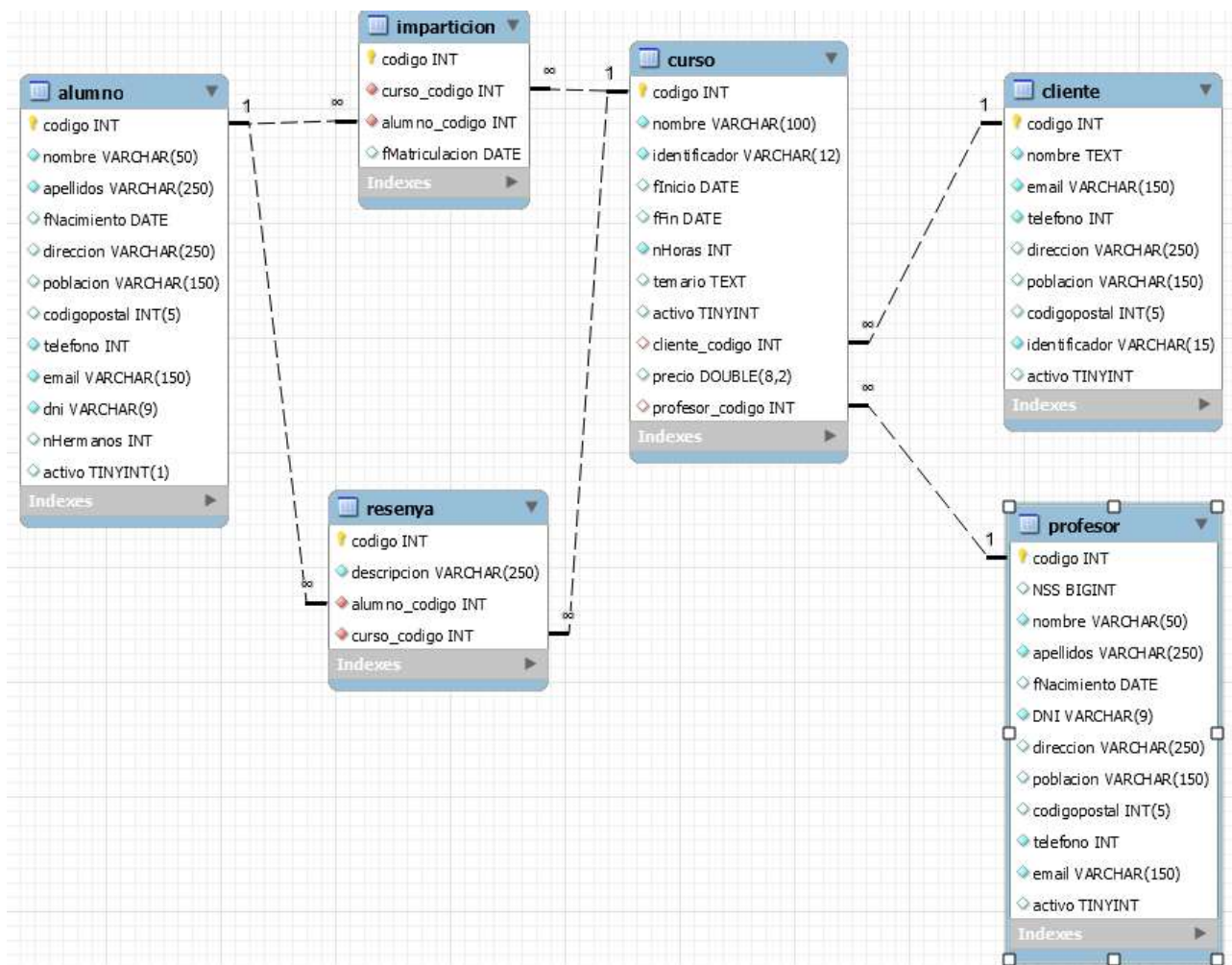


Diagrama Entidad Relación de la nueva tabla



Script de backup con la base de datos actual

```

-- phpMyAdmin SQL Dump
-- version 4.5.2
-- http://www.phpmyadmin.net
--
-- Servidor: 127.0.0.1
-- Tiempo de generación: 18-05-2017 a las 09:44:41
-- Versión del servidor: 5.7.9
-- Versión de PHP: 5.6.16
  
```

```

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET time_zone = "+00:00";
  
```

```

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;
  
```

```

-- Base de datos: `gestiondocente`
  
```

```

DROP DATABASE IF EXISTS `gestiondocente`;
CREATE DATABASE IF NOT EXISTS `gestiondocente` DEFAULT CHARACTER SET utf8 COLLATE utf8_bin;
USE `gestiondocente`;
  
```

```

DELIMITER $$
  
```

```

-- Procedimientos
  
```

```

DROP PROCEDURE IF EXISTS `alumnoCreate`$$
CREATE DEFINER='admin'@'%' PROCEDURE `alumnoCreate` (IN `papellidos` VARCHAR(250), IN `pcodigopostal` INT(5), IN `pdireccion` VARCHAR(250), IN `pdni` VARCHAR(9), IN `pemail` VARCHAR(150), IN `pfNacimiento` DATE, IN `pnHermanos` INT(2), IN `pnombre` VARCHAR(50), IN `ppoblacion` VARCHAR(150), IN `ptelefono` INT(9), OUT `pcodigo` INT) BEGIN
  
