USAR SCRIPT tablas\_ejerc\_crear\_datos.sql

1. Crea la vista DEP30 con los campos APELLIDO, OFICIO, SALARIO de los empleados del departamento 30 de la tabla EMPLEADOS. Créala dando nombre a las columnas: APELLIDO (APE), OFICIO (OFI), SALARIO (SAL)

CREATE VIEW DEP30 (APE, OFI, SAL)

AS SELECT apellido, oficio, salario FROM empleados

Where dept\_no=30;

1. Inserta los siguientes valores: ¿Qué sucede?

INSERT INTO dep30 VALUES (‘Pérez’, ‘EMPLEADO’,1300);

Dará error pq no se da valor a las columnas definidas como not null EMP\_NO y DEPT\_NO

1. Incrementa a través de la vista los salarios de los empleados del departamento 30 en 100 eruros.

UPDATE dep30 SET sal=sal+100;

1. ¿Qué haría la siguiente instrucción?

DELETE FROM dep30

Borraría todas las filas de la tabla empleados del departamento 30.

1. Crea una vista que contenga los datos de los empleados del departamento 10 con salario > 1200.

CREATE VIEW actividad5

AS

SELECT apellido, oficio, salario FROM empleados

WHERE dept\_no = 10 AND salario > 1200;

1. Crea a partir de las tablas EMPLEADOS y DEPARTAMENTOS una vista que contenga los campos: EMP\_NO, APELLIDO, DEPT\_NO Y DNOMBE.

CREATE VIEW EMP\_DEP (emp\_no, apellido, dept\_no, dnombre) AS

SELECT emp\_no,apellido, e.dept\_no, dnombre FROM empleados e, departamentos d

where e.dept\_no =d.dept\_no

1. Introduce el siguiente registro:

INSERT INTO emp\_dep VALUES (2222,’SUELA’,20,’INVESTIGACIÓN’);

Da error porque se intentan actualizar valores de dos tablas. Si fuese de una sola y los campos que no se rellenan no fuesen obligatorios dejaría.

1. Crear una vista llamada PAGOS de los empleados cuyo departamento sea el 10. Las columnas de la vista se llamarán: NOMBRE, SAL\_MES, SAL\_ANT Y DEPT\_NO cumpliendo:
   1. NOMBRE es la columnas apellido en minúsculas.
   2. SAL\_MES es la columna salario de la tabla empleados (salario mensual)
   3. SAL\_AN es el salario anual.

CREATE VIEW PAGOS (nombre,SAL\_MES, SAL\_AN, dept\_no) AS

SELECT LOWER(apellido), salario, salario\*12, dept\_no FROM empleados

WHERE dept\_no=10;

¿Bajo qué condiciones podré modificar la vista PAGOS? Pon ejemplos

Podremos modificarla siempre y cuando la columna que se va a modificar no sea la columna expresada en forma de cálculo(SAL\_AN) o la que se creó mediante la función LOWER()

UPDATE pagos SET sal\_mes =5000 Where nombre=’muñoz’;

1. Crea la vista VMEDIA a partir de las tablas EMPLEADOS y DEPARTAMENTOS. Esta vista contendrá por cada departamento el número de departamento, el nombre, la media de salario y el máximo salario. Prueba a hacer inserciones, modificaciones y borrados en la vista.

CREATE VIEW vmedia (ndep, nomdep, media, maximo) AS

SELECT e.dept\_no, dnombre, AVG(salario), MAX(salario)

FROM empleados e, departamentos d

WHERE e.dept\_no=d.dept\_no

GROUP BY e.dept\_no, dnombre;

*Lógicamente, cualquier manipulación de datos que se haga sobre esta vista será errónea.*

1. Borra las vistas creadas vmedia y pagos.

DROP VIEW vmedia, Pagos;

**SOBRE LAS TABLAS PERSONAL, PROFESORES Y CENTROS**

1. Crear una vista que se llame CONSERJES que contenga el nombre del centro y el nombre de sus conserjes.

CREATE VIEW conserjes AS

SELECT nombre,apellidos FROM centros, personal

WHERE funcion = 'CONSERJE'

AND centros.cod\_centro = personal.cod\_centro;

1. Añadir a la tabla PROFESORES una columna llamada COD\_ASIG con dos posiciones numéricas.

ALTER TABLE profesores ADD cod\_asig TINYINT ;

1. Crear la tabla TASIG con las siguientes columnas: COD\_ASIG numérico, 2 posiciones no admite nulos y NOM\_ASIG cadena de 20 caracteres.

CREATE TABLE tasig (

cod\_asig TINYINT NOT NULL,

nom\_asig VARCHAR(20)

);

1. Añadir la restricción de clave primaria a la columna COD\_ASIG de la tabla TASIG.

ALTER TABLE tasig ADD CONSTRAINT pk\_tasig PRIMARY KEY (cod\_asig);

1. Añadir la restricción de clave ajena a la columna COD\_ASIG de la tabla PROFESORES. Visualiza el nombre de las restricciones y las columnas y las columnas afectadas para las tablas TASIG y PROFESORES.

