CÁLCULO RELACIONAL

Dadas las siguientes tablas:

T

T#	Talla	Color
T1	1	Blanco
T2	1	Negro
T3	1	Rojo
T4	2	Blanco
T5	2	Negro
T6	3	Blanco

P

P#	Nombre	Ciudad
P1	Juan	Madrid
P2	Ana	Barcelona
P3	José	Málaga

TP

P#	T#
P1	T1
P1	T2
P1	T6
P2	T1
P2	T4
P3	T2

1. Crea la base de datos.
2. Crea las tablas correspondientes.
3. Introduce los datos que se adjuntan en las tablas.

---

1. Nombres de los proveedores que suministran la pieza "T2".

- Con producto cartesiano:

```
select nombre from p, t, tp where t.t_id='T2' and t.t_id=tp.t_id and
p.p_id=tp.p_id;
```

Esta consulta busca el contenido del campo "**nombre**" de la tabla "**p**", estableciendo como condiciones que el "**t_id**" de la tabla "**t**" debe ser igual a "**T2**", además de ser igual que el campo "**t_id**" en la tabla "**tp**", y que el campo "**p_id**" de la tabla "**p**" tiene que ser igual que el campo con el mismo nombre de la tabla "**tp**".

- Sin producto cartesiano:

```
select nombre from p join tp on p.p_id=tp.p_id and tp.t_id='T2' join t on
tp.t_id=t.t_id;
```

Esta otra realiza **la misma acción**, pero de una forma más **eficiente**.

Para empezar selecciona el campo "nombre" de la tabla "p". Seguidamente, une la tabla "tp" con la tabla anterior ("p") estableciendo como condición en la cláusula "on" que el campo "p_id" debe ser igual en las dos tablas.

A continuación, con una cláusula "**and**" incluye una condición más, que establece que el campo "**t_id**" en la tabla "**tp**" debe ser igual a "**T2**, para después unirlo con otro "**join**" con la tabla "**t**", con la condición en el "**on**" de que el campo "**t_id**" debe ser igual en las dos tablas.

- Resultado:

2. Número de los proveedores que suministran al menos una pieza roja.

- Con producto cartesiano:

```
select count(distinct tp.p_id) from p, tp, t where p.p_id=tp.p_id and
tp.t_id=t.t_id and t.color='Rojo';
```

La cláusula "**select count(distinct tp.p_id) from p, tp t**" muestra el número de coincidencia **únicas** (debido al "**distinct**") en el campo "**tp.p_id**" de las tablas "**p**" y "**t**".

El "**where**" especifica que las condiciones que se deben cumplir son:

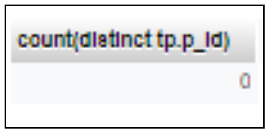
1. Que el campo "**p_id**" sea igual en las tablas "**p**" y "**tp**"
2. Que el campo "**t_id**" sea igual en las tablas "**tp**" y "**t**"
3. Que el campo "**color**" sea igual a "**Rojo**"

- Sin producto cartesiano:

```
select count(distinct tp.p_id) from p join tp on p.p_id=tp.p_id join t on
tp.t_id=t.t_id where t.color='Rojo';
```

La cláusula "**distinct**" establece que se seleccione el número de entradas **únicas** en el campo "**tp.p_id**".

  - Resultado:



3. Nombres de los proveedores que suministran todas las piezas.

  - Con producto cartesiano:

```
select distinct nombre from p where not exists(select * from t where not
exists(select * from tp where t.t_id=tp.t_id and tp.p_id=p.p_id));
```

Esta consulta selecciona, para empezar, los nombres **de forma única**. A continuación, con la cláusula "**where not exists**" especifica que solo entrarán las entradas en la tabla "**p**" donde **no existen** entradas relacionadas en la tabla "**t**" que no tengan relación con entradas en la tabla "**tp**" utilizando el campo "**t_id**".

- Sin producto cartesiano:

```
select nombre from p join tp on p.p_id=tp.p_id where not exists(select 1
from t where not exists(select 1 from tp where t.t_id=tp.t_id and
tp.p_id=p.p_id));
```

Aquí se utiliza la cláusula "**join**" para relacionar las tablas "**p**" y "**tp**" mediante el campo "**p_id**" y luego se utiliza la cláusula "**where not exists**" para filtrar los resultados de la tabla "**p**" en base a una **subconsulta**.