```

```

INSERT INTO alumno(nombre,apellidos,dni,email,direccion,codigopostal,poblacion,fNacimiento,telefono,nHermanos)
VALUES(LOWER(pnombre),LOWER(papellidos),LOWER(pdni),LOWER(pemail),LOWER(pdireccion),pcodigopostal,LOWER(ppoblacion),pfNacimiento,ptel
efono,pnHermanos);
SET pcodigo = LAST_INSERT_ID();

END$$

DROP PROCEDURE IF EXISTS `alumnoDelete`$$
CREATE DEFINER='admin'@'%' PROCEDURE `alumnoDelete` (IN `pcodigo` INT) NO SQL
BEGIN

DELETE FROM alumno WHERE codigo = pcodigo;

END$$

DROP PROCEDURE IF EXISTS `alumnogetAll`$$
CREATE DEFINER='admin'@'%' PROCEDURE `alumnogetAll` () NO SQL
BEGIN

SELECT `codigo` as alumnocodigo, `nombre` as alumnonombre, `apellidos` as alumnoapellidos,
`fNacimiento` as alumnofnacimiento, `direccion` as alumnodireccion, `poblacion` as alumnopoblacion, `codigopostal` as alumnocodigopostal, `telefono` as
alumnotelefono, `email` as alumnoemail, `dni` as alumnodni, `nHermanos` as alumnonhermanos, `activo` as alumnoactivo
FROM `alumno`

WHERE alumno.activo = true;
END$$

DROP PROCEDURE IF EXISTS `alumnogetByCurso`$$
CREATE DEFINER='root'@'localhost' PROCEDURE `alumnogetByCurso` (IN `pcursocodigo` INT) BEGIN
SELECT a.`codigo` as alumnocodigo, a.`nombre` as alumnonombre, a.`apellidos` as alumnoapellidos,
a.`fNacimiento` as alumnofnacimiento, a.`direccion` as alumnodireccion, a.`poblacion` as alumnopoblacion, a.`codigopostal` as alumnocodigopostal,
a.`telefono` as alumnotelefono, a.`email` as alumnoemail, a.`dni` as alumnodni, a.`nHermanos` as alumnonhermanos, a.`activo` as alumnoactivo
FROM `alumno` as a
inner join imparticion as i ON i.alumno_codigo = a.codigo
inner join curso as c ON i.curso_codigo = c.codigo
WHERE c.codigo = pcursocodigo
group by a.codigo;

END$$

DROP PROCEDURE IF EXISTS `alumnogetByDni`$$
CREATE DEFINER='root'@'localhost' PROCEDURE `alumnogetByDni` (IN `pdni` VARCHAR(9)) BEGIN
SELECT `codigo` as alumnocodigo, `nombre` as alumnonombre, `apellidos` as alumnoapellidos,
`fNacimiento` as alumnofnacimiento, `direccion` as alumnodireccion, `poblacion` as alumnopoblacion, `codigopostal` as alumnocodigopostal, `telefono` as
alumnotelefono, `email` as alumnoemail, `dni` as alumnodni, `nHermanos` as alumnonhermanos, `activo` as alumnoactivo
FROM `alumno`
WHERE dni = pdni;

END$$

DROP PROCEDURE IF EXISTS `alumnogetById`$$
CREATE DEFINER='admin'@'%' PROCEDURE `alumnogetById` (IN `pcodigo` INT) NO SQL
BEGIN

SELECT `codigo` as alumnocodigo, `nombre` as alumnonombre, `apellidos` as alumnoapellidos,
`fNacimiento` as alumnofnacimiento, `direccion` as alumnodireccion, `poblacion` as alumnopoblacion, `codigopostal` as alumnocodigopostal, `telefono` as
alumnotelefono, `email` as alumnoemail, `dni` as alumnodni, `nHermanos` as alumnonhermanos, `activo` as alumnoactivo
FROM `alumno`
WHERE codigo = pcodigo;

END$$

DROP PROCEDURE IF EXISTS `alumnoInforme`$$
CREATE DEFINER='root'@'localhost' PROCEDURE `alumnoInforme` (IN `pcodigo` INT) BEGIN
SELECT a.`codigo` as alumnocodigo, a.`nombre` as alumnonombre, a.`apellidos` as alumnoapellidos,
a.`fNacimiento` as alumnofnacimiento, a.`direccion` as alumnodireccion, a.`poblacion` as alumnopoblacion, a.`codigopostal` as alumnocodigopostal,
a.`telefono` as alumnotelefono, a.`email` as alumnoemail, a.`dni` as alumnodni, `nHermanos` as alumnonhermanos, a.`activo` as alumnoactivo,
c.codigo as cursocodigo, c.nombre as curso nombre,
c.identificador as cursoidentificador, c.flnicio as cursofinicio, c.ffin as cursoffin, c.nHoras as cursohoras, c.precio as cursoprecio
/*,SUM(cd.precio) as preciocurso*/
FROM alumno as a
LEFT JOIN imparticion as i ON i.alumno_codigo = a.codigo
LEFT JOIN curso as c ON c.codigo = i.curso_codigo
WHERE a.codigo = pcodigo;
END$$

DROP PROCEDURE IF EXISTS `alumnoUpdate`$$
CREATE DEFINER='admin'@'%' PROCEDURE `alumnoUpdate` (IN `papellidos` VARCHAR(250), IN `pcodigo` INT, IN `pcodigopostal` INT(5), IN
`pdireccion` VARCHAR(250), IN `pdni` VARCHAR(9), IN `pemail` VARCHAR(150), IN `pfNacimiento` DATE, IN `pnHermanos` INT(2), IN `pnombre`
VARCHAR(150), IN `ppoblacion` VARCHAR(150), IN `ptelefono` INT(9)) NO SQL
BEGIN

UPDATE alumno
SET nombre = LOWER(pnombre),apellidos = LOWER(papellidos), dni = LOWER(pdni),email =
LOWER(pemail),direccion=LOWER(pdireccion),codigopostal=pcodigopostal,poblacion=LOWER(ppoblacion),fNacimiento=pfNacimiento,telefono=ptelefono,
nHermanos=pnHermanos
WHERE codigo = pcodigo;

END$$

DROP PROCEDURE IF EXISTS `clienteCreate`$$

