

Notas útiles para resolver los ejercicios

- El formato que se utiliza en los ejercicios para describir las tablas del modelo relacional es el siguiente:

NOMBRE_TABLA (campo1, campo2 ...).

➡ Los campos que son **clave primaria** (PK, primary key) aparecen en negrita y subrayado.

➡ Los campos que son **claves externas** (FK, foreign key) aparecen en negrita. Además se aporta información adicional (tabla a la que referencia)

- **Formato de fecha:** para ver el formato admitido de fecha puedes ver la fecha actual con la instrucción:

```
select sysdate from dual;
```

Para modificar el formato de fecha admitido puedes utilizar la instrucción (modificando el formato a tu conveniencia):

```
Alter session set nls_date_format='DD-MM-YYYY';
```

- Comandos básicos LMD para manipular datos:

Visualiza todos los datos de una tabla:

```
select * from nombre_tabla;
```

Insertar datos en una tabla (todos los campos del registro):

```
insert into nombre_tabla values (valor1, valor2...)
```

Insertar datos en una tabla (no todos los campos del registro):

```
insert into nombre_tabla(campo1,campo3...) values (valor1, valor3...)
```

Ejercicios: crear tablas sencillas

1. Crea las siguientes tablas.

a) REVISTA (codigo, sala, estanteria)

- Código es un número de 4 dígitos
- Sala es un nombre (máximo 10 caracteres de longitud)
- Estantería es un número de 1 dígito.

Comprueba que se ha creado bien con DESCRIBE.

Solución:

```
create table REVISTA(  
codigo number(4) primary key,  
sala varchar2(10),  
estanteria number(1));
```

b) ACTOR (DNI, nombre_artístico, fecha_nac)

- DNI es un texto con letras y números de 10 caracteres
- Nombre_artístico es un texto de 20 caracteres como máximo
- Fecha_nac es la fecha de nacimiento

Comprueba que se ha creado bien con DESCRIBE

Solución:

```
create table actor(  
DNI varchar2(10) primary key,  
nombre_artistico varchar2(20),  
fecha_nac date);
```

c) Borra las tablas creadas (comando DROP).

Solución:

```
drop table revista;  
drop table actor;
```

Ejercicios: crear tablas. Campos numéricos con precisión y escala

2. Crea la siguiente tabla, en la que se almacenarán datos de muebles.

MUEBLE (cod, alto, ancho, fondo, precio, stock)

- Cod es un texto de 3 caracteres
- Alto, ancho, fondo, precio y stock son números
- El alto, ancho y fondo permiten 2 dígitos enteros y 2 decimales
- El precio permite 3 dígitos enteros y 2 decimales
- El stock permite 4 dígitos sin decimales.

Inserta los siguientes datos. Si alguno da algún error indica dónde está el problema.

Mueble 1: cod=12R
 Alto=60,03
 Ancho=25,30
 Fondo=50,05
 Precio= 102,99
 Stock=124

Mueble 2: cod=03S
 Alto=110,00
 Ancho=10,55
 Fondo=33,10
 Precio= 78,99
 Stock=35

Solución:

```
create table MUEBLE(
  cod varchar2(3) primary key,
  alto number(4,2),
  ancho number(4,2),
  fondo number (4,2),
  precio number(5,2),
  stock number(4)
);
insert into mueble values ('12R',60.03,25.30,50.05,102.99,124);
insert into mueble values ('03S',110.00,10.55,33.10,78.99,35); FALLA POR EL ALTO
```

3. Crea la siguiente tabla, para guardar artículos de una tienda:

PRECIO_ART (articulo, valor, fecha)

```
create table precio_art(
  articulo varchar2(10) primary key,
  valor number(5,-2),
  fecha date);
```

Inserta los siguientes datos. Si alguno da algún error indica dónde está el problema (ver [nota formato de fechas](#))

Registro 1: articulo = DELGER
 Valor = 10565
 Fecha = 23 de marzo de 2017 (modifica el formato)

Registro 2: articulo = SYNTR
 Valor = 100,72
 Fecha = 10 de enero de 2020 (modifica el formato)

Comprueba cómo se han guardado los datos en la tabla mediante la instrucción:

```
select * from precio;
```

¿Se ha modificado algún valor? ¿por qué?

Solución:

```
insert into precio_art values('DELGER',10565,'23/03/17');
insert into precio_art values('SYNTR',100.72,'10/01/20');
Select * from precio_art;
```

ARTICULO	VALOR	FECHA
DELGER	10600	23/03/17
SYNTR	100	10/01/20

Se modifica el campo valor, ya que al estar definido como valor number(5,-2), realiza un redondeo a la centena. Los decimales los ignora.

4. Se ejecuta una serie de instrucciones SQL en una base de datos Oracle. A continuación se muestra, para cada instrucción, la respuesta que proporciona la base de datos.

```
SQL> insert into pr values ('aad3', 5.6, '2');
1 fila creada.
```

```
SQL> insert into pr values ('3founc5bayt', 67.54, 's');
Informe de error -
ORA-12899: el valor es demasiado grande para la columna "ALUMNO"."PR"."CAMP01" (real:
11, máximo: 10)
```

```
SQL> insert into pr values ('polk6hygtf', 5467.98, 'd');
Informe de error -
ORA-01438: valor mayor que el que permite la precisión especificada para esta columna
```

```
SQL> insert into pr values ('polk6hygtf', 6.154, 'e');
1 fila creada.
```

```
SQL> insert into pr values ('pol', 5467.98, 'dr');
Informe de error -
ORA-01438: valor mayor que el que permite la precisión especificada para esta columna
```

```
SQL> insert into pr values ('polk6hygtf', 6.154, 'dr');
Informe de error -
ORA-12899: el valor es demasiado grande para la columna "ALUMNO"."PR"."CAMP03" (real:
2, máximo: 1)
```

```
SQL> insert into pr values ('polrgt', 67.93, 'A');
Informe de error -
ORA-01438: valor mayor que el que permite la precisión especificada para esta columna
```

```
insert into pr values ('abcde', 2.93, 2);
Informe de error -
ORA-00001: restricción única (ALUMNO.SYS_C007345) violada
```

```
SQL> select * from pr;
CAMP01          CAMPO2 C
-----
aad3              5,6 2
polk6hygtf        6,154 e
```

Interpreta esas instrucciones y escribe el comando necesario para crear la tabla PR.

