

UNIDAD 2 TALLER 3

Daniel Andrés Pinzón Jay

Id: 819793

Programa: Ingeniería de sistemas

Gestión de la información

Universidad cooperativa de Colombia

Sede: Bucaramanga

2024

PARTE 1: DISEÑO VISTAS

CREACION DE VISTAS

Vista: Consulta almacenada que se puede usar como una tabla virtual. Es decir, Una vista nos puede ayudar a guardar una consulta en sí misma dentro de una nueva tabla, esto significa que los datos no se almacenan 2 veces en la base de datos, lo cual sirve para acceder a la información de manera mas eficiente y cabe destacar que las vistas no ocupan espacio en memoria.

Estructura vista:

```
CREATE VIEW nombre_vista AS
SELECT nombre_Atributos_Deseados.
FROM tabla_Donde_Saldran_Los_Atributos
WHERE condición_Consulta
;
```

OJO: dentro de las vistas podemos usar:

- Estructuras Where
- JOINS y todos sus derivados
- Consultas anidadas
- Funciones de agregación (COUNT, AVG, MAX, MIN, SUM)

VISTAS EJERCICIO CASA SOFTWARE

Vista 1: Nuestra primera vista consistirá en obtener aquellos empleados de tipo programador que manejan el lenguaje de programación “Python”.

En nuestra base de datos se verá así:

```
MariaDB [CasaSoftware]> CREATE VIEW programadores_python AS
-> SELECT e.Nombre, p.Cedula, p.Lenguajes
-> FROM programador p
-> JOIN empleado e ON p.Cedula = e.Cedula
-> WHERE p.Lenguajes = "Python";
Query OK, 0 rows affected (0.028 sec)
```

(debemos realizar un JOIN entre empleado y programador ya que el nombre del programador se guarda en Empleado y no en programador)

Vista 2: En nuestra segunda vista recopilaremos los Empleados Analistas. Ya que “Analista” es un subtipo de la tabla “Empleado” pero al no contar con atributos propios no cuenta con tabla propia, así que esta vista nos servirá para recopilar los analistas en el sistema a través del siguiente código:

```
MariaDB [CasaSoftware]> CREATE VIEW Empleados_Analista AS
-> SELECT *
-> FROM Empleado
-> WHERE tipo = "Analista";
Query OK, 0 rows affected (0.035 sec)
```

INSERCIÓN DE DATOS EN CASA SOFTWARE:

Ya que nuestras 2 vistas creadas en esta base de datos se usan más que todo con la tabla “Empleado” Ingresaremos 5 EMPLEADOS los cuales son:

1. Empleado “Daniel Jay” tipo Programador cuyo lenguaje es Python

```
MariaDB [CasaSoftware]> INSERT INTO Empleado (Cedula, CodigoEmpleado, Nombre, Direccion, Titulacion, Años, tipo)
-> VALUES (1005, 01, "Daniel Jay", "Calle 123", "Ingeniero", 20, "Programador");
Query OK, 1 row affected (0.057 sec)

MariaDB [CasaSoftware]> INSERT INTO Programador (Cedula, Lenguajes)
-> VALUES (1005, "Python");
Query OK, 1 row affected (0.020 sec)
```

2. Empleado “Yeimy Corzo” tipo Programador cuyo lenguaje es Python

```
MariaDB [CasaSoftware]> INSERT INTO Empleado (Cedula, CodigoEmpleado, Nombre, Direccion, Titulacion, Años, tipo)
-> VALUES (1006, 02, "Yeimy Corzo", "Avenida 52", "Ingeniero", 20, "Programador");
Query OK, 1 row affected (0.024 sec)

MariaDB [CasaSoftware]> INSERT INTO Programador (Cedula, Lenguajes)
-> VALUES (1006, "Python");
Query OK, 1 row affected (0.021 sec)
```

3.Empleado “Pedro Rodriguez” tipo Programador cuyo lenguaje es Javascript

```
MariaDB [CasaSoftware]> INSERT INTO Empleado (Cedula, CodigoEmpleado, Nombre, Direccion, Titulacion, Años, tipo)
-> VALUES (1007, 03, "Pedro Rodriguez", "Carrera 456","Ingeniero", 25, "Programador");
Query OK, 1 row affected (0.002 sec)

MariaDB [CasaSoftware]> INSERT INTO Programador (Cedula, Lenguajes)
-> VALUES (1007, "Javascript");
Query OK, 1 row affected (0.020 sec)
```

Nota: Los anteriores al ser de tipo programador debimos hacer un “INSERT INTO” tanto en la tabla “Empleado” como en la tabla “Programador”, pero ahora con Analista no es necesario

4.Empleado “Juan Lopez” tipo Analista

```
MariaDB [CasaSoftware]> INSERT INTO Empleado (Cedula, CodigoEmpleado, Nombre, Direccion, Titulacion, Años, tipo)
-> VALUES (1008, 04, "Juan Lopez", "Calle 234","Analista", 32, "Analista");
Query OK, 1 row affected (0.020 sec)
```

5.Empleado “Julio Gonzalez” tipo Analista

```
MariaDB [CasaSoftware]> INSERT INTO Empleado (Cedula, CodigoEmpleado, Nombre, Direccion, Titulacion, Años, tipo)
-> VALUES (1009, 05, "Julio Gonzalez", "Carrera 48","Analista", 29, "Analista");
Query OK, 1 row affected (0.020 sec)
```

PRUEBA DE LAS VISTAS:

Vista 1: Para la prueba de la vista 1 “Programadores_python” deberá devolverme 2 datos ya que ingrese 3 Programadores, pero de esos 3 solo dos usa PYTHON y el otro usa JAVASCRIPT

```
MariaDB [CasaSoftware]> SELECT * FROM programadores_python;
+-----+-----+-----+
| Nombre | Cedula | Lenguajes |
+-----+-----+-----+
| Daniel Jay | 1005 | Python |
| Yeimy Corzo | 1006 | Python |
+-----+-----+-----+
2 rows in set (0.022 sec)
```

Vista 2: Para la prueba de la vista 2 “Empleado_analista” Me deberá devolver dos (los de Juan lopez y los de Julio Gonzalez)

```
MariaDB [CasaSoftware]> SELECT * FROM Empleados_Analista;
+-----+-----+-----+-----+-----+-----+-----+
| Cedula | CodigoEmpleado | Nombre | Direccion | Titulacion | Años | tipo |
+-----+-----+-----+-----+-----+-----+-----+
| 1008 | 4 | Juan Lopez | Calle 234 | Analista | 32 | Analista |
| 1009 | 5 | Julio Gonzalez | Carrera 48 | Analista | 29 | Analista |
+-----+-----+-----+-----+-----+-----+-----+
2 rows in set (0.022 sec)
```

Como podemos ver me devuelve perfectamente los datos de los Analistas.