```

```

CREATE DEFINER='root'@'localhost' PROCEDURE `clienteCreate` (IN `pnombre` TEXT, IN `pemail` VARCHAR(150), IN `ptelefono` INT, IN `pdireccion` VARCHAR(150), IN `ppoblacion` VARCHAR(150), IN `pcodigopostal` INT(5), IN `pidentificador` VARCHAR(50), OUT `pcodigo` INT) BEGIN
INSERT INTO cliente(`nombre`, `email`, `telefono`, `direccion`, `poblacion`, `codigopostal`, `identificador`)
VALUES(LOWER(pnombre),
      LOWER(pemail),
      ptelefono,
      LOWER(pdireccion),
      LOWER(ppoblacion),
      codigopostal,
      LOWER(pidentificador));

      SET pcodigo = LAST_INSERT_ID();

END$$

DROP PROCEDURE IF EXISTS `clientegetAll` $$
CREATE DEFINER='admin'@'%' PROCEDURE `clientegetAll` () NO SQL
BEGIN
      SELECT `codigo` as clientecodigo, `nombre` as clientenombre, `email` as clienteemail, `telefono` as
      clientetelefono, `direccion` as clientedireccion, `poblacion` as clientepoblacion, `codigopostal` as clientecodigopostal, `identificador` as
      clienteidentificador, activo as clienteactivo
      FROM `cliente`
      WHERE cliente.activo = true;
END$$

DROP PROCEDURE IF EXISTS `clientegetByld` $$
CREATE DEFINER='root'@'localhost' PROCEDURE `clientegetByld` (IN `pcodigo` INT) BEGIN
      SELECT `codigo` as clientecodigo, `nombre` as clientenombre, `email` as clienteemail, `telefono` as
      clientetelefono, `direccion` as clientedireccion, `poblacion` as clientepoblacion, `codigopostal` as clientecodigopostal, `identificador` as
      clienteidentificador, activo as clienteactivo
      FROM `cliente`
      WHERE codigo = pcodigo;
END$$

DROP PROCEDURE IF EXISTS `clientegetByIdentificador` $$
CREATE DEFINER='root'@'localhost' PROCEDURE `clientegetByIdentificador` (IN `pidentificador` VARCHAR(15)) BEGIN
      SELECT `codigo` as clientecodigo, `nombre` as clientenombre, `email` as clienteemail, `telefono` as
      clientetelefono, `direccion` as clientedireccion, `poblacion` as clientepoblacion, `codigopostal` as clientecodigopostal, `identificador` as
      clienteidentificador, activo as clienteactivo
      FROM `cliente`
      WHERE identificador = pidentificador;
END$$

DROP PROCEDURE IF EXISTS `clienteInforme` $$
CREATE DEFINER='root'@'localhost' PROCEDURE `clienteInforme` (IN `pcodigo` INT) BEGIN
SELECT c.`codigo` as clientecodigo, c.`nombre` as clientenombre, c.`email` as clienteemail, c.`telefono` as clientetelefono, c.`direccion` as
clientedireccion, c.`poblacion` as clientepoblacion, `codigopostal` as clientecodigopostal, c.`identificador` as clienteidentificador, c.activo as clienteactivo,
cu.codigo as cursocodigo, cu.nombre as cursonombre, cu.identificador as cursoidentificador, cu.fInicio as cursofinicio, cu.fFin as cursoffin, cu.nhoras as
cursonhoras, cu.precio as cursoprecio FROM cliente as c LEFT JOIN curso as cu ON cu.cliente_codigo = c.codigo WHERE c.codigo = pcodigo;
END$$

DROP PROCEDURE IF EXISTS `clienteUpdate` $$
CREATE DEFINER='root'@'localhost' PROCEDURE `clienteUpdate` (IN `pcodigo` INT, IN `pnombre` VARCHAR(150), IN `pemail` VARCHAR(150), IN
`ptelefono` INT(9), IN `pdireccion` VARCHAR(250), IN `ppoblacion` VARCHAR(150), IN `pcodigopostal` INT(5), IN `pidentificador` VARCHAR(15)) BEGIN
UPDATE `cliente`
SET `codigo`=pcodigo,
      `nombre`=pnombre,
      `email`=pemail,
      `telefono`=ptelefono,
      `direccion`=pdireccion,
      `poblacion`=ppoblacion,
      `codigopostal`=pcodigopostal,
      `identificador`=pidentificador;
END$$

DROP PROCEDURE IF EXISTS `cursocreate` $$
CREATE DEFINER='admin'@'%' PROCEDURE `cursocreate` (IN `pnombre` VARCHAR(50), IN `pidentificador` VARCHAR(50), IN `pfinicio` DATE, IN
`pffin` DATE, IN `ptemario` VARCHAR(50), IN `pprecio` DOUBLE(8,2), IN `pcodigo_profesor` INT, IN `pcodigo_cliente` INT, OUT `pcodigo` INT) NO SQL
BEGIN

INSERT INTO curso(nombre,identificador,fInicio,fFin,temario,precio,cliente_codigo,profesor_codigo)
VALUES(LOWER(pnombre),LOWER(pidentificador),pfinicio,pffin,LOWER(ptemario),pprecio,pcodigo_profesor,pcodigo_cliente);
SET pcodigo = LAST_INSERT_ID();

END$$

DROP PROCEDURE IF EXISTS `cursoDelete` $$
CREATE DEFINER='admin'@'%' PROCEDURE `cursoDelete` (IN `pcodigo` INT) NO SQL
BEGIN

UPDATE curso SET activo = false
WHERE codigo = pcodigo;

END$$

DROP PROCEDURE IF EXISTS `cursogetAll` $$
CREATE DEFINER='admin'@'%' PROCEDURE `cursogetAll` () NO SQL
BEGIN
SELECT
      c.`codigo` as cursocodigo, c.`nombre` as cursonombre, c.`identificador` as cursoidentificador,
      c.`finicio` as cursofinicio, c.`fFin` as cursoffin, c.`nHoras` as cursonhoras, c.`temario` as cursotemario, c.`activo` as cursoactivo, c.`precio` as cursoprecio,
      p.`codigo` as profesorcodigo, p.`NSS` as profesornss, p.`nombre` as profesornombre, p.`apellidos` as profesorapellidos, p.`fNacimiento` as

```

```

profesorfnacimiento, p.`DNI` as profesordni, p.`direccion` as profesordireccion, p.`poblacion` as profesorpoblacion, p.`codigopostal` as
profesorcodigopostal, p.`telefono` as profesortelefono, p.`email` as profesoremail, p.activo as profesoractivo, cli.codigo as clientecodigo, cli.`nombre` as
clientenombre, cli.`email` as clienteemail, cli.`telefono` as clientetelefono, cli.identificador as clienteidentificador, cli.`direccion` as clientedireccion,
cli.`poblacion` as clientepoblacion, cli.`codigopostal` as clientecodigopostal, cli.activo as clienteactivo
FROM curso c

```

```

        LEFT JOIN profesor p ON p.codigo = c.profesor_codigo
    LEFT JOIN cliente cli ON cli.codigo = c.cliente_codigo
WHERE c.activo = true;

```

END\$\$

DROP PROCEDURE IF EXISTS `cursogetById`\$\$

CREATE DEFINER=`admin`@`%` PROCEDURE `cursogetById` (IN `pcodigo` INT) NO SQL

BEGIN

SELECT

```

    c.`codigo` as cursocodigo, c.`nombre` as cursonombre, c.`identificador` as cursoidentificador, c.`fInicio` as cursofinicio, c.`fFin` as cursoffin, c.`nHoras` as
    cursonhoras, c.`temario` as cursotemario, c.`activo` as cursoactivo, c.`precio` as cursoprecio,
    p.`codigo` as profesorcodigo, p.`NSS` as profesornss, p.`nombre` as profesornombre, p.`apellidos` as profesorapellidos, p.`fNacimiento` as
    profesornacimiento, p.`DNI` as profesordni, p.`direccion` as profesordireccion, p.`poblacion` as profesorpoblacion, p.`codigopostal` as
    profesorcodigopostal, p.`telefono` as profesortelefono, p.`email` as profesoremail, p.activo as profesoractivo, cli.codigo as clientecodigo, cli.`nombre` as
    clientenombre, cli.`email` as clienteemail, cli.`telefono` as clientetelefono, cli.identificador as clienteidentificador, cli.`direccion` as clientedireccion,
    cli.`poblacion` as clientepoblacion, cli.`codigopostal` as clientecodigopostal, cli.activo as clienteactivo, i.codigo as imparticioncodigo,
    i.fMatriculacion as imparticionfmatriculacion,
    a.`codigo` as alumnocodigo, a.`nombre` as alunnonombre, a.`apellidos` as alumnoapellidos, a.`fNacimiento` as alumnofnacimiento, a.`direccion` as
    alumnodireccion, a.`poblacion` as alumnopoblacion, a.`codigopostal` as alumnocodigopostal, a.`telefono` as alumnotelefono, a.`email` as alumnoemail,
    a.`dni` as alumnodni, a.`nHermanos` as alumnnonhermanos, a.`activo` as alumnoactivo