Solución:

```
create table PR(
    campo1 varchar2(10),
    campo2 number(4,3),
    campo3 char(1) primary key);
```

Ejercicios: crear tablas con relaciones (PK, FK y not null)

5. Se diseñan las siguientes tablas relacionadas:

PISO (**ref**, **dni_propietario**, fecha_compra)
 dni_propietario es FK, referencia a PROPIETARIO (piso)
 fecha de compra es un campo obligatorio

PROPIETARIO (**dni**)

a) Interpreta el significado de los campos y sus relaciones. Crea las tablas empleando el formato de restricción de columna para todas las restricciones.

Solución:

Hay que crear primero PROPIETARIO, ya que PISO la referencia:

```
create table propietario(
    dni varchar2(9) primary key);

create table piso(
    refer varchar2(5) primary key,
    dni_propietario varchar2(9) references propietario(dni),
    fecha_compra date not null);
```

b) Inserta los siguientes datos:

- El propietario con DNI 85000000J tiene el piso de referencia 4501X, comprado el 29-03-2009
- El propietario con DNI 46000000N tiene 3 pisos, con referencias 1001A, 2120B y 1002V, comprados todos el día 10-02-2014
- El piso de referencia 7575F es propiedad de 98000000H desde hoy (sysdate)

Solución:

```
insert into propietario values ('85000000J');
insert into propietario values ('46000000N');
insert into propietario values ('98000000H');
insert into piso values ('4501X', '85000000J', '29-03-2009');
insert into piso values ('1001A', '46000000N', '10-02-2014');
insert into piso values ('1002V', '46000000N', '10-02-2014');
insert into piso values ('2120B', '46000000N', '10-02-2014');
insert into piso values ('7575F', '98000000H', sysdate);
```

c) ¿Puede un piso pertenecer a dos propietarios?

Solución:

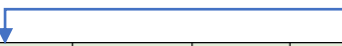
No es posible ya que solo hay un campo “dni_propietario” en la tabla PISO. Hay que recordar que el modelo de datos relacional no permite valores multivaluados en las columnas (más de un valor en una columna).

6. Se diseña la siguiente tabla:

PERSONA (**dni**, nombre, edad, **padre**)
padre es FK, referencia a PERSONA (dni)

a) Interpreta el significado de los campos y sus relaciones. Crea la tabla empleando el formato de restricción de columna para todas las restricciones.

Observa que hay una referencia (FK) a la misma tabla. Eso significa que solo podrán ponerse como padres en el campo “padre” personas que previamente estén dadas de alta en la tabla (campo “dni”).



PERSONA	<u>dni</u>	nombre	edad	padre

Hay que tener en cuenta que:

- Dni es un texto de 9 caracteres
- Nombre es un texto de 20 caracteres
- Edad es un número de 3 dígitos

Solución:

```
create table persona(
  dni varchar2(9) primary key,
  nombre varchar2(20),
  edad number(3),
  padre varchar2(9) references persona(dni));
```

b) Introduce los datos:

- Juan, con DNI 10000000A y 32 años
- Luis, con DNI 20000000A (54 años) y su hijo Luis con DNI 40000000A (18 años)
- Daniel con DNI 30000000A (73 años) y su hijo Alberto con DNI 50000000A (51 años) y su nieta Eva con DNI 60000000A (19 años).

Nota: para insertar una persona que no tenga padre, se puede poner en el campo padre el valor null, que es equivalente a vacío.

Solución:

```
insert into persona values ('10000000A','Juan',32, null);
insert into persona values ('20000000A','Luis', 54, null);
insert into persona values ('40000000A','Luis',18,'20000000A');
insert into persona values ('30000000A','Daniel',73,null);
insert into persona values ('50000000A','Alberto',51,'30000000A');
insert into persona values ('60000000A','Eva',16,'50000000A');
```

7. Se quiere una base de datos en la que se almacenen datos de trabajadores. Para cada trabajador interesa conocer su nombre, nif, apellidos, edad, teléfono y puesto.

Hay que tener en cuenta que un mismo trabajador puede tener varios teléfonos, pero no se conoce a priori cuántos (podría tener uno, dos, tres, cuatro... o ninguno).

- a) Crea dos tablas relacionadas: TRABAJADOR y TLFNO_TRABAJADOR. Elige los campos que debe tener cada una para cumplir los requisitos.

Considera que un teléfono solo puede ser de un trabajador (dos trabajadores no pueden tener el mismo teléfono). Recuerda que en el modelo relacional no puede haber campos multivaluados (una columna con más de un valor). Intenta que haya la menor información duplicada posible, para optimizar la base de datos.

Elige también el tipo de datos necesario para poder introducir los siguientes trabajadores, e introdúcelos en las tablas:

- Técnico de mantenimiento: Rubén Ruiz, de 33 años, con NIF 46000000W y teléfono 612000003.
- Conduccora: Ana Isabel Mateos, de 42 años, con NIF 35000000T y teléfonos 916780001 y 610500001.

Solución:

Las tablas serían:

```
TRABAJADOR(dni, nombre, apellidos, edad, puesto)
TLFNO_TRABAJADOR(telefono, dni)
FK: DNI referencia a TRABAJADOR(dni)
```

Con este diseño nos aseguramos que se pueden introducir tantos teléfonos para un trabajador como sea necesario: cada registro de la tabla TLFNO_TRABAJADOR será un teléfono.

Comandos:

```
create table trabajador(
dni varchar2(9) primary key,
nombre varchar2(20),
apellidos varchar2(40),
edad number(2),
puesto varchar2(30)
);
create table tlfno_trabajador(
telefono number(9) primary key,
dni varchar2(9),
foreign key (dni) references trabajador(dni)
);
insert into trabajador values('46000000W','Rubén','Ruiz',33,'Técnico de mantenimiento');
insert into tlfno_trabajador values (612000003, '46000000W');
insert into trabajador values('35000000T','Ana Isabel','Mateos',42,'Conduccora');
insert into tlfno_trabajador values (916780001, '35000000T');
insert into tlfno_trabajador values (610500001, '35000000T');
```

- b) ¿Qué cambios habría que hacer en las tablas para que dos empleados pudiesen tener el mismo teléfono? Borra las tablas y créalas de nuevo con la nueva condición.

