# Ejercicios Resueltos de SQL

Borja Sotomayor 1 de junio de 2002

*Q*c Borja Sotomayor, 2002

Este documento ha sido compuesto ´ıntegramente en LATEX.

Para producir el documento se ha utilizado pdfTEX-1.0a de la distribuci´on MiKTEX.

<http://www.miktex.org/>

Se han utilizado los paquetes amssymb, babel, color, fancybox, fancyvrb, graphicx, hyperref, ifthen, inputenc, marvosym, scrbook y scrpage (KOMA–Script), titlesec, titletoc, pdfcrypt, y url.

Los diagramas han sido elaboradas con Microsoft@ Visio@ 2002.

<http://www.microsoft.com/office/visio/>

## ´Indice general

1. [**La Tienda de Inform´atica**](#_bookmark0) **1**
2. [**Empleados**](#_bookmark1) **7**
3. [**Los Almacenes**](#_bookmark2) **13**
4. [**Pel´ıculas y Salas**](#_bookmark3) **17**
5. [**Los Directores**](#_bookmark4) **19**
6. [**Piezas y Proveedores**](#_bookmark5) **23**
7. [**Los Cient´ıficos**](#_bookmark6) **27**
8. [**Los Grandes Almacenes**](#_bookmark7) **29**
9. [**Los Investigadores**](#_bookmark8) **33**

# 1

## La Tienda de Inform´atica

FABRICANTES

|  |  |  |
| --- | --- | --- |
| **PK** | **Codigo** | **int identity** |
|  | **Nombre** | **nvarchar(100)** |

|  |  |  |
| --- | --- | --- |
| ARTICULOS | | |
| **PK** | **Codigo** | **int identity** |
| **FK1** | **Nombre Precio Fabricante** | **nvarchar(100) int**  **int** |

* 1. Obtener los nombres de los productos de la tienda.

SELECT Nombre FROM ARTICULOS

* 1. Obtener los nombres y los precios de los productos de la tienda.

SELECT Nombre, Precio FROM ARTICULOS

* 1. Obtener el nombre de los productos cuyo precio sea menor o igual a 200 ¤.

SELECT Nombre FROM ARTICULOS WHERE Precio > 200

* 1. Obtener todos los datos de los art´ıculos cuyo precio est´e entre los 60 ¤ y los 120 ¤

(ambas cantidades incluidas).

/\* Con AND \*/

SELECT \* FROM ARTICULOS

WHERE Precio >= 60 AND Precio <= 120

/\* Con BETWEEN \*/ SELECT \* FROM ARTICULOS

WHERE Precio BETWEEN 60 AND 120

*1 La Tienda de Informa´tica*

* 1. Obtener el nombre y el precio en pesetas (es decir, el precio en euros multiplicado por 166’386)

/\* Sin AS \*/

SELECT Nombre, Precio \* 166.386 FROM ARTICULOS

/\* Con AS \*/

SELECT Nombre, Precio \* 166.386 AS PrecioPtas FROM ARTICULOS

* 1. Seleccionar el precio medio de todos los productos.

SELECT AVG(Precio) FROM ARTICULOS

* 1. Obtener el precio medio de los art´ıculos cuyo c´odigo de fabricante sea 2.

SELECT AVG(Precio) FROM ARTICULOS WHERE Fabricante=2

* 1. Obtener el numero de art´ıculos cuyo precio sea mayor o igual a 180 ¤.

SELECT COUNT(\*) FROM ARTICULOS WHERE Precio >= 180

* 1. Obtener el nombre y precio de los art´ıculos cuyo precio sea mayor o igual a 180 ¤

y ordenarlos descendentemente por precio, y luego ascendentemente por nombre.

SELECT Nombre, Precio FROM ARTICULOS WHERE Precio >= 180

ORDER BY Precio DESC, Nombre

* 1. Obtener un listado completo de art´ıculos, incluyendo por cada articulo los datos del articulo y de su fabricante.

/\* Sin INNER JOIN \*/

SELECT \* FROM ARTICULOS, FABRICANTES

WHERE ARTICULOS.Fabricante = FABRICANTES.Codigo

/\* Con INNER JOIN \*/ SELECT \*

FROM ARTICULOS INNER JOIN FABRICANTES

ON ARTICULOS.Fabricante = FABRICANTES.Codigo

* 1. Obtener un listado de art´ıculos, incluyendo el nombre del art´ıculo, su precio, y el nombre de su fabricante.

*1 La Tienda de Inform´atica*

/\* Sin INNER JOIN \*/

SELECT ARTICULOS.Nombre, Precio, FABRICANTES.Nombre FROM ARTICULOS, FABRICANTES

WHERE ARTICULOS.Fabricante = FABRICANTES.Codigo

/\* Con INNER JOIN \*/

SELECT ARTICULOS.Nombre, Precio, FABRICANTES.Nombre FROM ARTICULOS INNER JOIN FABRICANTES

ON ARTICULOS.Fabricante = FABRICANTES.Codigo

* 1. Obtener el precio medio de los productos de cada fabricante, mostrando solo los c´odigos de fabricante.