```

```

        FROM curso c
    INNER JOIN cliente cli ON cli.codigo = c.cliente_codigo
    INNER JOIN profesor p ON p.codigo = c.profesor_codigo
    INNER JOIN imparticion i on i.curso_codigo = c.codigo
    INNER JOIN alumno a ON a.codigo = i.alumno_codigo
WHERE c.codigo = pcodigo;

```

END\$\$

DROP PROCEDURE IF EXISTS `cursogetByIdentificador`\$\$

CREATE DEFINER=`admin`@`%` PROCEDURE `cursogetByIdentificador` (IN `pidentificador` VARCHAR(50)) NO SQL

BEGIN

SELECT

```

    c.`codigo` as cursocodigo, c.`nombre` as cursonombre, c.`identificador` as cursoidentificador, c.`fInicio` as cursofinicio, c.`fFin` as cursoffin, c.`nHoras` as
    cursonhoras, c.`temario` as cursotemario, c.`activo` as cursoactivo, c.`precio` as cursoprecio,
    p.`codigo` as profesorcodigo, p.`NSS` as profesornss, p.`nombre` as profesornombre, p.`apellidos` as profesorapellidos, p.`fNacimiento` as
    profesornacimiento, p.`DNI` as profesordni, p.`direccion` as profesordireccion, p.`poblacion` as profesorpoblacion, p.`codigopostal` as
    profesorcodigopostal, p.`telefono` as profesortelefono, p.`email` as profesoremail, p.activo as profesoractivo, cli.codigo as clientecodigo, cli.`nombre` as
    clientenombre, cli.`email` as clienteemail, cli.`telefono` as clientetelefono, cli.identificador as clienteidentificador, cli.`direccion` as clientedireccion,
    cli.`poblacion` as clientepoblacion, cli.`codigopostal` as clientecodigopostal, cli.activo as clienteactivo, i.codigo as imparticioncodigo,
    i.fMatriculacion as imparticionfmatriculacion,
    a.`codigo` as alumnocodigo, a.`nombre` as alunnonombre, a.`apellidos` as alumnoapellidos, a.`fNacimiento` as alumnofnacimiento, a.`direccion` as
    alumnodireccion, a.`poblacion` as alumnopoblacion, a.`codigopostal` as alumnocodigopostal, a.`telefono` as alumnotelefono, a.`email` as alumnoemail,
    a.`dni` as alumnodni, a.`nHermanos` as alumnnonhermanos, a.`activo` as alumnoactivo

```

```

        FROM curso c
    INNER JOIN cliente cli ON cli.codigo = c.cliente_codigo
    INNER JOIN profesor p ON p.codigo = c.profesor_codigo
    INNER JOIN imparticion i on i.curso_codigo = c.codigo
    INNER JOIN alumno a ON a.codigo = i.alumno_codigo
WHERE c.identificador = pidentificador;

```

END\$\$

DROP PROCEDURE IF EXISTS `cursoupdate`\$\$

CREATE DEFINER=`root`@`%` PROCEDURE `cursoupdate` (IN `pcodigo` INT, IN `pnombre` VARCHAR(50), IN `pidentificador` VARCHAR(50), IN `pfinicio` DATE, IN `pffin` DATE, IN `pnhoras` INT, IN `ptemario` VARCHAR(50), IN `pprecio` DOUBLE(8,2), IN `pcliente_codigo` INT, IN `pprofesor_codigo` INT) NO SQL

BEGIN

```

UPDATE curso SET nombre = pnombre, identificador = pidentificador, finicio = pfinicio, fFin = pffin, nHoras = phoras, temario = ptemario, precio = pprecio,
cliente_codigo = pcliente_codigo, profesor_codigo = pprofesor_codigo
WHERE codigo = pcodigo;

```

END\$\$

DROP PROCEDURE IF EXISTS `profesorByDni`\$\$

CREATE DEFINER=`root`@`localhost` PROCEDURE `profesorByDni` (IN `dni` VARCHAR(9)) BEGIN

```

    SELECT `codigo` as profesorcodigo, `NSS` as profesornss, `nombre` as profesornombre, `apellidos` as
    profesorapellidos, `fNacimiento` as profesorfnacimiento, `DNI` as profesordni, `direccion` as profesordireccion, `poblacion` as profesorpoblacion,
    `codigopostal` as profesorcodigopostal, `telefono` as profesortelefono, `email` as profesoremail, activo as profesoractivo
    FROM `profesor`

```

```

    WHERE dni = pdni;

```

END\$\$

DROP PROCEDURE IF EXISTS `profesorCreate`\$\$

CREATE DEFINER=`root`@`localhost` PROCEDURE `profesorCreate` (IN `pnss` VARCHAR(150), IN `pnombre` VARCHAR(150), IN `papellidos` VARCHAR(250), IN `pfNacimiento` DATE, IN `pdni` VARCHAR(9), IN `pdireccion` VARCHAR(150), IN `ppoblacion` VARCHAR(150), IN `pcodigopostal` INT(5), IN `ptelefono` INT(9), IN `pemail` VARCHAR(150), OUT `pcodigo` INT) BEGIN

```

    INSERT INTO `profesor` (
        `nombre`,
        `apellidos`,
        `fNacimiento`,
        `DNI`,