VISTAS EJERCICIO AEREOLINEA

Vista 1: La primera vista consistirá en obtener los pilotos junto al modelo de avión que manejan. Como los datos de piloto se guardan mayoritariamente en la tabla persona deberemos realizar dos Joins, Uno de persona con piloto y otro propiamente de piloto con el modelo del avión.

```
MariaDB [Aereolinea]> CREATE VIEW piloto_ModeloAvion AS
-> SELECT p.nombre as nombre_piloto, a.modelo as modelo_avion
-> FROM Persona p
-> JOIN Piloto pil ON p.Cedula = pil.Cedula
-> JOIN Avion a ON pil.IDavion = a.IDavion
-> GROUP BY p.nombre;
Query OK, 0 rows affected (0.009 sec)
```

Vista 2: Dentro de esta nueva vista obtendremos el dato del recepcionista y todos los pasajeros que atendió

Dentro de este código realizaremos 3 JOINS. Esto debido a que necesitamos el nombre del recepcionista y el nombre del pasajero. Por lo que necesitamos dos JOINS para unir la llave primaria de "Persona" con la llave foránea de "Recepcionista" y "Pasajero" para obtener su nombre. Igual manera necesitamos otro JOIN que unirá la llave foránea de "Pasajero" llamada "CedulaRecepcionista" para unirla con la llave primaria de recepcionista.

GROUP_CONCAT () nos permitirá concatenar valores de varias filas en una sola. En nuestra base de datos se verá así:

```
MariaDB [Aereolinea]> CREATE VIEW Recepcionista_pasajero AS
-> SELECT p.Nombre AS nombre_recepcionista, GROUP_CONCAT(pp.Nombre) AS pasajeros_atendidos
-> FROM Persona p
-> JOIN Recepcionista r ON p.Cedula = r.Cedula
-> JOIN Pasajero pa ON r.Cedula = pa.CedulaRecepcionista
-> JOIN Persona pp ON pa.Cedula = pp.Cedula
-> GROUP BY p.Nombre;
Query OK, 0 rows affected (0.015 sec)
```

INSERCIÓN DE DATOS EN AEREOLINEA:

Lo primero que haremos será ingresar 4 aviones los cuales nos servirán para probar la vista de Piloto-ModeloAvion

```
MariaDB [Aereolinea]> INSERT INTO Avion (IDavion, Modelo, NumeroSerie, AltitudMaxima, VelocidadMaxima)
-> VALUES
-> (1001, "Boeing 737", 12345, 10000, 900),
-> (1002, "Airbus A320", 54321, 12000, 850),
-> (1003, "Embraer 145", 98765, 8000, 750),
```

Luego de esto ingresare 8 datos a Persona, de los cuales los primeros 3 serán Pilotos, los de cedula 4 y 5 serán Recepcionistas y los demás serán Pasajeros.

```

MariaDB [Aereolinea]> INSERT INTO Persona (Cedula, Nombre, FechaNac, Edad, Direccion, Genero) VALUES
-> (1, "Juan", '1990-05-15', 34, "Calle 123", "M"),
-> (2, "María", '1995-09-20', 29, "Avenida Principal", "F"),
-> (3, "Carlos", '1985-03-10', 39, "Carrera 45", "M"),
-> (4, "Ana", '2000-11-03', 23, "Calle 67", "F"),
-> (5, "Pedro", '1978-07-28', 46, "Avenida Central", "M"),
-> (6, "Laura", '1992-12-12', 32, "Carrera 89", "F"),
-> (7, "Daniel", '2000-01-12', 24, "Calle 212", "M"),
-> (8, "Juanita", '1999-12-23', 24, "Calle w4", "F");
Query OK, 8 rows affected (0.020 sec)
Records: 8 Duplicates: 0 Warnings: 0

```

Ahora ingresaremos los empleados a la tabla Empleado, luego dividimos cada empleado en su tabla correspondiente:

```

MariaDB [Aereolinea]> INSERT INTO Empleado (Cedula, IATAaereopuerto, Salario) VALUES
-> (1, 4145, 50000),
-> (2, 4145, 52000),
-> (3, 4145, 48000),
-> (4, 4145, 51000),
-> (5, 4145, 49000);
Query OK, 5 rows affected (0.004 sec)
Records: 5 Duplicates: 0 Warnings: 0

```

Pilotos:

```

MariaDB [Aereolinea]> INSERT INTO Piloto (Cedula, IDavion, AñosExperiencia, TipoAvion)
-> VALUES
-> (1, 1001, 5, "Comercial"),
-> (2, 1002, 2, "Comercial"),
-> (3, 1002, 8, "Comercial");
Query OK, 3 rows affected (0.020 sec)
Records: 3 Duplicates: 0 Warnings: 0

```

Recepcionistas:

```

MariaDB [Aereolinea]> INSERT INTO Recepcionista (Cedula, Cabina, Horario) VALUES
-> (4, 1, '09:00:00'),
-> (5, 2, '10:30:00');
Query OK, 2 rows affected (0.020 sec)
Records: 2 Duplicates: 0 Warnings: 0

```

Pasajeros:

```

MariaDB [Aereolinea]> INSERT INTO Pasajero (Cedula, CedulaRecepcionista) VALUES
-> (6, 4),
-> (7, 4),
-> (8, 5);
Query OK, 3 rows affected (0.020 sec)
Records: 3 Duplicates: 0 Warnings: 0

```

Hemos establecido que cada piloto maneja 1 avion diferente cada uno con su propio modelo y hemos establecido que el recepcionista con cedula "4" atendió a los Pasajeros con cedula "6" y "7". De igual manera el recepcionista con cedula "5" atendió al pasajero con cedula "8".

PRUEBA DE LAS VISTAS:

Vista 1: En la vista 1 nos debería devolver el nombre del piloto junto al modelo de avión que manejan.

```
MariaDB [Aereolinea]> SELECT * FROM piloto_ModeloAvion;
+-----+-----+
| nombre_piloto | modelo_avion |
+-----+-----+
| Carlos        | Airbus A320  |
| Juan          | Boeing 737   |
| María         | Airbus A320  |
+-----+-----+
3 rows in set (0.001 sec)
```

Vista 2: En la vista 2 nos debe devolver a los recepcionistas y las personas que atendió cada uno.

```
MariaDB [Aereolinea]> SELECT * FROM Recepcionista_pasajero;
+-----+-----+
| nombre_recepcionista | pasajeros_atendidos |
+-----+-----+
| Ana                  | Laura,Daniel        |
| Pedro                | Juanita              |
+-----+-----+
2 rows in set (0.013 sec)
```

VISTAS EJERCICIO ABOGADOS

Vista 1: La primera vista consistirá en agrupar a la gente que presento alguna denuncia “Acusador” bajo el motivo de “Violencia Física”.

```
MariaDB [Abogados]> CREATE VIEW Acusador_ViolenciaFisica AS
-> SELECT *
-> FROM Acusador
-> WHERE MotivoDemanda = "Violencia fisica";
Query OK, 0 rows affected (0.012 sec)
```

Vista 2: En esta vista vamos a recopilar a los testigos junto al ID de sus testimonios realizados (Puede ser más de uno). Usaremos la concatenación por si el testigo ha generado más de 1 testimonio.

```
MariaDB [Abogados]> CREATE VIEW Testigo_Acusatorio AS
-> SELECT p.cedula, p.nombre, GROUP_CONCAT(tes.NumTestimonio) AS IDtestimonio
-> FROM Persona p
-> JOIN Testigo t ON p.cedula = t.cedula
-> JOIN Testimonio tes ON t.cedula = tes.cedulaTestigo
-> GROUP BY p.nombre;
Query OK, 0 rows affected (0.025 sec)
```