Inserta a dos trabajadores que viven en el mismo domicilio y comparten teléfono fijo:

- Técnico de mantenimiento: Rubén Ruiz, de 33 años, con NIF 46000000W. Teléfono 916725899 y 612000003.
- Conduccora: Ana Isabel Mateos, de 42 años, con NIF 35000000T. Teléfono 916725899 y teléfono 610500001.

Solución:

Las tablas serían:

```
TRABAJADOR(dni, nombre, apellidos, edad, puesto)
TLFNO_TRABAJADOR(telefono, dni)
    FK: DNI referencia a TRABAJADOR(dni)
```

Al tener la tabla TLFNO_TRABAJADOR una clave primaria compuesta, podría repetirse el mismo teléfono en dos filas distintas, lo que permite asociar el mismo teléfono a dos trabajadores distintos.

Comandos:

```
drop table tlfno_trabajador;
drop table trabajador;
create table trabajador(
dni varchar2(9) primary key,
nombre varchar2(20),
apellidos varchar2(40),
edad number(2),
puesto varchar2(30)
);
create table tlfno_trabajador(
telefono number(9),
dni varchar2(9),
primary key (telefono,dni),
foreign key (dni) references trabajador(dni)
);
insert into trabajador values('46000000W','Rubén','Ruiz',33,'Técnico de mantenimiento');
insert into tlfno_trabajador values (612000003, '46000000W');
insert into tlfno_trabajador values (916725899, '46000000W');
insert into trabajador values('35000000T','Ana Isabel','Mateos',42,'Conduccora');
insert into tlfno_trabajador values (610500001, '35000000T');
insert into tlfno_trabajador values (916725899, '35000000T');
```

Ejercicios: crear tablas con restricciones

8. Se define la tabla:

MONTE(nombre, provincia, altura)

a) Crea la tabla, teniendo en cuenta:

- Sólo se podrán introducir montes más altos de 2000 metros.
- La altura es un campo obligatorio.

Crea todas las restricciones como restricciones de tabla.

b) Introduce los datos:

- Mulhacén 3479 metros. Granada.
- Aneto 3404 metros. Huesca

- Teide 3718 metros. Santa Cruz de Tenerife.
- Naranjo de Bulnes 2519 metros. Asturias

Solución:

```
create table monte(
nombre varchar2(30),
provincia varchar2(30),
altura number(4),
primary key (nombre),
check (altura>2000),
check (altura is not null));
```

Nota: se ha usado **check (altura is not null)** En lugar de definir el campo **altura number(4) not null** porque el enunciado especifica que todas las restricciones se definan con el formato de restricción de tabla. Not null no puede ponerse en ese formato, es necesario transformarlo en check. La restricción funciona igual.

```
insert into monte values('Mulhacén','Granada',3479);
insert into monte values('Aneto','Huesca',3404);
insert into monte values('Teide','Santa Cruz de Tenerife',3718);
insert into monte values('Naranjo de Bulnes','Asturias',2519);
```

c) Comprueba que al intentar insertar los siguientes datos, todos dan error.

- Mulhacén 3480 metros. Granada.
- Puig Major 1445 metros. Islas Baleares
- Almanzor. Ávila.

Solución:

```
insert into monte values('Mulhacén','Granada',3480); Da error por duplicar PK
insert into monte values('Puig Major','Islas Baleares',1445); Error por no cumplir el
check del campo altura mayor de 2000 (restricción de control violada)
insert into monte(nombre,provincia) values('Almanzor','Ávila'); Error por no cumplir
el check del campo altura not null
```

9. Se tiene la siguiente tabla definida:

EMPRESARIO (nif, empresa, nombre, mail)

a) **Antes de crear la tabla**, para los siguientes datos, marca cuales podrán insertarse y cuáles no (en el orden en que aparecen). Indica la causa.

Cuando no aparece un dato significa que no se conoce (no es un espacio en blanco).

	NIF	Empresa	Nombre	mail
1	85000000H	Gesco SL	Juan Colindres	director@gesco.es
2	44000000C	BotCO	Julia Martín	info@bot.com
3	13000000F	Tamar	Ana Palacios	informacion-tamar
4	77000000G	Tintor SA		info12345@gmail.com
5	75000000G	Next SA	Julia Martín	const-77@yahoo.es
6	99000000D	Gesco SL	Adrián Palomares	subdir@gesco.es
7	47000000S		Sergio Olivares	serol_89@goggle.com
8	62000000W	Alian	Jose María Lara	const-77@yahoo.es
9	54000000Q	DisCoTK	Agustín Moya	

Solución:

Fallarán las filas:

- 3 No tiene @ en el mail
- 6 La empresa está repetida
- 7 El nombre de empresa es obligatorio
- 8 El mail está repetido

b) Crea la tabla. Hay que tener en cuenta lo siguiente:

- No puede repetirse el nombre de empresa para dos empresarios.
- No puede repetirse el mail para dos empresarios.
- El mail debe contener el carácter @
- La empresa debe indicarse obligatoriamente para un empresario.

Crea todas las restricciones con el formato de restricción de columna.

Solución:

```
create table empresario(
    nif varchar2(9) primary key,
    empresa varchar2(30) not null unique,
    nombre varchar2(30),
    mail varchar2(30) unique check (mail like '%@%')
);
```

c) Inserta los datos del apartado a) comprobando si fallan los que habías señalado

Fallan los señalados en rojo:

```
insert into empresario values ('85000000H','Gesco SL','Juan
    Colindres','director@gesco.es');
insert into empresario values ('44000000C','BotCO','Julia Martín','info@bot.com');
insert into empresario values ('13000000F','Tamar','Ana Palacios','informacion-
    tamar');
insert into empresario values ('77000000G','Tintor SA',null,'info12345@gmail.com');
insert into empresario values ('75000000G','Next SA','Julia
    Martín','const_77@yahoo.es');
insert into empresario values ('99000000D','Gesco SL','Adrián
    Palomares','subdir@gesco.es');
insert into empresario values ('47000000S',null,'Sergio
    Olivares','serol_89@google.com');
insert into empresario values ('62000000W','Alian','Jose María
    Lara','const_77@yahoo.es');
insert into empresario values ('54000000Q','DisCoTK','Agustín Moya',null);
```

10. Se definen las siguientes tablas para crear una base de datos para una librería.

LIBRO (isbn, título, temática, posicion)

AUTOR (isbn, nombre)

FK: isbn referencia LIBRO(isbn)

a) En primer lugar, interpreta las tablas y, **sin crear las tablas**, responde a las preguntas:

¿Cuántos autores puede tener un libro?