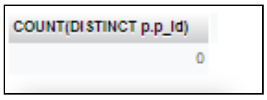
Esta subconsulta ("**(select 1 from t where not exists(select 1 from tp where t.t_id=tp.t_id and tp.p_id=p.p_id))**") selecciona las entradas de la tabla "**t**" donde **no existen** entradas relacionadas en la tabla "**tp**" mediante los campos "**t_id**" y "**p_id**", lo que permite seleccionar solo los proveedores que suministran todas las piezas, es decir, aquellos que están relacionados con todas las entradas de la tabla "**t**" y "**tp**".

- Resultado:

4. Número de los proveedores llamados "José" y que vivan en Madrid.

- Con producto cartesiano:

```
select count(distinct p.p_id) from p, tp, t where p.p_id=tp.p_id and
tp.t_id=t.t_id and p.nombre='José' and p.ciudad='Madrid';
```



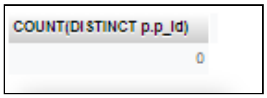
Esta consulta muestra los proveedores llamados "José" **Y** que viven en Madrid. Para mostrar los proveedores llamados "José" **O** que vivan en Madrid, se utiliza la siguiente:

```
select count(distinct p.p_id) from p where(p.nombre='José' or
p.ciudad='Madrid');
```

- Sin producto cartesiano:

```
select count(distinct p.p_id) from p where p.nombre='José' and
p.ciudad='Madrid';
```

- Resultado:



5. Número de proveedores y ciudades que suministran la pieza "T2".

- Con producto cartesiano:

```
select count(distinct p.p_id) , count(distinct p.ciudad) from p, tp, t where
t.t_id='T2' and t.t_id=tp.t_id and p.p_id=tp.p_id;
```

Primero se cuentan las **entradas únicas** del campo "**p.p_id**" y del campo "**p.ciudad**".

Después, se establecen la tablas de donde se saca la información y las condiciones para que el "**t_id**" sea "**T2**" y se establecen las relaciones mediante los campos "**t_id**" y "**p_id**".

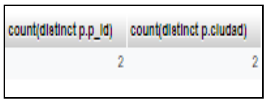
- Sin producto cartesiano:

```
select count(distinct p.p_id) , count(distinct p.ciudad) from p join tp on
p.p_id=tp.p_id join t on tp.t_id=t.t_id where t.t_id='t2';
```

Este código realiza una consulta para contar el número de **elementos únicos** en **dos campos específicos** de las tablas relacionadas "**p**", "**t**" y "**tp**".

Utiliza "**join**" para relacionar las tablas "**p**" con "**tp**" mediante el campo "**p_id**", y "**tp**" con "**t**" mediante el campo "**t_id**" y luego se utiliza una cláusula "**where**" para filtrar los resultados con el campo "**t_id='t2'**". La consulta selecciona el número de elementos únicos en los campos "**p.p_id**" y "**p.ciudad**" y los contabiliza.

- Resultado:



6. Colores de las piezas de los partes de la tabla "TP" suministrados por "P1".

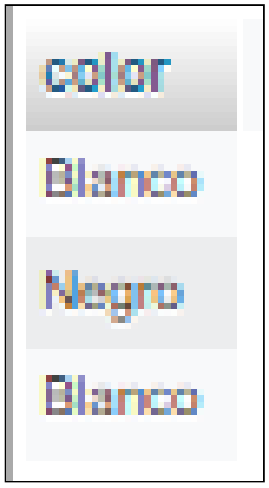
- Con producto cartesiano:

```
select t.color from t,tp,p where p.p_id='p1' and t.t_id=tp.t_id and
p.p_id=tp.p_id;
```

- Sin producto cartesiano:

```
select t.color from t join tp on t.t_id=tp.t_id join p on tp.p_id=p.p_id
where p.p_id='p1';
```

- Resultado:



7. Nombre de los proveedores de Málaga o que suministran piezas de color blanco.

- Con producto cartesiano:

```
select distinct p.nombre from p,tp,t where(p.ciudad='Málaga' or
t.color='Blanco') and t.t_id=tp.t_id and p.p_id=tp.p_id;
```

- Sin producto cartesiano:

```
select distinct p.nombre from p join tp on p.p_id=tp.p_id join t on
tp.t_id=t.t_id where p.ciudad='Málaga' or t.color='Blanco';
```