SELECT AVG(Precio), Fabricante FROM ARTICULOS GROUP BY Fabricante

* 1. Obtener el precio medio de los productos de cada fabricante, mostrando el nombre del fabricante.

/\* Sin INNER JOIN \*/

SELECT AVG(Precio), FABRICANTES.Nombre FROM ARTICULOS, FABRICANTES

WHERE ARTICULOS.Fabricante = FABRICANTES.Codigo GROUP BY FABRICANTES.Nombre

/\* Con INNER JOIN \*/

SELECT AVG(Precio), FABRICANTES.Nombre FROM ARTICULOS INNER JOIN FABRICANTES

ON ARTICULOS.Fabricante = FABRICANTES.Codigo GROUP BY FABRICANTES.Nombre

* 1. Obtener los nombres de los fabricantes que ofrezcan productos cuyo precio medio sea mayor o igual a 150 ¤.

/\* Sin INNER JOIN \*/

SELECT AVG(Precio), FABRICANTES.Nombre FROM ARTICULOS, FABRICANTES

WHERE ARTICULOS.Fabricante = FABRICANTES.Codigo GROUP BY FABRICANTES.Nombre

HAVING AVG(Precio) >= 150

/\* Con INNER JOIN \*/

SELECT AVG(Precio), FABRICANTES.Nombre FROM ARTICULOS INNER JOIN FABRICANTES

ON ARTICULOS.Fabricante = FABRICANTES.Codigo GROUP BY FABRICANTES.Nombre

HAVING AVG(Precio) >= 150

*1 La Tienda de Informa´tica*

* 1. Obtener el nombre y precio del art´ıculo m´as barato.

SELECT Nombre, Precio FROM ARTICULOS

WHERE Precio = (SELECT MIN(Precio) FROM ARTICULOS)

* 1. Obtener una lista con el nombre y precio de los art´ıculos m´as caros de cada pro- veedor (incluyendo el nombre del proveedor).

/\* Sin INNER JOIN \*/

SELECT A.Nombre, A.Precio, F.Nombre FROM ARTICULOS A, FABRICANTES F

WHERE A.Fabricante = F.Codigo AND A.Precio =

(

SELECT MAX(A.Precio) FROM ARTICULOS A

WHERE A.Fabricante = F.Codigo

)

/\* Con INNER JOIN \*/

SELECT A.Nombre, A.Precio, F.Nombre

FROM ARTICULOS A INNER JOIN FABRICANTES F

ON A.Fabricante = F.Codigo AND A.Precio =

(

SELECT MAX(A.Precio) FROM ARTICULOS A

WHERE A.Fabricante = F.Codigo

)

* 1. An˜adir un nuevo producto: Altavoces de 70 ¤ (del fabricante 2)

INSERT INTO ARTICULOS( Nombre , Precio , Fabricante) VALUES ( ’Altavoces’ , 70 , 2 )

* 1. Cambiar el nombre del producto 8 a ’Impresora Laser’

UPDATE ARTICULOS

SET Nombre = ’Impresora Laser’ WHERE Codigo = 8

* 1. Aplicar un descuento del 10 % (multiplicar el precio por 0’9) a todos los productos.

UPDATE ARTICULOS

SET Precio = Precio \* 0.9

*1 La Tienda de Inform´atica*

* 1. Aplicar un descuento de 10 ¤ a todos los productos cuyo precio sea mayor o igual a 120 ¤.

UPDATE ARTICULOS

SET Precio = Precio - 10 WHERE Precio >= 120

# 2

## Empleados

DEPARTAMENTOS

**PK**

**Codigo**

**int**

**Nombre Presupuesto**

**nvarchar(100) int**

|  |  |  |
| --- | --- | --- |
| EMPLEADOS | | |
| **PK** | **DNI** | **varchar(8)** |
| **FK1** | **Nombre Apellidos Departamento** | **nvarchar(100) nvarchar(255) int** |

* 1. Obtener los apellidos de los empleados.

SELECT Apellidos FROM EMPLEADOS

* 1. Obtener los apellidos de los empleados sin repeticiones.

SELECT DISTINCT Apellidos FROM EMPLEADOS

* 1. Obtener todos los datos de los empleados que se apellidan ’Lo´pez’.

SELECT \* FROM EMPLEADOS WHERE Apellidos = ’L´opez’

* 1. Obtener todos los datos de los empleados que se apellidan ’Lo´pez’ y los que se apellidan ’P´erez’.

*2 Empleados*

/\* Con OR \*/

SELECT \* FROM EMPLEADOS

WHERE Apellidos = ’L´opez’ OR Apellidos = ’P´erez’

/\* Con IN \*/

SELECT \* FROM EMPLEADOS

WHERE Apellidos IN (’L´opez’ , ’P´erez’)

* 1. Obtener todos los datos de los empleados que trabajan para el departamento 14.

SELECT \* FROM EMPLEADOS WHERE Departamento = 14

* 1. Obtener todos los datos de los empleados que trabajan para el departamento 37 y para el departamento 77.

/\* Con OR \*/

SELECT \* FROM EMPLEADOS

WHERE Departamento = 37 OR Departamento = 77

/\* Con IN \*/

SELECT \* FROM EMPLEADOS

WHERE Departamento IN (37,77)

* 1. Obtener todos los datos de los empleados cuyo apellido comience por ’P’.