Solución: Un libro puede tener 0, 1 o más de un autor.

- Si un "isbn" no aparece en la tabla AUTOR, no está guardado el autor de ese isbn (no se conoce o no se ha registrado).
- Si un isbn aparece en un registro de la tabla AUTOR significa que el "nombre" de ese registro es el autor del libro.
- Si un "isbn" aparece en varios registros de la tabla AUTOR, cada uno con distinto "nombre" (no pueden repetirse registros), todos esos nombres serán coautores del libro.

¿Puede un autor haber escrito más de un libro?

Solución: Un autor puede haber escrito más de un libro, aparecerá en la tabla AUTORES en varios registros el mismo "nombre" pero con distinto "isbn".

¿Pueden estar libros escritos por el mismo autor en estantes distintos?

Solución: El estante en el que se guarda un libro está en el campo "posición" y depende del "isbn". Es decir, cada "isbn" tendrá un estante, por lo que el autor no influye en el estante: puede haber libros escritos por el mismo autor que estén en estantes distintos.

b) Crea las tablas teniendo en cuenta:

- isbn es varchar2(10)
- titulo y nombre son varchar2(30)
- temática es varchar2(20)
- posición es varchar2(5)
- Las temáticas posibles son 'terror', 'aventuras' e 'infantil'.
- El título del libro es un campo obligatorio.
- La posición se refiere a la estantería donde está el libro, y la forma de nombrarla es 'EST-x', siendo x un número o letra identificativa (1 carácter). Por ejemplo: EST-1, EST-2, EST-a, EST-H, etc

Las restricciones de tipo "check" y "foreign key" deben crearse con el formato de restricción de tabla. El resto pueden crearse como se prefiera.

Solución:

```
create table libro(  
    isbn varchar2(10) primary key,  
    titulo varchar2(30) not null,  
    tematica varchar2(20),  
    posicion varchar2(5),  
    check (tematica in('terror','aventuras','infantil')),  
    check (posicion like 'EST-_%')  
);  
create table autor(  
    isbn varchar2(10),  
    nombre varchar2(30),  
    primary key (isbn, nombre),  
    foreign key (isbn) references libro (isbn) );
```

- c) Una vez creadas las tablas, comprueba el funcionamiento de las restricciones intentando insertar datos que no las cumplan. Debes construir un comando distinto que falle para cada restricción.

Solución:

```
insert into libro values ('123456','Drácula','misterio', 'EST-1');
Informe de error -
ORA-02290: restricción de control (...) violada

insert into libro (isbn,tematica,posicion) values ('123456','terror','EST-1');
Informe de error -
ORA-01400: no se puede realizar una inserción NULL en (.."LIBRO"."TITULO")

insert into libro values('123456','Drácula','misterio', 'POS1');
Informe de error -
ORA-02290: restricción de control (...) violada

insert into autor values('777777','Bram Stoker');
Informe de error -
ORA-02291: restricción de integridad (...) violada - clave principal no encontrada
```

Ejercicios: crear tablas con nombre de restricción

11. Se quiere implementar una base de datos para almacenar los extintores que tienen vagones de tren.

Se tienen una serie de vagones. Un vagón se identifica por el código del tren al que pertenece y el orden que ocupa dentro del él. Se conoce además su modelo y año de fabricación.

Se tienen una serie de extintores. Cada extintor tiene un número de referencia unívoco que lo identifica, e interesa conocer en qué vagón concreto está ubicado.

En un mismo vagón puede haber 0 extintores, 1 extintor, o más de 1.

Este es el diseño de las tablas:

VAGÓN (codigo_tren, orden, modelo, año_fab)

EXTINTOR (num_ref, codigo_tren, orden)

FK: (codigo_tren,orden) referencia a VAGÓN (codigo_tren, orden)

- a) Crea las tablas teniendo en cuenta:

- El código de tren es un texto de 10 caracteres
- El orden del vagón es un número de 2 dígitos. Debe estar comprendido entre el 1 y el 20.
- El modelo del vagón es un texto de 10 caracteres
- El año de fabricación del vagón es un número de 4 dígitos. Siempre debe ser posterior al 2015.

Crea todas las restricciones con nombre de restricción (a tu elección).

Solución:

```
create table vagon(
  cod_tren varchar2(10),
  orden number(2) constraint orden_rango_1_20 check (orden between 1 and 20),
  modelo varchar2(10),
```

```

año_fab number(4) constraint año_mayor_2015 check (año_fab > 2015),
constraint pk_vagon primary key (cod_tren,orden)
);
create table extintor(
num_ref number(4) constraint pk_extintor primary key,
cod_tren varchar2(10),
orden number(2),
constraint fk_extintor_vagon foreign key (cod_tren, orden) references vagon
(cod_tren,orden)
);

```

b) Revisa las vistas de las constraints y comprueba que se han creado correctamente.

Solución:

```
select * from user_constraints where table_name='VAGON';
```

OWNER	CONSTRAINT_NAME	CONSTRAINT_TYPE	TABLE_NAME	SEARCH_CONDITION
ALUMNO	ORDEN RANGO 1 20	C	VAGON	orden between 1 and 20
ALUMNO	AÑO MAYOR 2015	C	VAGON	año fab > 2015
ALUMNO	PK VAGON	P	VAGON	(null)

```
select * from user_cons_columns where table_name='VAGON';
```

OWNER	CONSTRAINT_NAME	TABLE_NAME	COLUMN_NAME	POSITION
ALUMNO	ORDEN RANGO 1 20	VAGON	ORDEN	(null)
ALUMNO	AÑO MAYOR 2015	VAGON	AÑO FAB	(null)
ALUMNO	PK VAGON	VAGON	COD TREN	1
ALUMNO	PK VAGON	VAGON	ORDEN	2

```
select * from user_constraints where table_name='EXTINTOR';
```