```

```

        `direccion`,
        `poblacion`,
        `codigopostal`,
        `telefono`,
        `email`)
VALUES (
    LOWER(pnombre),
    LOWER(papellidos),

        pfNacimiento,

    LOWER(pdni),
    LOWER(pdireccion),
    LOWER(poblacion),

        pcodigopostal,
        ptelefono,

    LOWER(pemail)
);
SET pcodigo = LAST_INSERT_ID();
if pnss != " THEN
        update profesor set nss = pnss where codigo = pcodigo;
END IF;
END$$

DROP PROCEDURE IF EXISTS `profesorgetAll`$$
CREATE DEFINER=`admin`@`%` PROCEDURE `profesorgetAll` () NO SQL
BEGIN
        SELECT `codigo` as profesorcodigo, `NSS` as profesornss, `nombre` as profesornombre, `apellidos` as
profesorapellidos, `fNacimiento` as profesorfnacimiento, `DNI` as profesordni, `direccion` as profesordireccion, `poblacion` as profesorpoblacion,
`codigopostal` as profesorcodigopostal, `telefono` as profesortelefono, `email` as profesoremail, activo as profesoractivo
        FROM `profesor`
        WHERE profesor.activo = true;
END$$

DROP PROCEDURE IF EXISTS `profesorgetByDni`$$
CREATE DEFINER=`root`@`localhost` PROCEDURE `profesorgetByDni` (IN `pdni` VARCHAR(9)) BEGIN
        SELECT `codigo` as profesorcodigo, `NSS` as profesornss, `nombre` as profesornombre, `apellidos` as
profesorapellidos, `fNacimiento` as profesorfnacimiento, `DNI` as profesordni, `direccion` as profesordireccion, `poblacion` as profesorpoblacion,
`codigopostal` as profesorcodigopostal, `telefono` as profesortelefono, `email` as profesoremail, activo as profesoractivo
        FROM `profesor`
        WHERE dni = pdni;
END$$

DROP PROCEDURE IF EXISTS `profesorgetByld`$$
CREATE DEFINER=`root`@`localhost` PROCEDURE `profesorgetByld` (IN `pcodigo` INT) BEGIN
        SELECT `codigo` as profesorcodigo, `NSS` as profesornss, `nombre` as profesornombre, `apellidos` as
profesorapellidos, `fNacimiento` as profesorfnacimiento, `DNI` as profesordni, `direccion` as profesordireccion, `poblacion` as profesorpoblacion,
`codigopostal` as profesorcodigopostal, `telefono` as profesortelefono, `email` as profesoremail, activo as profesoractivo
        FROM `profesor`
        WHERE codigo = pcodigo;
END$$

DROP PROCEDURE IF EXISTS `profesorgetByNss`$$
CREATE DEFINER=`root`@`localhost` PROCEDURE `profesorgetByNss` (IN `pnss` VARCHAR(12)) BEGIN
        SELECT `codigo` as profesorcodigo, `NSS` as profesornss, `nombre` as profesornombre, `apellidos` as
profesorapellidos, `fNacimiento` as profesorfnacimiento, `DNI` as profesordni, `direccion` as profesordireccion, `poblacion` as profesorpoblacion,
`codigopostal` as profesorcodigopostal, `telefono` as profesortelefono, `email` as profesoremail, activo as profesoractivo
        FROM `profesor`
        WHERE nss = pnss;
END$$

DROP PROCEDURE IF EXISTS `profesorUpdate`$$
CREATE DEFINER=`root`@`localhost` PROCEDURE `profesorUpdate` (IN `pnss` VARCHAR(12), IN `pnombre` VARCHAR(150), IN `papellidos`
VARCHAR(250), IN `pfNacimiento` DATE, IN `pdni` VARCHAR(9), IN `pdireccion` VARCHAR(250), IN `ppoblacion` VARCHAR(150), IN `pcodigopostal`
INT(5), IN `ptelefono` INT(9), IN `pemail` VARCHAR(150)) BEGIN
UPDATE `profesor`
SET `NSS`=pnss,
    `nombre`=pnombre,
    `apellidos`=papellidos,
    `fNacimiento`=pfNacimiento,
    `DNI`=pdni,
    `direccion`=pdireccion,
    `poblacion`=ppoblacion,
    `codigopostal`=pcodigopostal,
    `telefono`=ptelefono,
    `email`=pemail
WHERE codigo = pcodigo;

END$$

DELIMITER ;

-- -----
--
-- Estructura de tabla para la tabla `alumno`
--

```

```

DROP TABLE
IF EXISTS `alumno`;
CREATE TABLE IF NOT EXISTS `alumno` (
  `codigo` int(11) NOT NULL AUTO_INCREMENT COMMENT 'el campo clave de la tabla. Es auto generado.',
  `nombre` varchar(50) COLLATE utf8_bin NOT NULL,
  `apellidos` varchar(250) COLLATE utf8_bin NOT NULL,
  `fNacimiento` date DEFAULT NULL,
  `direccion` varchar(250) COLLATE utf8_bin DEFAULT NULL,
  `poblacion` varchar(150) COLLATE utf8_bin DEFAULT NULL,
  `codigopostal` int(5) UNSIGNED ZEROFILL DEFAULT NULL,
  `telefono` int(9) NOT NULL,
  `email` varchar(150) COLLATE utf8_bin NOT NULL,
  `dni` varchar(9) COLLATE utf8_bin NOT NULL,
  `nHermanos` int(2) DEFAULT '0',
  `activo` tinyint(1) DEFAULT '1',
  PRIMARY KEY (`codigo`),
  UNIQUE KEY `dni_UNIQUE` (`dni`)
) ENGINE=InnoDB AUTO_INCREMENT=7 DEFAULT CHARSET=utf8 COLLATE=utf8_bin;

--
-- Volcado de datos para la tabla `alumno`
--

INSERT INTO `alumno` (`codigo`, `nombre`, `apellidos`, `fNacimiento`, `direccion`, `poblacion`, `codigopostal`, `telefono`, `email`, `dni`, `nHermanos`,
`activo`) VALUES
(0, 'alumno', 'sin asignar', NULL, NULL, NULL, 00000, 0, 'aaaaaaa@aaaaa.com', '0000000x', 0, 0),
(1, 'sergio', 'aparicio vargas', '1977-12-01', ' ', ' ', 00000, 944110293, 'aaaa@aaaaa.com', '44974398z', 0, 1),
(2, 'maite', 'monasterio', '1986-11-11', ' ', ' ', 48007, 944110293, 'mmonasterio@gmail.com', '16071559x', 0, 1),
(4, 'enrique javier', 'ruiz jimenez', '2017-02-14', ' ', ' ', 00048, 944110239, 'enrique@gmail.com', '45677362y', 0, 1);

-----

--
-- Estructura de tabla para la tabla `cliente`
--

DROP TABLE IF EXISTS `cliente`;
CREATE TABLE IF NOT EXISTS `cliente` (
  `codigo` int(11) NOT NULL AUTO_INCREMENT,
  `nombre` text COLLATE utf8_bin NOT NULL,
  `email` varchar(150) COLLATE utf8_bin NOT NULL,
  `telefono` int(9) NOT NULL,
  `direccion` varchar(250) COLLATE utf8_bin DEFAULT NULL,
  `poblacion` varchar(150) COLLATE utf8_bin DEFAULT NULL,
  `codigopostal` int(5) UNSIGNED ZEROFILL DEFAULT NULL,
  `identificador` varchar(15) COLLATE utf8_bin NOT NULL,
  `activo` tinyint(4) DEFAULT '1',
  PRIMARY KEY (`codigo`)
) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=utf8 COLLATE=utf8_bin;

--
-- Volcado de datos para la tabla `cliente`
--

INSERT INTO `cliente` (`codigo`, `nombre`, `email`, `telefono`, `direccion`, `poblacion`, `codigopostal`, `identificador`, `activo`) VALUES
(0, 'Sin Definir', 'aaaaaa@aaaaa.com', 9444, NULL, NULL, NULL, '#####', 0),
(1, 'SERIKAT CONSULTORIA E INFORMÁTICA, S.A.', 'info@serikat.es', 944250100, 'c/ Ercilla 19', 'Bilbao', 48009, 'A-48476006', 1),
(2, 'lanbide - servicio vasco de empleo', 'info@lanbide.net', 945160601, 'Jose Atxotegi 1', 'Vitoria-Gazteiz', 01009, 'Q0100571', 1),
(3, 'hobetuz', 'bizkaia@hobetuz.eus', 944150808, 'gran vía, 35-6º', NULL, NULL, 'g48850127', 1);

-----

--
-- Estructura de tabla para la tabla `curso`
--

DROP TABLE IF EXISTS `curso`;
CREATE TABLE IF NOT EXISTS `curso` (
  `codigo` int(11) NOT NULL AUTO_INCREMENT,
  `nombre` varchar(100) COLLATE utf8_bin NOT NULL,
  `identificador` varchar(12) COLLATE utf8_bin NOT NULL,
  `fInicio` date DEFAULT NULL,
  `fFin` date DEFAULT NULL,
  `nHoras` int(4) NOT NULL,
  `temario` text COLLATE utf8_bin,
  `activo` tinyint(4) DEFAULT '1',
  `cliente_codigo` int(11) DEFAULT NULL,
  `precio` double(8,2) DEFAULT '0.00',
  `profesor_codigo` int(11) DEFAULT NULL,
  PRIMARY KEY (`codigo`),
  KEY `fk_curso_cliente_codigo_idx` (`cliente_codigo`),
  KEY `fk_curso_profesor_codigo_idx` (`profesor_codigo`)
) ENGINE=InnoDB AUTO_INCREMENT=5 DEFAULT CHARSET=utf8 COLLATE=utf8_bin;

--
-- Volcado de datos para la tabla `curso`
--

INSERT INTO `curso` (`codigo`, `nombre`, `identificador`, `fInicio`, `fFin`, `nHoras`, `temario`, `activo`, `cliente_codigo`, `precio`, `profesor_codigo`)