SELECT \* FROM EMPLEADOS

WHERE Apellidos LIKE ’P%’

* 1. Obtener el presupuesto total de todos los departamentos.

SELECT SUM(Presupuesto) FROM DEPARTAMENTOS

* 1. Obtener el numero de empleados en cada departamento.

SELECT Departamento, COUNT(\*) FROM EMPLEADOS

GROUP BY Departamento

* 1. Obtener un listado completo de empleados, incluyendo por cada empleado los datos del empleado y de su departamento.

SELECT \*

FROM EMPLEADOS INNER JOIN DEPARTAMENTOS

ON EMPLEADOS.Departamento = DEPARTAMENTOS.Codigo

*2 Empleados*

* 1. Obtener un listado completo de empleados, incluyendo el nombre y apellidos del empleado junto al nombre y presupuesto de su departamento.

/\* Sin etiquetas \*/

SELECT EMPLEADOS.Nombre, Apellidos, DEPARTAMENTOS.Nombre, Presupuesto FROM EMPLEADOS INNER JOIN DEPARTAMENTOS

ON EMPLEADOS.Departamento = DEPARTAMENTOS.Codigo

/\* Con etiquetas \*/

SELECT E.Nombre, Apellidos, D.Nombre, Presupuesto FROM EMPLEADOS E INNER JOIN DEPARTAMENTOS D

ON E.Departamento = D.Codigo

* 1. Obtener los nombres y apellidos de los empleados que trabajen en departamentos cuyo presupuesto sea mayor de 60.000 ¤.

/\* Sin subconsulta \*/

SELECT EMPLEADOS.Nombre, Apellidos

FROM EMPLEADOS INNER JOIN DEPARTAMENTOS

ON EMPLEADOS.Departamento = DEPARTAMENTOS.Codigo AND DEPARTAMENTOS.Presupuesto > 60000

/\* Con subconsulta \*/

SELECT Nombre, Apellidos FROM EMPLEADOS WHERE Departamento IN

(SELECT Codigo FROM DEPARTAMENTOS WHERE Presupuesto > 60000)

* 1. Obtener los datos de los departamentos cuyo presupuesto es superior al presupues- to medio de todos los departamentos.

SELECT \*

FROM DEPARTAMENTOS

WHERE Presupuesto > (

SELECT AVG(Presupuesto) FROM DEPARTAMENTOS

)

* 1. Obtener los nombres (u´nicamente los nombres) de los departamentos que tienen m´as de dos empleados.

*2 Empleados*

/\* Con subconsulta \*/

SELECT Nombre FROM DEPARTAMENTOS

WHERE Codigo IN (

SELECT Departamento FROM EMPLEADOS

GROUP BY Departamento HAVING COUNT(\*) > 2

)

/\* Con UNION. No funciona si dos departamentos tienen el mismo nombre \*/

SELECT DEPARTAMENTOS.Nombre

FROM EMPLEADOS INNER JOIN DEPARTAMENTOS

ON Departamento = Codigo GROUP BY DEPARTAMENTOS.Nombre HAVING COUNT(\*) > 2

* 1. An˜adir un nuevo departamento: ‘Calidad’, con presupuesto de 40.000 ¤ y c´odigo

11. An˜adir un empleado vinculado al departamento reci´en creado: Esther V´azquez,

DNI: 89267109

INSERT INTO DEPARTAMENTOS

VALUES ( 11 , ’Calidad’ , 40000)

INSERT INTO EMPLEADOS

VALUES ( ’89267109’ , ’Esther’ , ’V´azquez’ , 11)

* 1. Aplicar un recorte presupuestario del 10 % a todos los departamentos.

UPDATE DEPARTAMENTOS SET Presupuesto = Presupuesto \* 0.9

* 1. Reasignar a los empleados del departamento de investigacio´n (co´digo 77) al depar- tamento de informa´tica (c´odigo 14).

UPDATE EMPLEADOS SET Departamento = 14 WHERE Departamento = 77

* 1. Despedir a todos los empleados que trabajan para el departamento de inform´atica (co´digo 14).

DELETE FROM EMPLEADOS

WHERE Departamento = 14

* 1. Despedir a todos los empleados que trabajen para departamentos cuyo presupuesto sea superior a los 60.000 ¤.

*2 Empleados*

DELETE FROM EMPLEADOS

WHERE Departamento IN (

SELECT Codigo FROM DEPARTAMENTO

WHERE Presupuesto >= 60000

)

* 1. Despedir a todos los empleados.

DELETE FROM EMPLEADOS

# 3

## Los Almacenes

ALMACENES

|  |  |  |
| --- | --- | --- |
| **PK** | **Codigo** | **int identity** |
|  | **Lugar Capacidad** | **nvarchar(100) int** |

|  |  |  |
| --- | --- | --- |
| CAJAS | | |
| **PK** | **NumReferencia** | **char(5)** |
| **FK1** | **Contenido Valor Almacen** | **nvarchar(100) int**  **int** |

* 1. Obtener todos los almacenes

SELECT \* FROM ALMACENES

* 1. Obtener todas las cajas cuyo contenido tenga un valor superior a 150 ¤.