OWNER	CONSTRAINT_NAME	CONSTRAINT_TYPE	TABLE_NAME	S
ALUMNO	PK EXTINTOR	P	EXTINTOR	(
ALUMNO	FK EXTINTOR VAGON	R	EXTINTOR	(

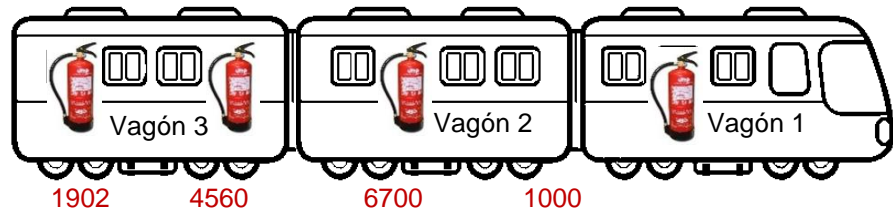
```
select * from user_cons_columns where table_name='EXTINTOR';
```

OWNER	CONSTRAINT_NAME	TABLE_NAME	COLUMN_NAME	POSITION
ALUMNO	PK EXTINTOR	EXTINTOR	NUM REF	1
ALUMNO	FK EXTINTOR VAGON	EXTINTOR	COD TREN	1
ALUMNO	FK EXTINTOR VAGON	EXTINTOR	ORDEN	2

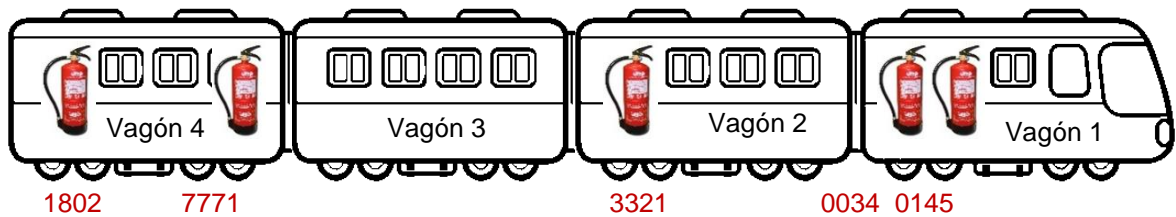
c) Inserta los datos del siguiente esquema. Ten en cuenta:

- Los números en rojo se refieren a los extintores.
- Los vagones 1 y 3 del tren 120A son del modelo 'D-2866 LUE'. El resto de vagones son del modelo 'H-333 DUE'.
- Todos los vagones se fabricaron en 2020, excepto las cabeceras (vagones de orden 1), que se fabricaron en 2018.

Tren código= 120A



Tren código= 033B



Solución:

```
insert into vagon values ('120A',3,'D-2866 LUE',2020);
insert into vagon values ('120A',2,'H-333 DUE',2020);
insert into vagon values ('120A',1,'D-2866 LUE',2018);
insert into vagon values ('033B',4,'D-2866 LUE',2020);
insert into vagon values ('033B',3,'D-2866 LUE',2020);
insert into vagon values ('033B',2,'D-2866 LUE',2020);
insert into vagon values ('033B',1,'D-2866 LUE',2018);
insert into extintor values (1902,'120A',3);
insert into extintor values (4560,'120A',3);
insert into extintor values (6700,'120A',2);
insert into extintor values (1000,'120A',1);
insert into extintor values (1802,'033B',4);
insert into extintor values (7771,'033B',4);
insert into extintor values (3321,'033B',2);
insert into extintor values (0034,'033B',1);
```

- d) Intenta insertar los siguientes datos, que deben fallar. Comprueba el mensaje de error que da el fallo y busca en él el nombre de la restricción que se ha violado.

Vagón: tren '120A', orden 21, modelo 'D-2866 LUE', año de fabricación 2020

Vagón: tren '120A', orden 4, modelo 'D-2866 LUE', año de fabricación 2015

Vagón: tren '120A', orden 3, modelo 'D-2866 LUE', año de fabricación 2020

Extintor: número de referencia 1902. Se encuentra en el vagón del tren '120A', orden 3

Extintor: número de referencia 1903. Se encuentra en el vagón del tren '120A', orden 4

Solución:

```
insert into vagon values ('120A',21,'D-2866 LUE',2020);
```

Informe de error -

ORA-02290: restricción de control (ALUMNO.ORDEN_RANGO_1_20) violada

```
insert into vagon values ('120A',4,'D-2866 LUE',2015);
Informe de error -
ORA-02290: restricción de control (ALUMNO.AÑO_MAYOR_2015) violada

insert into vagon values ('120A',3,'D-2866 LUE',2020);
Informe de error -
ORA-00001: restricción única (ALUMNO.PK_VAGON) violada

insert into extintor values (1902,'120A',3);
Informe de error -
ORA-00001: restricción única (ALUMNO.PK_EXTINTOR) violada

insert into extintor values (1903,'120A',4);
Informe de error -
ORA-02291: restricción de integridad (ALUMNO.FK_EXTINTOR_VAGON) violada - clave
principal no encontrada
```

Ejercicios: crear tablas con nombre de restricción y default

12. Se quiere almacenar información sobre los productos vendidos por una empresa de tornillos.

La empresa dispone de distintos vendedores, que se almacenan en la tabla Vendedor. Cada uno de los vendedores vende en una tienda en concreto.

Las tiendas se guardan en la tabla Tienda.

Los productos que se venden son tornillos, que se guardan en la tabla Tornillo.

Las ventas que hacen los vendedores se guardan en la tabla Venta. En esa tabla se refleja el vendedor, el producto vendido y en qué cantidad, y también la fecha de la venta.

Este es el diseño de la base de datos:

Tornillo (codigo, diametro)

FK: vendedor referencia vendedor(dni) borrado cascada

Vendedor (dni, calle, ciudad)

FK: calle, ciudad referencia a Ciudad(calle, ciudad)

Tienda (calle, ciudad, comunidad)

Venta (vendedor, cod_tornillo, diametro, fecha, cantidad)

FK: vendedor referencia vendedor(dni)

FK: cod_tornillo, diametro referencia a tornillo (código, diámetro)

- a) En primer lugar, interpreta las tablas y, **sin crear las tablas**, responde a las preguntas:

¿Pueden existir distintos diámetros de un mismo código de tornillo?

Solución: En la tabla Tornillo la clave primaria es una clave compuesta por “código” y “diámetro”, lo que significa que se puede repetir el “código” si cambia el diámetro: el mismo tipo de tornillo puede estar en diámetros distintos.