INSERCIÓN DE DATOS EN ABOGADOS:

Así como hicimos en el ejercicio de aerolínea lo primero que haremos será ingresar datos en la tabla persona y luego los dividiremos en sus respectivos subtipos (En el caso de estas vistas para probarlas hay que ingresar datos a la tabla “Acusador” y a la tabla “Testigo”).

```
MariaDB [Abogados]> INSERT INTO Persona (Cedula, Nombre, FechaNac, Edad, Direccion, Genero) VALUES
-> (1, "Paco Gonzalez", '1990-05-15', 34, "Calle 123", "M"),
-> (2, "Ana maria", '1995-09-20', 29, "Avenida Principal", "F"),
-> (3, "Luisa Serrano", '1985-03-10', 39, "Carrera 45", "F"),
-> (4, "Steven Acevedo", '2000-11-03', 23, "Calle 67", "M"),
-> (5, "Nicolas Velasco", '1978-07-28', 46, "Avenida Central", "M"),
-> (6, "Sirly Catherine", '1992-12-12', 32, "Carrera 89", "F"),
-> (7, "Daniel Pinzon", '2000-01-12', 24, "Calle 212", "M"),
-> (8, "Valentina Fiayo", '1999-12-23', 24, "Calle w4", "F");
Query OK, 8 rows affected (0.003 sec)
Records: 8 Duplicates: 0 Warnings: 0
```

Inserción tabla “Acusadores”:

Los datos primeros 5 datos (cedula 1,2,3,4,5) van a ser los demandantes – tabla “Acusador”

```
MariaDB [Abogados]> INSERT INTO Acusador (Cedula, MotivoDemanda, FechaOcurrido, DañosReclamados, AbogadoFiscal) VALUES
-> (1, "Violencia Fisica", '2024-02-23', "Psicologicos", 9),
-> (2, "Pension Alimentaria", '2023-12-22', "Alimenticio", 9),
-> (3, "Violencia Fisica", '2024-01-02', "Daño Físico", 9),
-> (4, "Violencia Fisica", '2023-09-02', "Daño Físico", 9),
-> (5, "Racismo", '2024-02-23', "Discriminacion", 9);
Query OK, 5 rows affected (0.020 sec)
Records: 5 Duplicates: 0 Warnings: 0
```


Inserción tabla "Testigo":

Los datos con cedula 6,7,8 en persona irán a el subtipo "Testigo"

```
MariaDB [Abogados]> INSERT INTO testigo (Cedula, RelacionVictima, RelacionVictimario) VALUES
-> (5, "Amigo", "Desconocido"),
-> (6, "Desconocido", "Desconocido"),
-> (7, "Desconocido", "Amigo");
Query OK, 3 rows affected (0.003 sec)
Records: 3 Duplicates: 0 Warnings: 0
```

Inserción tabla "Testimonio":

De los 3 testigo que creamos, vamos a ingresar dos testimonios al "6", uno al "7" y ninguno al 8.

```
MariaDB [Abogados]> INSERT INTO testimonio (NumTestimonio, CedulaTestigo, TipoTestimonio) VALUES
-> (1, 6, "El dia 22 de diciembre a las 5 pm vi como el acusado golpeo a el acusador"),
-> (2, 6, "El dia 22 de diciembre a las 6 pm el acusador estaba sangrando por lo golpes"),
-> (3, 7, "El dia 23 de agosto en el parque san pio presencie los insultos racistas");
Query OK, 3 rows affected (0.021 sec)
Records: 3 Duplicates: 0 Warnings: 0
```

PRUEBA DE LAS VISTAS:

Vista 1: En la vista 1 nos deberá arrojar los acusadores cuyo motivo de demanda sea "Violencia Física". Dentro de la tabla "Acusador" ingresamos 5 datos de los cuales 3 fueron de "Violencia Física" por lo tanto la vista nos debería devolver esos 3 datos.

```
MariaDB [Abogados]> SELECT * FROM Acusador_ViolenciaFisica;
+-----+-----+-----+-----+-----+
| Cedula | MotivoDemanda | FechaOcurrido | DañosReclamados | AbogadoFiscal |
+-----+-----+-----+-----+-----+
| 1      | Violencia Fisica | 2024-02-23    | Psicologicos     | 9              |
| 3      | Violencia Fisica | 2024-01-02    | Daño Físico      | 9              |
| 4      | Violencia Fisica | 2023-09-02    | Daño Físico      | 9              |
+-----+-----+-----+-----+-----+
3 rows in set (0.024 sec)
```

Podemos ver que efectivamente me devuelve únicamente los 3 datos mencionados;

Vista 2: Dentro de la vista 2 nos debería devolver tanto el testigo junto al número (ID) asignado a su testimonio (puede ser mas de 1).

```
MariaDB [Abogados]> SELECT * FROM Testigo_Acusatorio;
+-----+-----+-----+
| cedula | nombre          | IDtestimonio |
+-----+-----+-----+
| 7      | Daniel Pinzon   | 3            |
| 6      | Sirly Catherine | 1,2          |
+-----+-----+-----+
2 rows in set (0.025 sec)
```

Podemos ver que efectivamente se recopila el testigo junto a su testimonio, si el testigo no ha generado mínimo 1 testimonio no se recopila en la vista

VISTAS EJERCICIO JUGUETERIA

Vista 1: Dentro de mi base de datos establecí que 1 ingeniero puede administrar 1 o MUCHAS fábricas. Dentro de las fabricas existe un atributo “Fabricas” de tipo numérico. Por lo que recopilaremos en una vista los Ingenieros encargados de fabricas con 100 o mas maquinas. Recopilando el nombre del ingeniero, el id de la fabrica y el numero total de las maquinas.

```
MariaDB [Jugueteria]> CREATE VIEW Ingenieros_Fabricas_100Maquinas AS
-> SELECT p.nombre AS Ingeniero, GROUP_CONCAT(f.IDfabrica) AS IDfabrica, GROUP_CONCAT(f.Maquinas) AS NumeroMaquinas
-> FROM Persona p
-> JOIN Ingeniero i ON p.cedula = i.cedula
-> JOIN Fabrica f ON i.cedula = f.ingeniero
-> WHERE Maquinas >= 100
-> GROUP BY p.nombre;
```

Vista 2: Por último, realizaremos una vista que recopile los juguetes cuyo publico principal sea “Juguete”. Como queremos recopilar el ID y el nombre (tipo) usaremos un case

```
MariaDB [Jugueteria]> CREATE VIEW Juguetes_PublicoInfantil AS
-> SELECT j.IDjuguete, j.publicoObjetivo,
-> CASE
-> WHEN m.IDjuguete IS NOT NULL THEN "Mecánico"
-> WHEN d.IDjuguete IS NOT NULL THEN "Didáctico"
-> WHEN di.IDjuguete IS NOT NULL THEN "Digital"
-> END AS tipo_Juguete
-> FROM juguete j
-> LEFT JOIN mecanico m ON j.IDjuguete = m.IDjuguete
-> LEFT JOIN didactico d ON j.IDjuguete = d.IDjuguete
-> LEFT JOIN digital di ON j.IDjuguete = di.IDjuguete
-> WHERE j.publicoObjetivo = "infantil";
Query OK, 0 rows affected (0.035 sec)
```