- Resultado:

8. Nombres de los proveedores de Madrid junto a los códigos de los productos que suministran.

- Con producto cartesiano:

```
select p.nombre, t.t_id from p,tp,t where p.ciudad='Madrid' and
t.t_id=tp.t_id and p.p_id=tp.p_id;
```

- Sin producto cartesiano:

```
select p.nombre, t.t_id from p join tp on p.p_id=tp.p_id join t on
tp.t_id=t.t_id where p.ciudad='Madrid';
```

- Resultado:

nombre	t_id
Juan	T1
Juan	T2
Juan	T6

9. Ciudades de los proveedores junto a los códigos de los productos que suministran y el correspondiente color.

- Con producto cartesiano:

```
select p.ciudad, t.t_id, t.color from p,tp,t where t.t_id=tp.t_id and
p.p_id=tp.p_id;
```

- Sin producto cartesiano:

```
select p.ciudad, t.t_id, t.color from p join tp on p.p_id=tp.p_id join t on
tp.t_id=t.t_id;
```

- Resultado:

ciudad	t_id	color
Madrid	T1	Blanco
Madrid	T2	Negro
Madrid	T6	Blanco
Barcelona	T1	Blanco
Barcelona	T4	Blanco
Málaga	T2	Negro

10. Igual que el apartado anterior pero para el proveedor "P1".

- Con producto cartesiano:

```
select p.ciudad, t.t_id, t.color from p,tp,t where p.p_id='P1' and
t.t_id=tp.t_id and p.p_id=tp.p_id;
```

- Sin producto cartesiano:

```
select p.ciudad, t.t_id, t.color from p join tp on p.p_id=tp.p_id join t on
tp.t_id=t.t_id where p.p_id='P1';
```

- Resultado:

ciudad	t_id	color
Madrid	T1	Blanco
Madrid	T2	Negro
Madrid	T6	Blanco

Anexo

```
-- phpMyAdmin SQL Dump
-- version 5.2.0
-- https://www.phpmyadmin.net/
--
-- Host: 127.0.0.1:3306
-- Generation Time: Jan 11, 2023 at 11:37 PM
-- Server version: 8.0.31
-- PHP Version: 8.0.26

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
START TRANSACTION;
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 */;

--
-- Database: `ejercicio1`
--

-- --------------------------------------------------------

--
-- Table structure for table `p`
--
```

```
DROP TABLE IF EXISTS `p`;
CREATE TABLE IF NOT EXISTS `p` (
  `p_id` varchar(2) NOT NULL,
  `nombre` varchar(20) NOT NULL,
  `ciudad` varchar(25) NOT NULL,
  PRIMARY KEY (`p_id`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

--
-- Dumping data for table `p`
--

INSERT INTO `p` (`p_id`, `nombre`, `ciudad`) VALUES
('P1', 'Juan', 'Madrid'),
('P2', 'Ana', 'Barcelona'),
('P3', 'José', 'Málaga');

-- --------------------------------------------------------

--
-- Table structure for table `t`
--

DROP TABLE IF EXISTS `t`;
CREATE TABLE IF NOT EXISTS `t` (
  `t_id` varchar(2) NOT NULL,
  `talla` int NOT NULL,
  `color` varchar(10) NOT NULL,
  PRIMARY KEY (`t_id`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

--
-- Dumping data for table `t`
--

INSERT INTO `t` (`t_id`, `talla`, `color`) VALUES
('T1', 1, 'Blanco'),
('T2', 1, 'Negro'),
('T3', 1, 'Rojo'),
('T4', 2, 'Blanco'),
('T5', 2, 'Negro'),
('T6', 3, 'Blanco');

-- --------------------------------------------------------

--
-- Table structure for table `tp`
--

DROP TABLE IF EXISTS `tp`;
CREATE TABLE IF NOT EXISTS `tp` (
  `p_id` varchar(2) NOT NULL,
  `t_id` varchar(2) NOT NULL,
```

```
  KEY `p_id` (`p_id`),
  KEY `t_id` (`t_id`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

--
-- Dumping data for table `tp`
--

INSERT INTO `tp` (`p_id`, `t_id`) VALUES
('P1', 'T1'),
('P1', 'T2'),
('P1', 'T6'),
('P2', 'T1'),
('P2', 'T4'),
('P3', 'T2');
COMMIT;

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