¿Puede un vendedor vender distintos tornillos el mismo día?

Solución: Las ventas se reflejan en la tabla Venta. En esa tabla se refleja qué vendedor ha hecho la venta, qué tornillo ha vendido, en que fecha y que cantidad. Como la clave primaria incluye la fecha de venta, significa que puede repetirse esa fecha en varios registros, siempre y cuando no se repitan el resto de campos de la clave primaria, por lo que sí puede vender un vendedor distintos tipos de tornillos el mismo día.

¿Pueden tener dos tiendas que están en distinta ciudad el mismo nombre de calle?

Solución: Una tienda se identifica por “calle” y “ciudad”, por lo que puede haber dos tiendas que tengan el mismo nombre de calle en distintas ciudades, lo que no podría repetirse en la tabla Tienda es la combinación de “calle” y “ciudad”.

¿por qué se incluye en la tabla Vendedor los campos “calle” y “ciudad” ?

Solución: Como cada vendedor vende en una única tienda, en la tabla Vendedor queda reflejada la tienda en la que vende (que se identifica por calle y ciudad).

¿Puede haber más de un vendedor que venda en la misma tienda?

Solución: En la tabla Vendedor se indica la tienda en la que vende (identificada por calle y ciudad). La clave de esa tabla es “dni”, por lo que podrían repetirse las tiendas (“calle” y “ciudad”) para distintos “dni”: distintos vendedores pueden vender en la misma tienda.

b) Crea las tablas. Debes incluir, aparte de las restricciones de clave primaria y externa, las siguientes:

- Los códigos de producto que vende la empresa son: AA1, AA2, AA3, AA4 y AA5. No puede haber ninguno más.
- El diámetro de un tornillo debe ser menor de 90.
- La cantidad de producto vendidos debe ser 5 como mínimo y 50 como máximo.
- La dirección de la venta debe empezar por ‘C’ o por ‘P’ obligatoriamente (corresponde a Calle y Plaza, respectivamente)
- La comunidad de una tienda es un campo obligatorio.
- Solo puede haber una tienda por comunidad autónoma: el campo “comunidad” no se puede repetir en la tabla Tienda.
- Si no se indica la fecha en una venta, la fecha por defecto será el día de hoy (sysdate).
- Si no se indica la cantidad en una venta, la cantidad por defecto será 25.

Para determinar el tipo de campo en cada caso, puedes consultar el apartado b) y revisar los datos que se van a insertar.

Las restricciones deben crearse con formato de columna siempre que sea posible.

Todas las restricciones de tipo “check” y “unique” deben tener nombre de restricción (a tu elección).

Solución:

```
create table tornillo(  
codigo varchar2(3) constraint codigo_lista_AAx check (codigo in('AA1', 'AA2',  
    'AA3','AA4','AA5')),  
diametro number(2) constraint diametro_menor_90 check (diametro < 90),  
primary key(codigo,diametro)  
);
```



```

create table tienda(
calle varchar2(30) constraint calle_C_P check (calle like 'C%' or calle like 'P%'),
ciudad varchar2(30),
comunidad varchar2(30) constraint comunidad_no_repeticion unique not null,
primary key(calle,ciudad)
);
create table vendedor(
dni varchar2(9) primary key,
calle varchar2(30),
ciudad varchar2(30),
foreign key (calle,ciudad) references tienda(calle,ciudad)
);
create table venta(
vendedor varchar2(9) references vendedor(dni),
cod_tornillo varchar2(3),
diametro number(2),
fecha date default sysdate,
cantidad number(2) default 25 constraint cantidad_rango_5_50 check (cantidad between
5 and 50),
primary key (vendedor,cod_tornillo,diametro,fecha),
foreign key(cod_tornillo,diametro) references tornillo(codigo,diametro));

```

c) Revisa las vistas de las constraints y comprueba que se han creado correctamente.

Solución:

```

select * from user_constraints where table_name='TORNILLO';
select * from user_cons_columns where table_name='TORNILLO';
select * from user_constraints where table_name='TIENDA';
select * from user_cons_columns where table_name='TIENDA';
select * from user_constraints where table_name='VENDEDOR';
select * from user_cons_columns where table_name='VENDEDOR';
select * from user_constraints where table_name='VENTA';
select * from user_cons_columns where table_name='VENTA';

```

d) Inserta las siguientes ventas. Ten en cuenta que para hacer inserciones en la tabla Venta previamente tienes que insertar vendedores, tiendas y productos.

DNI del vendedor	Dirección de la venta	Ciudad de la venta	Comunidad de la venta	Producto vendido (código y diámetro en mm)	Cantidad de productos vendidos	Fecha venta
10000000F	C. Jorge Juan	Madrid	Madrid	AA1, 20	30	12-2-21
50000000U	C. Pradillo	Oviedo	Asturias	AA5, 15	10	12-2-21
90000000T	C. Jorge Juan	Granada	Andalucía	AA2, 20	12	13-5-21
70000000Y	Plaza Mayor	Salamanca	Castilla León	AA3, 10	8	20-5-21
10000000F	C. Jorge Juan	Madrid	Madrid	AA2, 10	5	18-4-21
10000000F	C. Jorge Juan	Madrid	Madrid	AA2, 20	7	20-4-21

Solución:

```

insert into tienda values ('C. Jorge Juan', 'Madrid','Madrid');
insert into tienda values ('C. Pradillo','Oviedo','Asturias');

```

```
insert into tienda values ('C. Jorge Juan','Granada','Andalucía');
insert into tienda values ('Plaza Mayor','Salamanca','Castilla León');
insert into tornillo values ('AA1',20);
insert into tornillo values ('AA2',10);
insert into tornillo values ('AA2',20);
insert into tornillo values ('AA3',10);
insert into tornillo values ('AA5',15);
insert into vendedor values ('10000000F','C. Jorge Juan', 'Madrid');
insert into vendedor values ('50000000U','C. Pradillo', 'Oviedo');
insert into vendedor values ('90000000T','C. Jorge Juan', 'Granada');
insert into vendedor values ('70000000Y','Plaza Mayor', 'Salamanca');
insert into venta values('10000000F','AA1',20,'12-02-21',30);
insert into venta values('50000000U','AA5',15,'12-02-21',10);
insert into venta values ('90000000T','AA2',20,'13-02-21',12);
insert into venta values('70000000Y','AA3',10,'20-05-21',8);
insert into venta values('10000000F','AA2',10,'18-04-21',5);
insert into venta values('10000000F','AA2',20,'20-04-21',7);
```

- e) Inserta la siguiente venta, en la que no se indica cantidad ni fecha. Comprueba que funcionan correctamente los valores por defecto.