ALTER TABLE profesores ADD CONSTRAINT

fk\_tasig FOREIGN KEY (cod\_asig)

REFERENCES tasig (cod\_asig);

1. Crea una vista de nombre CENTROSPROFESORES que contenga el nombre del centro, los apellidos y la especialidad del profesor.

CREATE VIEW centrosprofesores AS

SELECT nombre,apellidos, especialidad

FROM centros, profesores

WHERE centros.cod\_centro = profesores.cod\_centro;

1. Crea una vista de nombre PROFESORESCENTROS que contenga por cada código de centro el número de profesores que hay.

CREATE OR REPLACE VIEW profesorescentros (cod, num) AS

SELECT centros.cod\_centro, count(\*)

FROM centros, profesores

WHERE centros.cod\_centro = profesores.cod\_centro

GROUP BY centros.cod\_centro;

1. Crear tabla de nombre AUDITORIA con tres columnas, una contendrá datos de tipo fecha, otra contendrá la hora y la tercera columna contendrá una cadena de caracteres.

CREATE TABLE auditoria(

nombre VARCHAR(15),

fecha DATE,

hora TIME

);

**MAS EJERCICIOS DE CREACCIÓN DE OBJETOS**

1. Crea la tabla FABRICANTES con las siguientes columnas y restricciones:

|  |  |
| --- | --- |
| cod\_fabricante TINYINT  nombre VARCHAR(15)  pais VARCHAR(15) | * La clave primaria es COD\_FABRICANTE. No puede ser nula. Se define como autoincremental. * Las columnas NOMBRE y PAIS han de almacenarse en mayúscula. * Valor por defecto para el país “ESPAÑA” |

CREATE TABLE fabricantes (

cod\_fabricante TINYINT NOT NULL IDENTITY PRIMARY KEY,

nombre VARCHAR(15),

pais VARCHAR(15) DEFAULT 'ESPAÑA',

CONSTRAINT CK\_NO1 CHECK(nombre = UPPER(nombre)),

CONSTRAINT CK\_NO2 CHECK(pais = UPPER(pais))

);

1. Crea la tabla ARTICULOS con las siguientes columnas y restricciones:

|  |  |
| --- | --- |
| articulo VARCHAR(20)  cod\_fabricante TINYINT  peso TINYINT  categoria VARCHAR(10)  precio\_venta FLOAT  precio\_costo FLOAT  existencias INT | * La clave primaria está formada por las columnas: ARTICULO, COD\_FABRICANTE, PESO y CATEGORIA. No pueden tener valor nulo. * PRECIO\_VENTA, PRECIO\_COSTO y PESO han de ser > 0. * CATEGORIA ha de ser 'Primera', 'Segunda' o 'Tercera'. |

CREATE TABLE articulos(

articulo VARCHAR(20) NOT NULL,

cod\_fabricante TINYINT NOT NULL,

peso TINYINT NOT NULL,

categoria VARCHAR(10) NOT NULL,

precio\_venta FLOAT,

precio\_costo FLOAT,

existencias INT,

CONSTRAINT CK1\_AR CHECK (peso > 0),

CONSTRAINT CK2\_AR CHECK (precio\_venta > 0),

CONSTRAINT CK3\_AR CHECK (precio\_costo > 0),

CONSTRAINT PK\_ART PRIMARY KEY

(articulo,cod\_fabricante,peso,categoria),

CONSTRAINT CK\_CAT CHECK(categoria IN('Primera','Segunda', 'Tercera'))

);

1. Añade a la tabla FABRICANTES una nueva columna de tipo VARCHAR(10) para que almacene la clase de fabricante y la restricción UNIQUE a la columna NOMBRE.

ALTER TABLE fabricantes ADD clase VARCHAR(10);

ALTER TABLE fabricantes ADD constraint U\_nombre UNIQUE (nombre);

1. Modifica la columna PRECIO\_VENTA de la tabla ARTICULOS a tipo de dato DECIMAL(7,2)

ALTER TABLE articulos ALTER COLUMN precio\_venta DECIMAL(7,2),

NO se puede porque existe una restricción tipo check sobre ese campo. Habría que quitar primero la restricción, hacerlo y volver a ponerla.

ALTER TABLE articulos DROP ck2\_AR

ALTER TABLE articulos ALTER COLUMN precio\_venta DECIMAL(7,2)

ALTER TABLE articulos ADD CONSTRAINT CK2\_AR CHECK (precio\_venta > 0)

1. Añade la restricción de clave ajena a la columna COD\_ FABRICANTE de la tabla ARTICULOS para que referencie a la tabla FABRICANTES. Realiza un borrado en cascada.

ALTER TABLE articulos ADD CONSTRAINT claveajena

FOREIGN KEY (cod\_fabricante)

REFERENCES fabricantes (cod\_fabricante)

ON DELETE CASCADE;