SELECT \* FROM CAJAS WHERE Valor > 150

* 1. Obtener los tipos de contenidos de las cajas.

SELECT DISTINCT Contenido FROM CAJAS

* 1. Obtener el valor medio de todas las cajas.

SELECT AVG(Valor) FROM CAJAS

* 1. Obtener el valor medio de las cajas de cada almacen.

SELECT Almacen, AVG(Valor) FROM CAJAS

GROUP BY Almacen

*3 Los Almacenes*

* 1. Obtener los c´odigos de los almacenes en los cuales el valor medio de las cajas sea superior a 150 ¤.

SELECT Almacen, AVG(Valor) FROM CAJAS

GROUP BY Almacen

HAVING AVG(Valor) > 150

* 1. Obtener el numero de referencia de cada caja junto con el nombre de la ciudad en el que se encuentra.

SELECT NumReferencia, Lugar

FROM ALMACENES INNER JOIN CAJAS

ON ALMACENES.Codigo = CAJAS.Almacen

* 1. Obtener el numero de cajas que hay en cada almac´en.

/\* Esta consulta no tiene en cuenta los almacenes vac´ıos \*/ SELECT Almacen, COUNT(\*)

FROM CAJAS

GROUP BY Almacen

/\* Esta consulta tiene en cuenta los almacenes vac´ıos \*/ SELECT Codigo, COUNT(NumReferencia)

FROM ALMACENES LEFT JOIN CAJAS

ON ALMACENES.Codigo = CAJAS.Almacen GROUP BY Codigo

* 1. Obtener los c´odigos de los almacenes que esta´n saturados (los almacenes donde el numero de cajas es superior a la capacidad).

SELECT Codigo FROM ALMACENES

WHERE Capacidad < (

SELECT COUNT(\*) FROM CAJAS

WHERE Almacen = Codigo

)

* 1. Obtener los numeros de referencia de las cajas que esta´n en Bilbao.

*3 Los Almacenes*

/\* Sin subconsultas \*/ SELECT NumReferencia

FROM ALMACENES LEFT JOIN CAJAS

ON ALMACENES.Codigo = CAJAS.Almacen WHERE Lugar = ’Bilbao’

/\* Con subconsultas \*/ SELECT NumReferencia

FROM CAJAS

WHERE Almacen IN (

SELECT Codigo FROM ALMACENES

WHERE Lugar = ’Bilbao’

)

* 1. Insertar un nuevo almac´en en Barcelona con capacidad para 3 cajas.

INSERT INTO ALMACENES(Lugar,Capacidad) VALUES(’Barcelona’,3)

* 1. Insertar una nueva caja, con nu´mero de referencia ‘H5RT’, con contenido ‘Papel’, valor 200, y situada en el almac´en 2.

INSERT INTO CAJAS VALUES(’H5RT’,’Papel’,200,2)

* 1. Rebajar el valor de todas las cajas un 15 %.

UPDATE CAJAS SET Valor = Valor \* 0.85

* 1. Rebajar un 20 % el valor de todas las cajas cuyo valor sea superior al valor medio de todas las cajas.

UPDATE CAJAS SET Valor = Valor \* 0.80

WHERE Valor > (SELECT AVG(Valor) FROM CAJAS)

* 1. Eliminar todas las cajas cuyo valor sea inferior a 100 ¤.

DELETE FROM CAJAS WHERE Valor < 100

* 1. Vaciar el contenido de los almacenes que est´an saturados.

*3 Los Almacenes*

DELETE FROM CAJAS WHERE Almacen IN (

SELECT Codigo FROM ALMACENES

WHERE Capacidad < (

SELECT COUNT(\*) FROM CAJAS

WHERE Almacen = Codigo

)

)

# 4

## Pel´ıculas y Salas

|  |  |  |
| --- | --- | --- |
| SALAS | | |
| **PK** | **Codigo** | **int identity** |
| FK1 | **Nombre**  Pelicula | **nvarchar(100)**  int |

|  |  |  |
| --- | --- | --- |
| PELICULAS | | |
| **PK** | **Codigo** | **int identity** |
|  | **Nombre**  CalificacionEdad | **nvarchar(100)**  int |

* 1. Mostrar el nombre de todas las pel´ıculas.

SELECT Nombre FROM PELICULAS

* 1. Mostrar las distintas califi iones de edad que existen.

SELECT DISTINCT CalificacionEdad FROM PELICULAS

* 1. Mostrar todas las pel´ıculas que no han sido calificadas.

SELECT \* FROM PELICULAS WHERE CalificacionEdad IS NULL

* 1. Mostrar todas las salas que no proyectan ninguna pel´ıcula.

SELECT \* FROM SALAS WHERE Pelicula IS NULL

* 1. Mostrar la informacio´n de *todas* las salas y, si se proyecta alguna pel´ıcula en la sala, mostrar tambi´en la informacio´n de la pel´ıcula.