```

```

VALUES
(1, 'Desarrollo de Aplicaciones con Tecnologías Web', '18482675', '2017-01-09', '2017-06-13', 510, NULL, 1, 2, 300000.00, 1),
(2, 'Desarrollo de Bases de Datos y Programación orientada a Objetos', '18488225', '2017-02-20', '2017-09-29', 630, 'IFCD0112_FIC.pdf', 1, 2, 400000.00, 1),
(3, 'Publicidad en internet', '18482678', '2017-03-27', '2017-03-30', 10, NULL, 1, 3, 1500.00, 1),
(4, 'Programación en Bases de Datos relaciones con Oracle 12c', 'CI67', '2017-05-02', '2017-05-30', 30, 'CI67.pdf', 1, 3, 3500.00, 1);

-----

--
-- Estructura de tabla para la tabla `imparticion`
--

DROP TABLE IF EXISTS `imparticion`;
CREATE TABLE IF NOT EXISTS `imparticion` (
  `codigo` int(11) NOT NULL AUTO_INCREMENT,
  `curso_codigo` int(11) NOT NULL,
  `alumno_codigo` int(11) NOT NULL,
  `fMatriculacion` date DEFAULT NULL,
  PRIMARY KEY (`codigo`),
  KEY `fk_imparticion_alumno_codigo_idx` (`alumno_codigo`),
  KEY `fk_imparticion_curso_codigo_idx` (`curso_codigo`)
) ENGINE=InnoDB AUTO_INCREMENT=51 DEFAULT CHARSET=utf8 COLLATE=utf8_bin;

--
-- Volcado de datos para la tabla `imparticion`
--

INSERT INTO `imparticion` (`codigo`, `curso_codigo`, `alumno_codigo`, `fMatriculacion`) VALUES
(9, 3, 2, NULL),
(10, 3, 4, NULL),
(15, 2, 1, NULL),
(16, 2, 2, NULL),
(17, 2, 4, NULL),
(48, 1, 1, NULL),
(49, 1, 2, NULL),
(50, 1, 4, NULL);

-----

--
-- Estructura de tabla para la tabla `profesor`
--

DROP TABLE IF EXISTS `profesor`;
CREATE TABLE IF NOT EXISTS `profesor` (
  `codigo` int(11) NOT NULL AUTO_INCREMENT,
  `NSS` bigint(12) DEFAULT NULL,
  `nombre` varchar(50) COLLATE utf8_bin NOT NULL,
  `apellidos` varchar(250) COLLATE utf8_bin NOT NULL,
  `fNacimiento` date DEFAULT NULL,
  `DNI` varchar(9) COLLATE utf8_bin NOT NULL,
  `direccion` varchar(250) COLLATE utf8_bin DEFAULT NULL,
  `poblacion` varchar(150) COLLATE utf8_bin DEFAULT NULL,
  `codigopostal` int(5) UNSIGNED ZEROFILL DEFAULT NULL,
  `telefono` int(9) NOT NULL,
  `email` varchar(150) COLLATE utf8_bin NOT NULL,
  `activo` tinyint(4) DEFAULT '1',
  PRIMARY KEY (`codigo`)
) ENGINE=InnoDB AUTO_INCREMENT=2 DEFAULT CHARSET=utf8 COLLATE=utf8_bin;

--
-- Volcado de datos para la tabla `profesor`
--

INSERT INTO `profesor` (`codigo`, `NSS`, `nombre`, `apellidos`, `fNacimiento`, `DNI`, `direccion`, `poblacion`, `codigopostal`, `telefono`, `email`, `activo`)
VALUES
(0, 0, 'profesor', 'sin asignar', NULL, '00000000z', NULL, NULL, NULL, 0, 'aaaaaaaa@aaaaa.aaa', 0),
(1, 481234567840, 'Ander', 'ipartek', '1976-11-24', '30693142x', 'Av. Mazustegi 9', 'Bilbao', 48009, 944110293, 'auraga@ipartek.com', 1);

--
-- Restricciones para tablas volcadas
--

--
-- Filtros para la tabla `curso`
--

ALTER TABLE `curso`
  ADD CONSTRAINT `fk_curso_cliente_codigo` FOREIGN KEY (`cliente_codigo`) REFERENCES `cliente` (`codigo`) ON DELETE NO ACTION ON UPDATE NO ACTION,
  ADD CONSTRAINT `fk_curso_profesor_codigo` FOREIGN KEY (`profesor_codigo`) REFERENCES `profesor` (`codigo`) ON DELETE NO ACTION ON UPDATE NO ACTION;

--
-- Filtros para la tabla `imparticion`
--

ALTER TABLE `imparticion`
  ADD CONSTRAINT `fk_imparticion_alumno_codigo` FOREIGN KEY (`alumno_codigo`) REFERENCES `alumno` (`codigo`) ON DELETE NO ACTION ON UPDATE NO ACTION,
  ADD CONSTRAINT `fk_imparticion_curso_codigo` FOREIGN KEY (`curso_codigo`) REFERENCES `curso` (`codigo`) ON DELETE NO ACTION ON UPDATE NO ACTION;