(El CASE será usado para saber si que categoría es el Juguete)

INSERCIÓN DE DATOS EN JUGUETERIA:

Para probar las 2 vistas de esta B.D necesitamos crear personas y luego definirlos como ingenieros.

```
MariaDB [Jugueteria]> INSERT INTO Persona (Cedula, Nombre, FechaNac, Edad, Direccion, Genero) VALUES
-> (1, "Mariana Peña", '1990-05-15', 34, "Calle 123", "F"),
-> (2, "Mayra Perez", '1995-09-20', 29, "Avenida Principal", "F"),
-> (3, "Oscar pinto", '1985-03-10', 39, "Carrera 45", "M");
Query OK, 3 rows affected (0.021 sec)
Records: 3 Duplicates: 0 Warnings: 0
```

Inserción tabla “Ingeniero”:

```
MariaDB [Jugueteria]> INSERT INTO Ingeniero (Cedula, AñosExperiencia, Especialidad) VALUES
-> (1, 10, "Mecatronica"),
-> (2, 5, "Sistemas"),
-> (3, 2, "Sistemas");
Query OK, 3 rows affected (0.021 sec)
Records: 3 Duplicates: 0 Warnings: 0
```

Inserción tabla “Fabrica”:

A cada ingeniero le daremos 3 fábricas, pero lo importante será el #de maquinas ya que en la vista solo se recopilará a los ingenieros encargados de fábricas con +100 fábricas.

```
MariaDB [Jugueteria]> INSERT INTO Fabrica (IDfabrica, Ingeniero, Maquinas, Direccion) VALUES
-> (1001,1, 23, "Calle 12"),
-> (1002,1, 84, "Calle 34"),
-> (1003,1, 100, "Calle 34"),
-> (1004,2, 101, "Carrera 17"),
-> (1005,2, 112, "Calle 32"),
-> (1006,2, 84, "Calle 78"),
-> (1007,3, 200, "Calle 34"),
-> (1008,3, 120, "Calle 09"),
-> (1009,3, 105, "Calle 112");
Query OK, 9 rows affected (0.006 sec)
Records: 9 Duplicates: 0 Warnings: 0
```

Inserción tabla “Juguete”:

```
MariaDB [Jugueteria]> INSERT INTO Juguete (IDjuguete, Arquitecto, IDfabrica, Peso, Precio, PublicoObjetivo)
-> VALUES
-> (01, 4, 1001, 200, 20000, "Infantil"),
-> (02, 5, 1002, 150, 15000, "Infantil"),
-> (03, 6, 1003, 180, 18000, "Juvenil"),
-> (04, 4, 1004, 220, 22000, "Infantil"),
-> (05, 5, 1005, 190, 19000, "Infantil"),
-> (06, 6, 1006, 210, 21000, "Juvenil"),
-> (07, 4, 1007, 240, 24000, "Infantil"),
-> (08, 5, 1008, 170, 17000, "Juvenil"),
-> (09, 6, 1009, 230, 23000, "Infantil");
Query OK, 9 rows affected (0.020 sec)
Records: 9 Duplicates: 0 Warnings: 0
```

De estos 9 juguetes creados, vamos a dividirlos su ingreso en los subtipos así: 3 para mecánico, 3 para digital y 3 didáctico.

Inserción tabla "Mecanico":

```
MariaDB [Jugueteria]> INSERT INTO Mecanico (IDjuguete, DuracionBateria) VALUES  
-> (01, 50),  
-> (02, 35),  
-> (03, 45);
```

Inserción tabla "Didactico":

```
MariaDB [Jugueteria]> INSERT INTO Didactico (IDjuguete, AreaDidactica) VALUES  
-> (04, "Matematicas"),  
-> (05, "Matematicas"),  
-> (06, "Ciencias");
```

Inserción tabla "Digital":

```
MariaDB [Jugueteria]> INSERT INTO Digital (IDjuguete, Plataforma) VALUES  
-> (07, "PC"),  
-> (08, "Celular"),  
-> (09, "Celular");
```

PRUEBA DE LAS VISTAS:

Vista 1: En la vista 1 nos deberá mostrar los ingenieros que tengan alguna fabrica a cargo y que esta tenga 100 o más maquinas.

```
MariaDB [Jugueteria]> SELECT * FROM Ingenieros_Fabricas_100Maquinas;  
+-----+-----+-----+  
| Ingeniero | IDfabrica | NumeroMaquinas |  
+-----+-----+-----+  
| Mariana Peña | 1003 | 100 |  
| Mayra Perez | 1004,1005 | 101,112 |  
| Oscar pinto | 1007,1008,1009 | 200,120,105 |  
+-----+-----+-----+  
3 rows in set (0.004 sec)
```

Vista 2: En la vista 2 nos mostrara todos los juguetes para publico infantil y su tipo.

```
MariaDB [Jugueteria]> SELECT * FROM Juguetes_PublicoInfantil;  
+-----+-----+-----+  
| IDjuguete | publicoObjetivo | tipo_Juguete |  
+-----+-----+-----+  
| 1 | Infantil | Mecánico |  
| 2 | Infantil | Mecánico |  
| 4 | Infantil | Didáctico |  
| 5 | Infantil | Didáctico |  
| 7 | Infantil | Digital |  
| 9 | Infantil | Digital |  
+-----+-----+-----+
```

PARTE 2: CONSULTAS ANIDADAS

CONSULTAS EJERCICIO CASA SOFTWARE

1.Consulta escalar: Se requiere obtener el proyecto con el presupuesto más alto.

Datos actuales:

```
MariaDB [CasaSoftware]> select * from proyecto;
```

codProyecto	Nombre	Descripcion	Cliente	Presupuesto	numHoras	fechaInicio
1	Agropecuaria	Gestion productos agricolas	Maicito	1000000	150	2024-04-1
2	B.D_Financiera	Base datos area financiera	Banco Republica	15000000	350	2024-04-2
3	B.D_Escolar	Base de datos escolares	Divino Amor	3500000	90	2021-04-2
4	VentaOnline	Software venta Online	ZonaModa	2000000	160	2021-05-0
5	EnviosRapidisimo	Software gestion envios online	Rappli	8000000	290	2019-08-1
6	DenunciasUrbanas	Software realizacion denuncias	Alcaldia	30000000	500	2019-08-2

Consulta:

```
MariaDB [CasaSoftware]> SELECT *  
-> FROM Proyecto  
-> WHERE Presupuesto =  
-> (SELECT MAX(Presupuesto)  
-> FROM Proyecto);
```

codProyecto	Nombre	Descripcion	Cliente	Presupuesto	numHoras	fechaInicio
6	DenunciasUrbanas	Software realizacion denuncias	Alcaldia	30000000	500	2019-08-29

1 row in set (0.001 sec)

2.Consulta Pertenencia: Se requiere conocer el nombre de los empleados y su tipo que no están trabajando en ningún proyecto en este momento