SELECT \*

FROM SALAS LEFT JOIN PELICULAS

ON SALAS.Pelicula = PELICULAS.Codigo

*4 Pel´ıculas y Salas*

* 1. Mostrar la informacio´n de *todas* las pel´ıculas y, si se proyecta en alguna sala, mostrar tambi´en la informaci´on de la sala.

SELECT \*

FROM SALAS RIGHT JOIN PELICULAS

ON SALAS.Pelicula = PELICULAS.Codigo

* 1. Mostrar los nombres de las pel´ıculas que no se proyectan en ninguna sala.

/\* Con JOIN \*/

SELECT PELICULAS.Nombre

FROM SALAS RIGHT JOIN PELICULAS

ON SALAS.Pelicula = PELICULAS.Codigo WHERE SALAS.Pelicula IS NULL

/\* Con Subconsulta \*/ SELECT Nombre FROM PELICULAS

WHERE Codigo NOT IN (

SELECT Pelicula FROM SALAS WHERE Pelicula IS NOT NULL

)

* 1. An˜adir una nueva pel´ıcula ‘Uno, Dos, Tres’, para mayores de 7 an˜os.

INSERT INTO PELICULAS(Nombre,CalificacionEdad) VALUES(’Uno, Dos, Tres’,7)

* 1. Hacer constar que todas las pel´ıculas no califi han sido califi ‘no reco- mendables para menores de 13 an˜os’.

UPDATE PELICULAS SET CalificacionEdad=13 WHERE CalificacionEdad IS NULL

* 1. Eliminar todas las salas que proyectan pel´ıculas recomendadas para todos los pu´bli- cos.

DELETE FROM SALAS WHERE Pelicula IN

(SELECT Codigo FROM PELICULAS WHERE CalificacionEdad = 0)

# 5

## Los Directores

|  |  |  |
| --- | --- | --- |
| DIRECTORES | | |
| **PK** | **DNI** | **varchar(8)** |
| FK2 FK1 | **NomApels** DNIJefe Despacho | **nvarchar(255)** varchar(8) int |

|  |  |  |
| --- | --- | --- |
| DESPACHOS | | |
| **PK** | **Numero** | **int** |
|  | **Capacidad** | **int** |

* 1. Mostrar el DNI, nombre y apellidos de todos los directores.

SELECT DNI, NomApels FROM DIRECTORES

* 1. Mostrar los datos de los directores que no tienen jefes.

SELECT \* FROM DIRECTORES WHERE DNIJefe IS NULL

* 1. Mostrar el nombre y apellidos de cada director, junto con la capacidad del despacho en el que se encuentra.

SELECT NomApels, Despacho, Capacidad FROM DIRECTORES INNER JOIN DESPACHOS

ON DIRECTORES.Despacho = DESPACHOS.Numero

* 1. Mostrar el numero de directores que hay en cada despacho.

*5 Los Directores*

/\* Sin tener en cuenta despachos vacios \*/ SELECT Despacho, COUNT(\*)

FROM DIRECTORES

GROUP BY Despacho

/\* Teniendo en cuenta despachos vacios \*/ SELECT Numero, COUNT(DNI)

FROM DESPACHOS LEFT JOIN DIRECTORES

ON DESPACHOS.Numero = DIRECTORES.Despacho GROUP BY Numero

* 1. Mostrar los datos de los directores cuyos jefes no tienen jefes.

SELECT \* FROM DIRECTORES WHERE DNIJefe IN

(SELECT DNI FROM DIRECTORES WHERE DNIJefe IS NULL)

* 1. Mostrar los nombres y apellidos de los directores junto con los de su jefe.

/\* Con INNER JOIN. No muestra directores que no tienen jefes. \*/ SELECT d1.NomApels, d2.NomApels

FROM DIRECTORES d1 INNER JOIN DIRECTORES d2 ON d1.DNIJefe = d2.DNI

/\* Con LEFT JOIN. Si muestra directores sin jefe. \*/ SELECT d1.NomApels, d2.NomApels

FROM DIRECTORES d1 LEFT JOIN DIRECTORES d2 ON d1.DNIJefe = d2.DNI

* 1. Mostrar el numero de despachos que esta´n sobreutilizados.

SELECT Numero FROM DESPACHOS

WHERE Capacidad < (

SELECT COUNT(\*) FROM DIRECTORES

WHERE Despacho = Numero

)

* 1. An˜adir un nuevo director llamado Paco P´erez, DNI 28301700, sin jefe, y situado en el despacho 124.

INSERT INTO DIRECTORES VALUES(’28301700’, ’Paco P´erez’, NULL, 124)

* 1. Asignar a todos los empleados apellidados P´erez un nuevo jefe con DNI 74568521.

*5 Los Directores*

UPDATE DIRECTORES SET DNIJefe = ’74568521’ WHERE NomApels LIKE ’%P´erez%’

* 1. Despedir a todos los directores, excepto a los que no tienen jefe.

DELETE FROM DIRECTORES WHERE DNIJefe IS NOT NULL

# 6

## Piezas y Proveedores

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PIEZAS | | |  |  |
| **PK** | **Codigo** | **int identity** |  |
|  |  |
|  | **Nombre** | **nvarchar(100)** |  |

|  |  |  |
| --- | --- | --- |
| SUMINISTRA | | |
| **PK,FK1 PK,FK2** | **CodigoPieza IdProveedor** | **int char(4)** |
|  | **Precio** | **int** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | PROVEEDORES | | |
|  | **PK** | **Id** | **char(4)** |
|  |  |
|  |  |  | **Nombre** | **nvarchar(100)** |

* 1. Obtener los nombres de todas las piezas.