DNI del vendedor	Dirección de la venta	Ciudad de la venta	Comunidad de la venta	Producto vendido (código y diámetro en mm)	Cantidad de productos vendidos	Fecha venta
10000000F	C. Jorge Juan	Madrid	Madrid	AA1, 20		

Solución:

```
insert into venta(vendedor, cod_tornillo, diametro) values ('10000000F','AA1',20);
Al hacer select * from venta aparece:
```

VENDEDOR	COD_TORNILLO	DIAMETRO	FECHA	CANTIDAD
10000000F	AA1		20-12-21	30
50000000U	AA5		15-12-21	10
90000000T	AA2		20-13-21	12
70000000Y	AA3		10-20-21	8
10000000F	AA2		10-18-21	5
10000000F	AA2		20-20-21	7
10000000F	AA1		20-13-08-21	25

- f) Intenta insertar los siguientes datos. Si las tablas están creadas correctamente, todos ellos deben fallar. Indica la causa del fallo en cada caso y revisa el mensaje de error que aparece.

- El vendedor 10000000F vende 7 tornillos 'AA1' de diámetro 20 el 12-2-21
- El vendedor 50000000U vende 10 tornillos 'AA3' con diámetro 20 el 12-12-21
- El vendedor 50000000U vende 2 tornillos 'AA7' con diámetro 20 el 12-11-21
- El vendedor 90000000T vende 54 tornillos 'AA1' con diámetro 20 el 1-5-21
- Se da de alta una tienda en la "Plaza Mayor" de Madrid, comunidad de Madrid.
- Se da de alta una tienda en la "Plaza Mayor" de la ciudad Riofrio sin indicar comunidad (null).
- Se da de alta el tornillo 'AA6' con diámetro 95.

Solución:

```
insert into venta values('10000000F','AA1',20,'12-02-21',7);
```

Informe de error -

ORA-00001: restricción única (ALUMNO.SYS_C007367) violada (se duplica la PK)

```
insert into venta values('50000000U','AA3',20,'12-12-21',10);
Informe de error -
ORA-02291: restricción de integridad (ALUMNO.SYS_C007395) violada - clave principal
no encontrada (no encuentra la PK referenciada: no existe el tornillo AA3 de diam.20)

insert into venta values('50000000U','AA7',20,'12-11-21',2);
Informe de error -
ORA-02290: restricción de control (ALUMNO.CANTIDAD_RANGO_5_50) violada

insert into venta values ('90000000T','AA1',20,'01-05-21',54);
Informe de error -
ORA-02290: restricción de control (ALUMNO.CANTIDAD_RANGO_5_50) violada

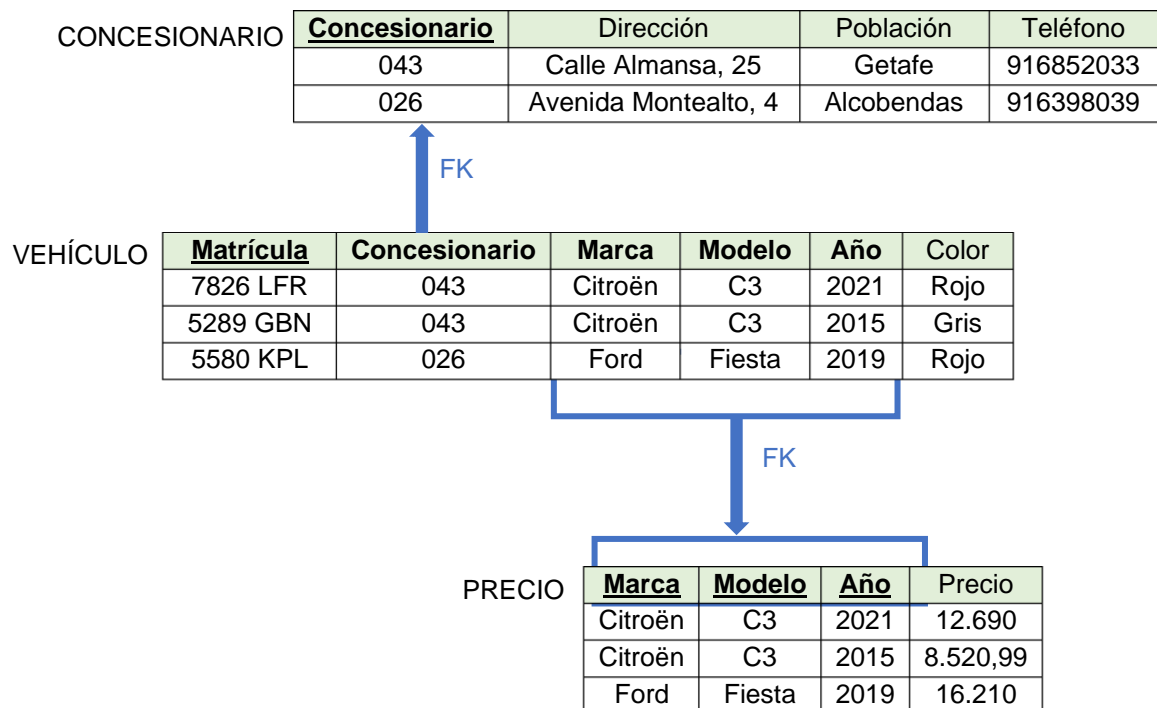
insert into tienda values ('Plaza Mayor','Madrid','Madrid');
Informe de error -
ORA-00001: restricción única (ALUMNO.COMUNIDAD_NO_REPETICION) violada (ya está la
comunidad de Madrid)

insert into tienda(calle,ciudad) values ('Plaza Mayor','Riofrio');
Informe de error -
ORA-01400: no se puede realizar una inserción NULL en ("ALUMNO"."TIENDA"."COMUNIDAD")

insert into tornillo values ('AA6',95);
Informe de error -
ORA-02290: restricción de control (ALUMNO.DIAMETRO_MENOR_90) violada
```

Ejercicios: crear tablas. Renombrar, borrado y truncado.