Datos actuales:

```
MariaDB [CasaSoftware]> select * from Empleado;
```

Cedula	CodigoEmpleado	Nombre	Direccion	Titulacion	Años	tipo
1005	1	Daniel Jay	Calle 123	Ingeniero	20	Programador
1006	2	Yeimy Corzo	Avenida 52	Ingeniero	20	Programador
1007	3	Pedro Rodríguez	Carrera 456	Ingeniero	25	Programador
1008	4	Juan Lopez	Calle 234	Analista	32	Analista
1009	5	Julio Gonzalez	Carrera 48	Analista	29	Analista
1010	6	Valentina fiayo	Calle 13u	Analista	29	Analista
1011	7	Katherin Pinzon	Calle 52	Analista	23	Analista
1012	8	Oscar Pinto	Carrera 15	Ingeniero	54	Programador
1013	9	William Reyes	Carrera 34	Analista	18	Analista

```
MariaDB [CasaSoftware]> select * from Proyecto_Empleado;
```

CodProyecto	Cedula	numHoras	Costo	Jefe_proyecto
1	1005	12	100000	Gustavo
1	1006	34	100000	Gustavo
2	1005	68	1500000	Maria
3	1005	34	350000	Juan
4	1006	23	145000	Paco
4	1007	67	100000	Paco
4	1009	34	134500	Paco
5	1007	67	130000	Pedro
6	1008	50	150000	Julio
6	1009	50	150000	Julio
6	1011	56	150000	Julio

Dentro de mi base de datos tengo asignado todos los empleados a algún proyecto menos a los que tienen cedula 1010,1012 y 1013.

Consulta:

```
MariaDB [CasaSoftware]> SELECT e.Cedula, e.Nombre, e.tipo
-> FROM Empleado e
-> WHERE e.Nombre NOT IN (
->     SELECT e.nombre
->     FROM Empleado e
->     JOIN Proyecto_Empleado pe ON pe.cedula = e.cedula
-> );
```

Cedula	Nombre	tipo
1010	Valentina fiayo	Analista
1012	Oscar Pinto	Programador
1013	William Reyes	Analista

3.Consulta ALL: Se requiere obtener todos los datos de los proyectos en los cuales participan únicamente empleados tipo “Analista”

Los empleados tipo programador son los que poseen la cedula 1005,1006,1007 y 1012

Datos actuales:

```
MariaDB [CasaSoftware]> select * from proyecto_empleado;
```

CodProyecto	Cedula	numHoras	Costo	Jefe_proyecto
1	1005	12	100000	Gustavo
1	1006	34	100000	Gustavo
2	1005	68	1500000	Maria
3	1005	34	350000	Juan
4	1006	23	145000	Paco
4	1007	67	100000	Paco
4	1009	34	134500	Paco
5	1007	67	130000	Pedro
6	1008	50	150000	Julio
6	1009	50	150000	Julio
6	1011	56	150000	Julio

Podemos apreciar que el codProyecto es el único proyecto que hasta el momento se le han asignado solo trabajadores tipo “Analista” (cedula 1008,1009 y 1011 son analistas) por lo que la consulta deberá devolver solo ese proyecto.

Consulta:

```
MariaDB [CasaSoftware]> SELECT *
-> FROM Proyecto p
-> WHERE "Analista" = ALL (
-> SELECT e.tipo
-> FROM Proyecto_Empleado pe
-> JOIN Empleado e ON pe.Cedula = e.Cedula
-> WHERE pe.CodProyecto = p.codProyecto
-> );
```

codProyecto	Nombre	Descripcion	Cliente	Presupuesto	numHoras	fechaInicio
6	DenunciasUrbanas	Software realizacion denuncias	Alcaldia	30000000	500	2019-08-29

4.Consulta ANY: Se requiere obtener todos los proyectos en los que participa al menos 1 programador

Un poco parecida a la consulta anterior, ahora debemos buscar los proyectos que tengan al menos 1 programador. Por lo que acá no importara si el proyecto también posee analistas.

```
MariaDB [CasaSoftware]> SELECT *
-> FROM Proyecto p
-> WHERE "Programador" = ANY (
-> SELECT e.tipo
-> FROM Proyecto_Empleado pe
-> JOIN Empleado e ON pe.Cedula = e.Cedula
-> WHERE pe.CodProyecto = p.codProyecto
-> );
```

codProyecto	Nombre	Descripcion	Cliente	Presupuesto	numHoras	fechaInicio
1	Agropecuaria	Gestion productos agricolas	Maicito	1000000	150	2024-04-1
2	B.D_Financiera	Base datos area financiera	Banco Republica	15000000	350	2024-04-2
3	B.D_Escolar	Base de datos escolares	Divino Amor	3500000	90	2021-04-2
4	VentaOnline	Software venta Online	ZonaModa	2000000	160	2021-05-0
5	EnviosRapidisimo	Software gestion envios online	Rappli	8000000	290	2019-08-1

5.Consulta EXISTS: Seleccionar todos los Prototipos si existe al menos algún prototipo supera la versión 2.43.

Lo primero que haremos es crear algunos productos y luego insertarlos en la tabla prototipo, vamos a imaginar algo un poco fantasioso y vamos a hacer un proyecto (8) donde se va a desarrollar un robot .

```
MariaDB [CasaSoftware]> INSERT INTO producto (Codigo, Nombre, Descripcion, Estado, Analista, Fase, Proyecto, Tipo)
-> VALUES
-> (2001, "Mano Derecha", "Mano Derecha del robot", "En proceso", 1008, 1, 8, "Prototipos"),
-> (2002, "Mano Izquierda", "Mano Izquierda del robot", "En proceso", 1008, 1, 8, "Prototipos"),
-> (2003, "Cabeza", "Pie izquierdo robot", "En proceso", 1009, 2, 8, "Prototipos"),
-> (2004, "Pie derecho", "Pie Derecho robot", "En proceso", 1009, 2, 8, "Prototipos"),
-> (2005, "Cabeza", "Cabeza robot", "En proceso", 1011, 3, 8, "Prototipos");
```

```
MariaDB [CasaSoftware]> insert into Prototipos (Codigo, Version, Ubicacion) VALUES
-> (2001, 1.34, "Bucaramanga"),
-> (2002, 0.21, "Bucaramanga"),
-> (2003, 2.42, "Bucaramanga");
```

Ya que "EXISTS" es una consulta de tipo booleano primero realizaremos la consulta cuando ningún prototipo supera la versión 2.43.

```
MariaDB [CasaSoftware]> SELECT *
-> FROM Prototipos
-> WHERE EXISTS (
-> SELECT *
-> FROM Prototipos
-> WHERE Version > 2.43
-> );
Empty set (0.000 sec)
```

Ahora haremos la prueba con un prototipo que supere la versión 2.43

```
MariaDB [CasaSoftware]> insert into Prototipos (Codigo, Version, Ubicacion) VALUES
-> (2004, 3.04, "Bucaramanga");
Query OK, 1 row affected (0.020 sec)

MariaDB [CasaSoftware]> SELECT *
-> FROM Prototipos
-> WHERE EXISTS (
-> SELECT *
-> FROM Prototipos
-> WHERE Version > 2.43
-> );
+-----+-----+-----+
| Codigo | Version | Ubicacion |
+-----+-----+-----+
| 2001 | 1.34 | Bucaramanga |
| 2002 | 0.21 | Bucaramanga |
| 2003 | 2.42 | Bucaramanga |
| 2004 | 3.04 | Bucaramanga |
+-----+-----+-----+
4 rows in set (0.001 sec)
```

En este caso primero insertamos un prototipo con una versión mayor a la 2.43 (en este caso 3.04) por lo que ahora se cumple la condición y nos devuelve todos los prototipos registrados.