SELECT Nombre FROM PIEZAS

* 1. Obtener todos los datos de todos los proveedores.

SELECT \* FROM PROVEEDORES

* 1. Obtener el precio medio al que se nos suministran las piezas.

SELECT CodigoPieza, AVG(Precio) FROM SUMINISTRA

GROUP BY CodigoPieza

* 1. Obtener los nombres de los proveedores que suministran la pieza 1.

*6 Piezas y Proveedores*

/\* Sin subconsulta \*/ SELECT PROVEEDORES.Nombre

FROM PROVEEDORES INNER JOIN SUMINISTRA

ON PROVEEDORES.Id = SUMINISTRA.IdProveedor

AND SUMINISTRA.CodigoPieza = 1

/\* Con subconsulta \*/ SELECT Nombre

FROM PROVEEDORES WHERE Id IN

(SELECT IdProveedor FROM SUMINISTRA WHERE CodigoPieza = 1)

* 1. Obtener los nombres de las piezas suministradas por el proveedor cuyo c´odigo es HAL.

/\* Sin subconsulta \*/ SELECT PIEZAS.Nombre

FROM PIEZAS INNER JOIN SUMINISTRA

ON PIEZAS.Codigo = SUMINISTRA.CodigoPieza AND SUMINISTRA.IdProveedor = ’HAL’

/\* Con subconsulta IN \*/ SELECT Nombre

FROM PIEZAS

WHERE Codigo IN

(SELECT CodigoPieza FROM SUMINISTRA WHERE IdProveedor = ’HAL’)

/\* Con subconsulta EXISTS \*/ SELECT Nombre

FROM PIEZAS WHERE EXISTS (

SELECT \* FROM SUMINISTRA

WHERE IdProveedor = ’HAL’

AND CodigoPieza = PIEZAS.Codigo

)

* 1. Obtener los nombres de los proveedores que suministran las piezas m´as caras, indicando el nombre de la pieza y el precio al que la suministran.

*6 Piezas y Proveedores*

SELECT p1.Nombre, pr1.Nombre, Precio FROM PIEZAS p1 INNER JOIN

(SUMINISTRA s1 INNER JOIN PROVEEDORES pr1

ON s1.IdProveedor = pr1.Id) ON p1.Codigo = s1.CodigoPieza WHERE Precio IN

(

SELECT MAX(Precio) FROM SUMINISTRA s2

GROUP BY s2.CodigoPieza

HAVING s2.CodigoPieza = p1.Codigo

)

* 1. Hacer constar en la base de datos que la empresa ”Skellington Supplies”(codigo TNBC) va a empezar a suministrarnos tuercas (co´digo 1) a 7 pesetas cada tuerca.

INSERT INTO SUMINISTRA VALUES ( ’TNBC’ , 1 , 7 )

* 1. Aumentar los precios en una unidad.

UPDATE SUMINISTRA SET Precio = Precio + 1

* 1. Hacer constar en la base de datos que la empresa ”Susan Calvin Corp.”(RBT) no va a suministrarnos ninguna pieza (aunque la empresa en s´ı va a seguir constando en nuestra base de datos).

DELETE FROM SUMINISTRA

WHERE IdProveedor = ’RBT’

* 1. Hacer constar en la base de datos que la empresa ”Susan Calvin Corp.”(RBT) ya no va a suministrarnos clavos (c´odigo 4)

DELETE FROM SUMINISTRA

WHERE IdProveedor = ’RBT’ AND CodigoPieza = 4

# 7

## Los Cient´ıficos

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CIENTIFICOS | | |  |  |
| **PK** | **DNI** | **varchar(8)** |  |
|  |  |
|  | **NomApels** | **nvarchar(255)** |  |

|  |  |  |
| --- | --- | --- |
| ASIGNADO\_A | | |
| **PK,FK1 PK,FK2** | **Cientifico Proyecto** | **varchar(8) char(4)** |
|  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | PROYECTO | | |
|  | **PK** | **Id** | **char(4)** |
|  |  |  | **Nombre Horas** | **nvarchar(255) int** |

* 1. Sacar una relacio´n completa de los cient´ıfi asignados a cada proyecto. Mostrar DNI, Nombre del cient´ıfi Identificador del proyecto y nombre del proyecto.

/\* Sin JOIN \*/

SELECT DNI, NomApels, Id, Nombre

FROM CIENTIFICOS C, ASIGNADO\_A A, PROYECTO P

WHERE C.DNI = A.Cientifico AND A.Proyecto = P.Id

/\* Con JOIN \*/

SELECT DNI, NomApels, Id, Nombre FROM CIENTIFICOS C INNER JOIN

(ASIGNADO\_A A INNER JOIN PROYECTO P

ON A.Proyecto = P.Id) ON C.DNI = A.Cientifico

* 1. Obtener el numero de proyectos al que est´a asignado cada cient´ıfi (mostrar el DNI y el nombre).