13. Se tiene una base de datos de una empresa comercializadora de vehículos, compuesta por tres tablas. Analiza las tablas, sus claves y las relaciones entre ellas.



a) Crea las tablas teniendo en cuenta:

- Selecciona los tipo de datos teniendo en cuenta los datos de las tablas, y las posibles inserciones futuras.
- La matrícula debe tener un espacio intercalado.
- Si no se indica año, se considerará 2021.
- El precio es un número que debe admitir 6 dígitos enteros y 2 decimales.
- El precio debe ser mayor de 5.000
- El color debe solo puede ser Rojo, Azul, Gris, Blanco y Negro.

Crea las restricciones con formato de columna siempre que sea posible. Las de tipo “foreign key” deben tener nombre de restricción.

Solución:

```
create table concesionario(
    concesionario number(3) primary key,
    direccion varchar2(50),
    población varchar2(30),
    telefono number(9));
create table precio(
    marca varchar2(20),
    modelo varchar2(20),
    año number(4) default 2021,
    precio number(8,2) check (precio > 5000),
    primary key(marca,modelo,año));
create table vehiculo(
    matricula varchar2(8) primary key check (matricula like '% %'),
    concesionario number(3) constraint fk_vehiculo_concesionario references
        concesionario(concesionario),
    marca varchar2(20),
    modelo varchar2(20),
    año number(4),
    color varchar2(20) check (color in ('Rojo','Azul','Gris','Blanco','Negro')),
    constraint fk_vehiculo_precio foreign key (marca,modelo,año) references
        precio(marca,modelo,año));
```

b) Inserta los datos que aparecen en las tablas.

El formato de los números decimales en la base de datos por defecto se introduce con un punto para la separación de los decimales y sin punto de los miles (es configurable). Por ejemplo 1.525,99 se introduciría como 1525.99.

Solución:

```
insert into concesionario values(43,'Calle Almansa, 25','Getafe',916852033);
insert into concesionario values(26,'Avenida Montealto, 4','Alcobendas',916398039);
insert into precio values('Citroën','C3',2021,12690);
insert into precio values('Citroën','C3',2015,8520.99);
insert into precio values('Ford','Fiesta',2019,16210);
insert into vehiculo values('7826 LFR',43,'Citroën','C3',2021,'Rojo');
insert into vehiculo values('5289 GBN',43,'Citroën','C3',2015,'Gris');
insert into vehiculo values('5580 KPL',26,'Ford','Fiesta',2019,'Rojo');
```

c) Intenta truncar la tabla “concesionario” reservando el espacio y explica qué ocurre.

Solución:

Truncate table concesionario reuse storage;

No se puede realizar porque hay referencias a esa tabla (la tabla "vehiculo" la está referenciando).

Informe de error -

ORA-02266: claves únicas/primarias en la tabla referidas por claves ajenas activadas

02266. 00000 - "unique/primary keys in table referenced by enabled foreign keys"

*Cause: An attempt was made to truncate a table with unique or primary keys referenced by foreign keys enabled in another table. Other operations not allowed are dropping/truncating a partition of a partitioned table or an ALTER TABLE EXCHANGE PARTITION.

*Action: Before performing the above operations the table, disable the foreign key constraints in other tables. You can see what constraints are referencing a table by issuing the following command: SELECT * FROM USER_CONSTRAINTS WHERE TABLE_NAME = "tabnam";

d) Cambia el nombre a la tabla "precio". El nuevo nombre será "precio_vehiculo".

Solución:

Rename precio to precio_vehiculo;

e) La tabla "vehiculo" referencia a la tabla "precio_vehiculo". Comprueba en las vistas de las constraints que la referencia entre ambas tablas se ha actualizado correctamente con el nuevo nombre.

Solución:

select * from user_constraints where table_name='VEHICULO';

OWNER	CONSTRAINT_NAME	CONSTRAINT_TYPE	TABLE_NAME	SEARCH_CONDITION	SEARCH_CONDITION	R_OWNER	R_CONSTRAINT_NAME	DELETE_RULE
ALUMNO	FK_VEHICULO_CONCESIONARIO	R	VEHICULO	(null)	(null)	ALUMNO	SYS_C007396	NO ACTION
ALUMNO	FK_VEHICULO_PRECIO	R	VEHICULO	(null)	(null)	ALUMNO	SYS_C007398	NO ACTION
ALUMNO	SYS_C007399	C	VEHICULO	matric...	matri...	(null)	(null)	(null)
ALUMNO	SYS_C007400	C	VEHICULO	color ...	color...	(null)	(null)	(null)
ALUMNO	SYS_C007401	P	VEHICULO	(null)	(null)	(null)	(null)	(null)

Como puede verse, la FK de VEHICULO tiene el nombre FK_VEHICULO_PRECIO y referencia a la PK SYS_C007398. Si comprobamos las restricciones de la tabla PRECIO_VEHICULO podemos comprobar que la restricción SYS_C007398 coincide con la clave primaria de esa tabla.

select * from user_constraints where table_name='PRECIO_VEHICULO';

OWNER	CONSTRAINT_NAME	CONSTRAINT_TYPE	TABLE_NAME	SEARCH_CONDITION
ALUMNO	SYS_C007397	C	PRECIO_VEHICULO	precio > 5000
ALUMNO	SYS_C007398	P	PRECIO_VEHICULO	(null)

Por último, comprobamos que la restricción SYS_C007398 corresponde a los campos "marca", "modelo" y "año":

select * from user_cons_columns where table_name='PRECIO_VEHICULO';

OWNER	CONSTRAINT_NAME	TABLE_NAME	COLUMN_NAME	POSITION
ALUMNO	SYS_C007397	PRECIO_VEHICULO	PRECIO	(null)
ALUMNO	SYS_C007398	PRECIO_VEHICULO	MARCA	1
ALUMNO	SYS_C007398	PRECIO_VEHICULO	MODELO	2
ALUMNO	SYS_C007398	PRECIO_VEHICULO	AÑO	3

- f) Utiliza el siguiente comando para borrar la tabla “concesionario”:

```
drop table concesionario;
```

Explica qué ocurre y qué solución habría si se quisiera borrar esa tabla a toda costa.

Solución:

```
drop table concesionario cascade constraints;  
Se borraría también la FK en la tabla “vehículo”.
```