6.Consulta From: Se requiere saber el promedio de productos realizados por fase en el proyecto 8

Datos actuales tabla fase:

```
MariaDB [CasaSoftware]> select * from fase;
```

numFase	Nombre	Estado	fechaInicio	fechaFIN	Proyecto
1	Desarrollo Manos	En proceso	2024-03-23	NULL	8
2	Desarrollo Pies	En proceso	2024-04-01	NULL	8
3	Desarrollo cabeza	En proceso	2024-04-21	NULL	8

Datos actuales tabla Producto:

```
MariaDB [CasaSoftware]> select * from producto;
```

Codigo	Nombre	Descripcion	Estado	Analista	Fase	Proyecto	Tipo
2001	Mano Derecha	Mano Derecha del robot	En proceso	1008	1	8	Prototipos
2002	Mano Izquierda	Mano Izqueirda del robot	En proceso	1008	1	8	Prototipos
2003	Cabeza	Pie izquierdo robot	En proceso	1009	2	8	Prototipos
2004	Pie derecho	Pie Derecho robot	En proceso	1009	2	8	Prototipos
2005	Cabeza	Cabeza robot	En proceso	1011	3	8	Prototipos

Consulta:

```
MariaDB [CasaSoftware]> SELECT AVG(TotalProductos) AS Promedio_Productos_Fase
-> FROM (
->   SELECT Fase, COUNT(*) AS TotalProductos
->   FROM Producto
->   WHERE Proyecto = 8
->   GROUP BY Fase
-> ) AS Productos_Fase;
```

Promedio_Productos_Fase
1.6667

1 row in set (0.001 sec)

Como podemos ver en promedio se desarrollan 1.6 Productos por fase en el Proyecto No.8 "Creación de un robot".

CONSULTAS EJERCICIO AEREOLINEA

Consulta escalar: Se requiere conocer el total de aeropuerto que tienen el mismo número de puertas de embarque que el aeropuerto con más Puertas de embarque.

Datos actuales tabla Aereopuerto:

IATA	Nombre	Direccion	puertas_embarque
1123	Olaya herrera	Dirección 234	8
2743	Marcos rueda	Calle 324	3
3413	Kennedy	Carrera 34	2
3421	San Juan	Calle 23	3
3458	Embrujo	Carrera 765	8
4145	Dorado	Calle 52	3
4325	Palo negro	Calle 532	4
8766	Rojas pinilla	Dirección 5	2

En este momento el máximo de Puertas de embarque es 8 y hay dos aeropuertos que poseen 8 Puertas de embarque.

```
MariaDB [Aereolinea]> SELECT COUNT(*) AS Total_Aeropuertos
-> FROM Aereopuerto
-> WHERE Puertas_embarque = (
->     SELECT MAX(Puertas_embarque)
->     FROM Aereopuerto
-> );
```

Total_Aeropuertos
2

1 row in set (0.001 sec)

Como podemos ver efectivamente nos muestra que hay dos aeropuertos cuyo número de puertas de embarque es igual al máximo de puertas de embarque que tienen los aeropuertos.

Consulta Pertenencia: Se requiere conocer todas las terminales que operan en el Aeropuerto “Dorado”.

Datos actuales tabla terminal_aeropuerto:

```
MariaDB [Aereolinea]> select * from terminal_aeropuerto;
```

IDterminal	IATAaeropuerto
1	4145
2	4145
3	2743
4	8766
5	4145

El aeropuerto “Dorado” tiene IATA 4145 por lo que por ahora hay 3 terminales registradas ahí, vamos a ver si nos devuelve las 3:

```
MariaDB [Aereolinea]> SELECT *  
-> FROM Terminal  
-> WHERE IDterminal IN (  
->     SELECT IDterminal  
->     FROM Terminal_Aeropuerto ta  
->     JOIN Aeropuerto a ON ta.IATAaeropuerto = a.IATA  
->     WHERE a.Nombre = "Dorado"  
-> );
```

IDterminal	Nombre	Políticas	RutasActivas
1	Avianca	Revisión de seguridad obligatoria antes del embarque	8
2	Purina	Servicio de WiFi gratuito disponible en toda la terminal	16
5	Nariño	Los menores de 10 años no pagan	9

Como podemos ver se nos devuelven los datos correspondientes a el aeropuerto “Dorado” con la ayuda del comando “IN” y especificando que el nombre de la relación sea “Dorado”. También podríamos haberlo hecho con el ID del aeropuerto (4145).

Consulta ALL: Seleccionar todos los aeropuertos que posean todas sus aerolíneas con al menos 10 rutas activas.

Para la realización de este ejercicio ampliaremos los registros de terminales y de terminal_aeropuerto.

Datos actuales tabla terminal:

```
MariaDB [Aereolinea]> select * from terminal;
```

IDterminal	Nombre	Políticas	RutasActivas
1	Avianca	Revisión de seguridad obligatoria antes del embarque	8
2	Puringa	Servicio de WiFi gratuito disponible en toda la terminal	16
3	Dubai	Control de pasaportes requerido para vuelos internacionales	12
4	Galan	Devolucion en caso de no cumplir expectativas	6
5	Nariño	Los menores de 10 años no pagan	9
6	Juan Jose	Prohibido fumar en todas las áreas del terminal	13
7	ViajaRapido	Descuento del 50% en alimentos para pasajeros con vuelos retrasados	8
8	Narita	Entrega gratuita de kits de viaje a pasajeros de larga distancia	11
9	Changi	Área de juego para niños disponible en la sala de espera	9
10	Barajas	Servicio de carritos de equipaje gratuito disponible en la terminal	10

Datos actuales tabla terminal_aeropuerto:

```
MariaDB [Aereolinea]> select * from terminal_aeropuerto;
```

IDterminal	IATAaeropuerto
1	4145
2	4145
2	4325
3	2743
4	8766
5	4145
6	1123
7	3413
7	3421
8	1123
9	2743
10	3458
10	8766

13 rows in set (0.001 sec)

Consulta:

```
MariaDB [Aereolinea]> SELECT *
-> FROM Aereopuerto a
-> WHERE 10 <= ALL (
->   SELECT t.RutasActivas
->   FROM Terminal t
->   JOIN Terminal_Aeropuerto te ON te.IDterminal = t.IDterminal
->   WHERE a.IATA = te.IATAaeropuerto
-> );
```

IATA	Nombre	Direccion	puertas_embarque
1123	Olaya herrera	Dirección 234	8
3458	Embrujo	Carrera 765	8
4325	Palo negro	Calle 532	4

Aquí podemos apreciar los aeropuertos con terminales con más de 10 rutas activas.