SELECT DNI, NomApels, COUNT(Proyecto) FROM CIENTIFICOS LEFT JOIN ASIGNADO\_A

ON CIENTIFICOS.DNI = ASIGNADO\_A.Cientifico

GROUP BY DNI, NomApels

*7 Los Cient´ıficos*

* 1. Obtener el numero de cient´ıfi asignados a cada proyecto (mostrar el identifi de proyecto y el nombre del proyecto).

SELECT Id, Nombre, COUNT(Proyecto) FROM PROYECTO LEFT JOIN ASIGNADO\_A ON PROYECTO.Id = ASIGNADO\_A.Proyecto

GROUP BY Id, Nombre

* 1. Obtener el numero de horas de dedicaci´on de cada cientifico.

SELECT DNI,NomApels, SUM(Horas) FROM CIENTIFICOS C LEFT JOIN

(ASIGNADO\_A A INNER JOIN PROYECTO P

ON A.Proyecto = P.Id) ON C.DNI = A.Cientifico GROUP BY DNI,NomApels

* 1. Obtener el DNI y nombre de los cient´ıficos que se dedican a m´as de un proyecto y cuya dedicaci´on media a cada proyecto sea superior a las 80 horas.

/\* Con dos subconsultas \*/ SELECT DNI, NomApels

FROM CIENTIFICOS C WHERE 1 <

(

SELECT COUNT(\*) FROM ASIGNADO\_A

WHERE Cientifico = C.DNI

) AND 80 < (

SELECT AVG(Horas)

FROM PROYECTO INNER JOIN ASIGNADO\_A ON PROYECTO.Id = ASIGNADO\_A.Proyecto

WHERE Cientifico = C.DNI

)

/\* Juntando tablas y con HAVING \*/ SELECT DNI, NomApels

FROM CIENTIFICOS C, ASIGNADO\_A A, PROYECTO P

WHERE C.DNI = A.Cientifico AND P.Id = A.Proyecto GROUP BY DNI, NomApels

HAVING COUNT(Proyecto) > 1 AND AVG(Horas) > 80

# 8

## Los Grandes Almacenes

|  |  |  |  |
| --- | --- | --- | --- |
| CAJEROS | | | |
| **PK** | **Codigo** | | **int identity** |
|  | **NomApels** | | **nvarchar(255)** |
|  | |  |  |
|  | |  | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PRODUCTOS | | |  |  |
| **PK** | **Codigo** | **int identity** |  |
|  | **Nombre Precio** | **nvarchar(100) int** |  |  |

|  |  |  |
| --- | --- | --- |
| VENTA | | |
| **PK,FK1 PK,FK2 PK,FK3** | **Cajero Maquina Producto** | **int int int** |
|  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | MAQUINAS\_REGISTRADORAS | | |
|  | **PK** | **Codigo** | **int identity** |
|  |  |
|  |  |  | **Piso** | **int** |

* 1. Mostrar el nu´mero de ventas de cada producto, ordenado de m´as a menos ventas.

SELECT Codigo, Nombre, COUNT(VENTA.Producto) FROM PRODUCTOS LEFT JOIN VENTA

ON PRODUCTOS.Codigo = VENTA.Producto GROUP BY Codigo, Nombre

ORDER BY COUNT(VENTA.Producto) DESC

* 1. Obtener un informe completo de ventas, indicando el nombre del cajero que rea-

lizo´

la venta, nombre y precios de los productos vendidos, y piso en el que se

encuentra la maquina registradora donde se realizo la venta.

*8 Los Grandes Almacenes*

/\* Sin JOIN \*/

SELECT NomApels, Nombre, Precio, Piso

FROM VENTA V, CAJEROS C, PRODUCTOS P, MAQUINAS\_REGISTRADORAS M

WHERE V.Cajero = C.Codigo AND V.Producto = P.Codigo AND V.Maquina = M.Codigo

/\* Con JOIN \*/

SELECT NomApels, Nombre, Precio, Piso FROM CAJEROS C INNER JOIN

(PRODUCTOS P INNER JOIN (MAQUINAS\_REGISTRADORAS M INNER JOIN VENTA V

ON V.Maquina = M.Codigo) ON V.Producto = P.Codigo)

ON V.Cajero = C.Codigo

* 1. Obtener las ventas totales realizadas en cada piso.

SELECT Piso, SUM(Precio)

FROM VENTA V, PRODUCTOS P, MAQUINAS\_REGISTRADORAS M

WHERE V.Producto = P.Codigo AND V.Maquina = M.Codigo GROUP BY Piso

* 1. Obtener el codigo y nombre de cada empleado junto con el importe total de sus ventas.

SELECT C.Codigo, C.NomApels, SUM(Precio) FROM PRODUCTOS P INNER JOIN

(CAJEROS C LEFT JOIN VENTA V

ON V.Cajero = C.Codigo) ON V.Producto = P.Codigo GROUP BY C.Codigo, NomApels

* 1. Obtener el c´odigo y nombre de aquellos cajeros que hayan realizado ventas en pisos cuyas ventas totales sean inferiores a los 500 ¤.