```



```

UPDATE NO ACTION;

/*I40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*I40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*I40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;

```

Script para crear nueva tabla.

```

-- MySQL Workbench Synchronization
-- Generated: 2021-02-09 23:06
-- Model: New Model
-- Version: 1.0
-- Project: Name of the project
-- Author: Curso JAVA

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';

CREATE TABLE IF NOT EXISTS `gestiondocente`.`resenya` (
  `codigo` INT(11) NOT NULL AUTO_INCREMENT,
  `descripcion` VARCHAR(250) NOT NULL,
  `alumno_codigo` INT(11) NOT NULL,
  `curso_codigo` INT(11) NOT NULL,
  PRIMARY KEY (`codigo`),
  INDEX `fk_resenya_alumno1_idx` (`alumno_codigo` ASC) VISIBLE,
  INDEX `fk_resenya_curso1_idx` (`curso_codigo` ASC) VISIBLE,
  CONSTRAINT `fk_resenya_alumno1`
    FOREIGN KEY (`alumno_codigo`)
      REFERENCES `gestiondocente`.`alumno` (`codigo`)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION,
  CONSTRAINT `fk_resenya_curso1`
    FOREIGN KEY (`curso_codigo`)
      REFERENCES `gestiondocente`.`curso` (`codigo`)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_bin;

SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;

```

UF2176

SQL listar últimos 5 cursos

```
SELECT c.codigo, c.nombre, c.identificador, c.fInicio, c.fFin, c.nHoras, c.temario, c.activo,
       (SELECT nombre FROM cliente where codigo = c.cliente_codigo) as cliente, c.precio,
       (SELECT nombre FROM profesor where codigo = c.profesor_codigo) as profesor
FROM curso c
ORDER BY c.fInicio DESC
LIMIT 5;
```

SQL listar últimos 5 usuarios creados

```
SELECT nombre, apellidos, fnacimiento, direccion, poblacion, codigoPostal, telefono, email, dni FROM alumno
ORDER BY codigo DESC
LIMIT 5;
```

	nombre	apellidos	fnacimiento	direccion	poblacion	codigoPostal	telefono	email	dni
▶	Lechuck	El Fantasma	1652-01-06	NULL	NULL	NULL	55555555	lechuck@piratas.com	87654321r
	Guybrush	Threedwood	1652-01-06	NULL	NULL	NULL	55555555	guybrush@piratas.com	12345678u
	enrique javier	ruiz jimenez	2017-02-14			00048	944110239	enrique@gmail.com	45677362y
	Perico	De Los Palotes	NULL	NULL	NULL	NULL	944110240	perico@delospalotes.com	12345678f
	maite	monasterio	1986-11-11			48007	944110293	mmonasterio@gmail.com	16071559x

SQL detalle curso + número de reseñas

```
SELECT c.nombre, c.identificador, c.nHoras as horas, p.nombre as profesor, (SELECT COUNT(*) FROM resenya
WHERE curso_codigo = c.codigo) AS numero_resenyas
FROM curso c
LEFT JOIN profesor p
ON c.profesor_codigo = p.codigo
GROUP BY c.codigo;
```

	nombre	identificador	horas	profesor	numero_resenyas
▶	Desarrollo de Aplicaciones con Tecnologías Web	18482675	510	Ander	0
	Desarrollo de Bases de Datos y Programación or...	18488225	630	Ander	0
	Publicidad en internet	18482678	10	Ander	0
	Programación en Bases de Datos relaciones con...	CI67	30	Ander	0
	Iniciación a la piratería	18482679	50	Arturo	3
	Piratería avanzada	18482680	300	Arturo	2
	Elaboración de Grog	18482681	30	Arturo	1

SQL listado usuarios ordenado por número de reseñas

```
SELECT a.nombre, a.apellidos, a.telefono, a.email, count(r.codigo) as numero_resenyas
FROM alumno a
LEFT JOIN resenya r
ON a.codigo = r.alumno_codigo
GROUP BY a.codigo
ORDER BY numero_resenyas DESC;
```

	nombre	apellidos	telefono	email	numero_resenyas
▶	Lechuck	El Fantasma	55555555	lechuck@piratas.com	3
	Guybrush	Threedwood	55555555	guybrush@piratas.com	2
	Perico	De Los Palotes	944110240	perico@delospalotes.com	1
	alumno	sin asignar	0	aaaaaaa@aaaaa.com	0
	sergio	aparicio vargas	944110293	aaaa@aaaa.com	0
	maite	monasterio	944110293	mmonasterio@gmail.com	0
	enrique javier	ruiz jimenez	944110239	enrique@gmail.com	0

SQL Resumen de número de cursos de todos los años

```
SELECT YEAR(finicio) as anyo, count(*) as numero_cursos FROM curso
GROUP BY YEAR(finicio);
```