Consulta ANY: Se requiere obtener los aeropuertos que tienen al menos 8 torres de control

Datos actuales tabla TorreControl:

IDtorre	IATAaeropuerto	sistemacomunicacion
1	1123	VHF
2	2743	UHF
3	3413	HF
4	3421	VHF
5	3458	UHF
6	4145	HF
7	4325	VHF
8	8766	UHF
9	1123	HF
10	2743	VHF
11	3413	UHF
12	3421	HF
13	3458	VHF
14	4145	UHF
15	4325	HF
16	8766	VHF
17	1123	UHF
18	2743	HF
19	3413	VHF
20	3421	UHF
21	3458	HF
22	4145	VHF
23	4325	UHF
24	8766	HF
25	1123	VHF
26	1123	VHF

Consulta:

```
MariaDB [Aerolinea]> SELECT *  
-> FROM Aeropuerto  
-> WHERE IATA = ANY (  
->     SELECT IATAaeropuerto  
->     FROM TorreControl  
->     GROUP BY IATAaeropuerto  
->     HAVING COUNT(*) >= 5  
-> );
```

IATA	Nombre	Direccion	puertas_embarque
1123	Olaya herrera	Dirección 234	8

1 row in set (0.001 sec)

Podemos ver que solo hay un aeropuerto que solo cuenta con 5 o más Torres de control, específicamente las torres con ID 1,9,17,25,26.

Consulta EXISTS: Consultar todos los aviones, solo si existen al menos 1 avión de cada tipo dentro de la base de datos.

Datos actuales tabla Helicóptero:

```
MariaDB [Aereolinea]> select * from helicóptero;
```

IDavion	PesoMaximo
1004	235
1008	234
1010	452
1014	243

Datos actuales tabla aeropuerto_avion:

```
MariaDB [Aereolinea]> select * from aeropuerto_avion;
```

IATAaeropuerto	IDavion
1123	1001
1123	1010
2743	1002
2743	1011
3413	1003
3413	1004
3413	1012
3421	1005
3421	1013
3458	1006
3458	1014
4145	1007
4325	1008
8766	1009

14 rows in set (0.001 sec)

Consulta:

```
MariaDB [Aereolinea]> SELECT *
-> FROM Aeropuerto A
-> WHERE EXISTS (
-> SELECT *
-> FROM Aeropuerto_Avion AA
-> JOIN Helicoptero H ON AA.IDavion = H.IDavion
-> WHERE A.IATA = AA.IATAaeropuerto);
```

IATA	Nombre	Direccion	puertas_embarque
1123	Olaya herrera	Dirección 234	8
3413	Kennedy	Carrera 34	2
3458	Embrujo	Carrera 765	8
4325	Palo negro	Calle 532	4

Existen 4 aeropuertos con helicópteros registrados

Consulta From: Se quiere obtener el recepcionista que menos personas ha atendido.

Gracias a que en la primera parte de este taller creamos una vista Recepcionista_Pasajero, podemos ver los pasajeros atendidos por recepcionista

Datos actuales vista recepcionsita_pasajero:

```
MariaDB [Aereolinea]> select * from recepcionista_pasajero;
```

nombre_recepcionista	pasajeros_atendidos
Ana	Laura, Daniel
Pedro	Juanita

Ana es la recepcionista que más pasajeros atendió hasta ahora en los registros entonces nuestra consulta nos debe devolver a Ana + el número de pasajeros atendidos.

Consulta:

```
MariaDB [Aereolinea]> SELECT p.Nombre AS Recepcionista, MAX(NumPasajeros) AS Pasajeros_atendidos
-> FROM (
->   SELECT CedulaRecepcionista, COUNT(*) AS NumPasajeros
->   FROM Pasajero
->   GROUP BY CedulaRecepcionista
-> ) AS totalPasajeros
-> JOIN Recepcionista r ON totalPasajeros.CedulaRecepcionista = r.Cedula
-> JOIN Persona p ON r.Cedula = p.Cedula;
```

Recepcionista	Pasajeros_atendidos
Ana	2

Podemos ver que efectivamente nos devolvió el dato de Ana como la recepcionista que mas personas ha atendido hasta ahora en el registro actual.

CONSULTAS EJERCICIO ABOGADOS

Consulta escalar: Se requiere conocer todos los abogados cuyos honorarios sean iguales al máximo de los honorarios registrados en la base de datos.

Datos actuales Tabla abogado:

```
MariaDB [Abogados]> select * from abogado;
```

Cedula	Honarios	CasosGanados	AñosExperiencia	Maestria
1	450000	5	3	Derecho
2	520000	7	4	Derecho
3	480000	6	5	Derecho
4	510000	8	6	Derecho
5	490000	7	4	Derecho
6	470000	6	3	Derecho
7	540000	9	5	Derecho
8	530000	8	4	Derecho
9	600000	10	5	Derecho
10	600000	4	2	Derecho

Consulta:

```
MariaDB [Abogados]> SELECT *  
-> FROM Abogado  
-> WHERE Honarios = (  
->     SELECT MAX(Honarios)  
->     FROM Abogado);
```

Cedula	Honarios	CasosGanados	AñosExperiencia	Maestria
9	600000	10	5	Derecho
10	600000	4	2	Derecho

Los abogados con cedula 9 y 10 son los que tienen mayores honorarios.

Consulta Pertenencia: Se requiere conocer todos los testigos que no le dieron su testimonio al frente del juez “Lopez Obrador”.

Datos actuales Tabla juez_testigo:

```
MariaDB [Abogados]> select * from juez_testigo;
+-----+-----+
| CedulaJuez | CedulaTestigo |
+-----+-----+
|          10 |             5 |
|          11 |             6 |
|          11 |             7 |
+-----+-----+
3 rows in set (0.000 sec)
```

NOTA: Lopez obrador es el juez con cedula 11.

Consulta:

Aquí haremos algo curioso ya que combinaremos la consulta de pertinencia con una consulta escalar, esto debido a que el nombre lopez obrador se guarda en la tabla persona y no en la de juez como tal creo que es la forma mas fácil de resolver el ejercicio.

```
MariaDB [Abogados]> SELECT *
-> FROM Testigo
-> WHERE Cedula NOT IN (
->     SELECT CedulaTestigo
->     FROM Juez_Testigo
->     WHERE CedulaJuez = (
->         SELECT Cedula
->         FROM Persona
->         WHERE Nombre = "Lopez Obrador"
->     )
-> );
+-----+-----+-----+
| Cedula | RelacionVictima | RelacionVictimario |
+-----+-----+-----+
|       5 | Amigo           | Desconocido        |
+-----+-----+-----+
1 row in set (0.001 sec)
```

Podemos ver que efectivamente nos muestra el único testigo que no ha testificado frente al juez “Lopez obrador”

Consulta ALL: Seleccionar los abogados cuyos años de experiencia sean mayores a el promedio de años de experiencia de todos los abogados.