*8 Los Grandes Almacenes*

SELECT Codigo, NomApels FROM CAJEROS

WHERE Codigo IN (

SELECT Cajero FROM VENTA

WHERE Maquina IN (

SELECT Codigo FROM MAQUINAS\_REGISTRADORAS WHERE Piso IN

(

SELECT Piso

FROM VENTA V, PRODUCTOS P, MAQUINAS\_REGISTRADORAS M

WHERE V.Producto = P.Codigo AND V.Maquina = M.Codigo GROUP BY Piso

HAVING SUM(Precio) < 500

)

)

)

*Ejercicios de Bases de Datos* 31

# 9

## Los Investigadores

FACULTAD

**PK**

**Codigo**

**int**

**Nombre**

**nvarchar(100)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| INVESTIGADORES | | |  |  |
| **PK** | **DNI** | **varchar(8)** |  |
| **FK1** | **NomApels Facultad** | **nvarchar(255) int** |  |  |

|  |  |  |
| --- | --- | --- |
| RESERVA | | |
| **PK,FK1 PK,FK2** | **DNI**  **NumSerie** | **varchar(8) char(4)** |
|  | **Comienzo Fin** | **datetime datetime** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | EQUIPOS | | |
|  | **PK** | **NumSerie** | **char(4)** |
|  |  | **FK1** | **Nombre Facultad** | **nvarchar(100) int** |

* 1. Obtener el DNI y nombre de aquellos investigadores que han realizado m´as de una reserva.

/\* Juntando tablas \*/ SELECT I.DNI, NomApels

FROM INVESTIGADORES I LEFT JOIN RESERVA R ON R.DNI = I.DNI

GROUP BY I.DNI, NomApels HAVING COUNT(R.DNI) > 1

/\* Con subconsulta \*/ SELECT DNI, NomApels

FROM INVESTIGADORES WHERE DNI IN

(

SELECT DNI FROM RESERVA GROUP BY DNI

HAVING COUNT(\*) > 1

)

33

*9 Los Investigadores*

* 1. Obtener un listado completa de reservas, incluyendo los siguientes datos:

DNI y nombre del investigador, junto con el nombre de su facultad.

Numero de serie y nombre del equipo reservado, junto con el nombre de la facultad a la que pertenece.

Fecha de comienzo y fi de la reserva.

SELECT I.DNI, NomApels, F\_INV.Nombre, E.NumSerie, E.Nombre, F\_EQUIP.Nombre, Comienzo, Fin

FROM RESERVA R,INVESTIGADORES I,EQUIPOS E,FACULTAD F\_INV, FACULTAD F\_EQUIP WHERE R.DNI = I.DNI

AND R.NumSerie = E.NumSerie AND I.Facultad = F\_INV.Codigo

AND E.Facultad = F\_EQUIP.Codigo

* 1. Obtener el DNI y el nombre de los investigadores que han reservado equipos que no son de su facultad.

/\* Juntando tablas \*/

SELECT DISTINCT I.DNI,NomApels

FROM RESERVA R,INVESTIGADORES I,EQUIPOS E WHERE R.DNI = I.DNI

AND R.NumSerie = E.NumSerie AND I.Facultad <> E.Facultad

/\* Con EXISTS \*/ SELECT DNI, NomApels

FROM INVESTIGADORES I WHERE EXISTS

(

SELECT \*

FROM RESERVA R INNER JOIN EQUIPOS E

ON R.NumSerie = E.NumSerie WHERE R.DNI = I.DNI

AND I.Facultad <> E.Facultad

)

* 1. Obtener los nombres de las facultades en las que ningu´n investigador ha realizado una reserva.

*9 Los Investigadores*

SELECT Nombre FROM FACULTAD

WHERE Codigo IN (

SELECT Facultad

FROM INVESTIGADORES I LEFT JOIN RESERVA R ON I.DNI = R.DNI

GROUP BY Facultad HAVING COUNT(R.DNI)=0

)

* 1. Obtener los nombres de las facultades con investigadores ’ociosos’ (investigadores que no han realizado ninguna reserva).

SELECT Nombre FROM FACULTAD

WHERE Codigo IN (

SELECT Facultad FROM INVESTIGADORES WHERE DNI NOT IN

(

SELECT DNI FROM RESERVA

)

)

* 1. Obtener el nu´mero de serie y nombre de los equipos que nunca han sido reservados.

/\* Juntando tablas \*/ SELECT E.NumSerie, Nombre

FROM EQUIPOS E LEFT JOIN RESERVA R

ON R.NumSerie = E.NumSerie GROUP BY E.NumSerie,Nombre HAVING COUNT(R.NumSerie)=0

/\* Con subconsulta IN \*/

SELECT NumSerie, Nombre FROM EQUIPOS WHERE NumSerie NOT IN

(

SELECT NumSerie FROM RESERVA

)

/\* Con EXISTS \*/ SELECT NumSerie, Nombre

FROM EQUIPOS E WHERE NOT EXISTS (

SELECT \* FROM RESERVA R

WHERE R.NumSerie = E.NumSerie

)

*Ejercicios de Bases de Datos* 35