	anyo	numero_cursos
▶	2017	4
	2019	2
	2021	1

UF2177

Lista de cursos

GET ▼ http://localhost:8080/api/cursos Send

200 OK53.5 ms1234 B

Body ▼

Auth ▼

Query

Header

Docs

Preview ▼

Header ⌵

Cookie

Timeline



Select a body type from above

```
1 {
2   {
3     "codigo": 1,
4     "nombre": "Desarrollo de Aplicaciones con Tecnologías Web",
5     "identificador": "12482675",
6     "profesor": {
7       "codigo": 1,
8       "nombre": "Ander",
9       "apellidos": "Ipartak",
10      "cursos": []
11    },
12    "horas": 510
13  },
14  {
15    "codigo": 2,
16    "nombre": "Desarrollo de Bases de Datos y Programación orientada a Objetos",
17    "identificador": "12482225",
18    "profesor": {
19      "codigo": 1,
20      "nombre": "Ander",
21      "apellidos": "Ipartak",
22      "cursos": []
23    },
24    "horas": 630
25  },
26  {
27    "codigo": 3,
28    "nombre": "Publicidad en Internet",
29    "identificador": "12482678",
30    "profesor": {
31      "codigo": 1,
32      "nombre": "Ander",
33      "apellidos": "Ipartak",
34      "cursos": []
35    },
36    "horas": 10
37  },
38  {
39    "codigo": 4,
40    "nombre": "Programación en Bases de Datos relaciones con Oracle 12c",
41    "identificador": "C167",
42    "profesor": {
43      "codigo": 1,
44      "nombre": "Ander",
45      "apellidos": "Ipartak",
46      "cursos": []
47    },
48    "horas": 30
49  },
50  {
51    "codigo": 5,
52    "nombre": "Introducción a la piratería",
53    "identificador": "12482679",
54    "profesor": {
55      "codigo": 2,
56      "nombre": "Arturo",
57      "apellidos": "Montañez",
58      "cursos": []
59    },
60    "horas": 50
61  },
62  {
63    "codigo": 6,
64    "nombre": "Piratería avanzada",
65    "identificador": "12482680",
66    "profesor": {
67      "codigo": 2,
68      "nombre": "Arturo",
69      "apellidos": "Montañez",
70      "cursos": []
71    },
72    "horas": 300
73  },
74  }
```

Detalle curso

GET ▼ http://localhost:8080/api/cursos/5 Send

200 OK 15.9 ms 166 B

Body ▼ Auth ▼ Query Header Docs

Preview ▼ Header 3 Cookie Timeline

```
1 {  
2   "codigo": 5,  
3   "nombre": "Iniciacion a la pirateria",  
4   "identificador": "18482679",  
5   "profesor": {  
6     "codigo": 2,  
7     "nombre": "Arturo",  
8     "apellidos": "Montañez",  
9     "cursos": []  
10  },  
11   "nhoras": 50  
12 }
```

Crear nueva reseña

POST ▼ http://localhost:8080/api/resenyas Send

201 Created 23.1 ms 71 B

JSON ▼ Auth ▼ Query Header 1 Docs

Preview ▼ Header 3 Cookie Timeline

```
1 {  
2   "descripcion": "Muy util",  
3   "alumnoCodigo": 7,  
4   "cursoCodigo": 7  
5 }
```

```
1 {  
2   "codigo": 10,  
3   "descripcion": "Muy util",  
4   "alumnoCodigo": 7,  
5   "cursoCodigo": 7  
6 }
```

Modificar Reseña

PUT ▼ http://localhost:8080/api/resenyas/14 Send

200 OK 16.4 ms 93 B

JSON ▼ Auth ▼ Query Header 1 Docs

Preview ▼ Header 3 Cookie Timeline

```
1 {  
2   "codigo": 14,  
3   "descripcion": "Hacia falta un curso como este",  
4   "alumnoCodigo": 5,  
5   "cursoCodigo": 7  
6 }  
7
```

```
1 {  
2   "codigo": 14,  
3   "descripcion": "Hacia falta un curso como este",  
4   "alumnoCodigo": 5,  
5   "cursoCodigo": 7  
6 }
```

Eliminar Reseña

DELETE	http://localhost:8080/api/resenyas/9	Send	204 No Content	15.1 ms	0 B
Body	Auth	Query	Header	Docs	Preview
				Header 1	Cookie
				Timeline	
No body returned for response					

Documentar servicio REST

swagger: "2.0"

info:

description: "Prueba final módulo MF0226_3."

version: "1.0.0"

title: "MF0226-3"

host: "localhost:8080"

basePath: "/api"

tags:

- name: "curso"
description: "Cursos de la academia"
- name: "resenya"
description: "Reseñas de los cursos"
- name: "profesor"
description: "Profesores de la academia"

paths:

/curso:

get:

tags:

- "curso"

summary: "Listado de cursos"

description: "Listado de todos los cursos"

operationId: "obtenerTodos"

produces:

- "application/xml"

- "application/json"

responses:

"200":

description: "Operacion exitosa"

schema:

\$ref: "/cursos"

"400":

description: "No se encuentra"

"404":

description: "No se encuentra"

/cursos/{id}:

get:

tags:

- "curso"

summary: "Encontrar curso por id"

description: "Returns a single pet"

operationId: "obtenerPorId"

produces:

- "application/xml"

- "application/json"

parameters:

- name: "id"

in: "path"

description: "id del curso"

required: true

type: "integer"

format: "int64"

responses:
"200":
 description: "soperación exitosa"
 schema:
 \$ref: "/cursos/{1}"
"400":
 description: "Id no valida"
"404":
 description: "Curso no encontrado"

/resenya:
post:
 tags:
 - "resenya"
 summary: "Insertar reseña"
 description: ""
 operationId: "insertar"
 produces:
 - "application/xml"
 - "application/json"
 responses:
 "200":
 description: "operacion exitosa"
 schema:
 \$ref: "/resenyas"
 "400":
 description: "Reseña invalida"

/resenya/{id}:
put:
 tags:
 - "resenya"
 summary: "Modifica una resena existente"
 description: ""
 operationId: "modificar"
 consumes:
 - "application/json"
 - "application/xml"
 produces:
 - "application/xml"
 - "application/json"
 parameters:
 - name: "id"
 in: "path"
 description: "The name that needs to be fetched. Use user1 for testing. "
 required: true
 type: "integer"
 responses:
 "400":
 description: "ID no valido"
 "404":
 description: "Reseña no encontrada"
 "405":
 description: "Excepcion de validación"

delete:
 tags:
 - "resenya"
 summary: "Borrar una reseña por su ID"
 description: "Requiere valores enteros positivos"
 operationId: "borrar"
 produces:
 - "application/xml"
 - "application/json"
 parameters:
 - name: "id"
 in: "path"
 description: "ID de la reseña que se quiere borrar"

required: true
type: "integer"
minimum: 1.0
format: "int64"
responses:
"400":
description: "ID ivalida"
"404":
description: "Reseña no encontrada"

definitions:
Profesor:
type: "object"
properties:
codigo:
type: "integer"
format: "int64"
nombre:
type: "string"
apellidos:
type: "string"
xml:
name: "Profesor"
Curso:
type: "object"
properties:
codigo:
type: "integer"
format: "int64"
nombre:
type: "string"
identificador:
type: "string"
nHoras:
type: "integer"
format: "int64"
xml:
name: "Curso"
Resenya:
type: "object"
properties:
codigo:
type: "integer"
format: "int64"
descripcion:
type: "string"

alumnoCodigo:
type: "integer"
format: "int32"
cursoCodigo:
type: "integer"
format: "int32"
xml:
name: "Resenya"