Consulta:

```
MariaDB [Abogados]> SELECT *
-> FROM Abogado
-> WHERE AñosExperiencia > ALL (
->     SELECT AVG(AñosExperiencia)
->     FROM Abogado
-> );
```

Cedula	Honarios	CasosGanados	AñosExperiencia	Maestria
3	480000	6	5	Derecho
4	510000	8	6	Derecho
7	540000	9	5	Derecho
9	600000	10	5	Derecho

4 rows in set (0.021 sec)

Consulta ANY: Se requiere seleccionar todos los abogados que tengan mas años de experiencia que al menos un juez.

Consulta:

```
MariaDB [Abogados]> SELECT *
-> FROM Abogado
-> WHERE AñosExperiencia > ANY (
->     SELECT AñosExperiencia
->     FROM Juez
-> );
```

Cedula	Honarios	CasosGanados	AñosExperiencia	Maestria
12	550000	10	12	Derecho

Consulta EXISTS: Consultar todos los jueces que tengan al menos 1 veredicto asociado

Datos actuales Tabla Testimonio:

```
MariaDB [Abogados]> select * from veredicto;
```

IDveredicto	CedulaJuez	Evidencia	Conclusion
777	10	Fotografías	Culpable

En este momento solo hay 1 testimonio asociado al juez con cedula 10, entonces nos debe devolver ese juez.

Consulta:

```
MariaDB [Abogados]> SELECT *  
-> FROM Juez j  
-> WHERE EXISTS (  
->     SELECT *  
->     FROM Veredicto v  
->     WHERE v.CedulaJuez = j.Cedula  
-> );
```

Cedula	AñosExperiencia	Especializacion
10	10	Derecho

1 row in set (0.001 sec)

Efectivamente nos devolvió el juez que ya tiene un veredicto asociado.

Consulta From: Calcular el promedio de los honorarios de todos los abogados registrados en la base de datos.

Datos actuales Tabla Abogado:

```
MariaDB [Abogados]> select * from abogado;
```

Cedula	Honarios	CasosGanados	AñosExperiencia	Maestria
1	450000	5	3	Derecho
2	520000	7	4	Derecho
3	480000	6	5	Derecho
4	510000	8	6	Derecho
5	490000	7	4	Derecho
6	470000	6	3	Derecho
7	540000	9	5	Derecho
8	530000	8	4	Derecho
9	600000	10	5	Derecho
10	600000	4	2	Derecho
12	550000	10	12	Derecho

11 rows in set (0.001 sec)

Consulta:

```
MariaDB [Abogados]> SELECT AVG(Sumatoria) AS PromedioSalarios
-> FROM (
->     SELECT SUM(Honarios) AS sumatoria
->     FROM Abogado
->     GROUP BY Cedula
-> ) AS SalariosAbogados;
```

PromedioSalarios
521818.1818

El promedio de los honorarios de los abogados es de 521.818 pesos.

CONSULTAS EJERCICIO JUGUETERIA

Consulta escalar: Se requiere obtener todos los juguetes que tienen precios iguales a el menor de los precios de los juguetes.

Datos actuales Tabla Juguete:

```
MariaDB [Jugueteria]> select * from Juguete
-> ;
```

IDjuguete	Arquitecto	IDfabrica	Peso	Precio	PublicoObjetivo
1	4	1001	200	20000	Infantil
2	5	1002	150	15000	Infantil
3	6	1003	180	18000	Juvenil
4	4	1004	220	22000	Infantil
5	5	1005	190	19000	Infantil
6	6	1006	210	21000	Juvenil
7	4	1007	240	24000	Infantil
8	5	1008	170	17000	Juvenil
9	6	1009	230	23000	Infantil

Consulta:

```
MariaDB [Jugueteria]> SELECT *
-> FROM Juguete
-> WHERE Precio = (
->     SELECT MIN(Precio)
->     FROM Juguete
-> );
```

IDjuguete	Arquitecto	IDfabrica	Peso	Precio	PublicoObjetivo
2	5	1002	150	15000	Infantil

El juguete con menor valor vale 15.000 pesos.

Consulta Pertenencia: Se quiere obtener todos los clientes registrados (afiliados) que no han realizado ninguna compra

En este momento tenemos 5 clientes afiliados en nuestra base de datos, de los cuales solo 2 han realizado compras, por lo que nos debería devolver la consulta 3 clientes.

```
MariaDB [Jugueteria]> select * from cliente;
```

Cedula	FormaPago	Tipo
7	Efectivo	Afiliado
8	Tarjeta	Afiliado
9	Efectivo	Afiliado
10	Tarjeta	Afiliado
11	Efectivo	Afiliado

```
5 rows in set (0.000 sec)
```



```
MariaDB [Jugueteria]> select * from afiliado;
```

Cedula	TarjetaAfiliacion
7	10671
8	10672
9	10673
10	10674
11	10675

```
MariaDB [Jugueteria]> select * from juguete_cliente;
```

IDjuguete	CedulaCliente
2	7
8	11

Consulta:

```
MariaDB [Jugueteria]> SELECT p.cedula as cedula_afiliado, p.nombre as nombre_afiliado, a.TarjetaAfiliacion
-> FROM Afiliado a
-> JOIN Persona p ON a.Cedula = p.Cedula
-> WHERE a.Cedula NOT IN (
->     SELECT CedulaCliente
->     FROM Juguete_Cliente
-> );
```

cedula_afiliado	nombre_afiliado	TarjetaAfiliacion
8	Juan Abril	10672
9	Juan rivera	10673
10	Nicolas garcia	10674

```
3 rows in set (0.001 sec)
```

Podemos ver que la consulta nos devuelve correctamente los afiliados que no han realizado ninguna compra.

Consulta ALL: Seleccionar el nombre, cedula y tipo de algún cliente que haya comprado solamente juguetes tipo “mecánico”.

Datos actuales Tabla Mecánico y tabla juguete_cliente:

```
MariaDB [Jugueteria]> select * from mecanico;
+-----+-----+
| IDjuguete | DuracionBateria |
+-----+-----+
| 1 | 50 |
| 2 | 35 |
| 3 | 45 |
+-----+-----+
3 rows in set (0.000 sec)

MariaDB [Jugueteria]> select * from juguete_cliente;
+-----+-----+
| IDjuguete | CedulaCliente |
+-----+-----+
| 2 | 7 |
| 3 | 7 |
| 4 | 10 |
| 6 | 12 |
| 8 | 11 |
+-----+-----+
```

Consulta:

```
MariaDB [Jugueteria]> SELECT p.Nombre AS NombreCliente, p.Cedula, c.tipo
-> FROM Cliente c
-> JOIN Persona p ON c.Cedula = p.Cedula
-> WHERE p.Cedula = ALL (
->     SELECT jc.CedulaCliente
->     FROM Juguete_Cliente jc
->     JOIN Mecanico m ON jc.IDjuguete = m.IDjuguete
-> );
+-----+-----+-----+
| NombreCliente | Cedula | tipo |
+-----+-----+-----+
| Karen sierra | 7 | Afiliado |
+-----+-----+-----+
1 row in set (0.001 sec)
```

Como podemos ver hay una cliente que ha comprado solo juguetes tipo “mecánico”.

Consulta ANY: Se requiere obtener el nombre y el tipo de los clientes que hayan comprado al menos un juguete didáctico.

Datos actuales tabla didáctico y tabla ventas juguete_cliente:

```
MariaDB [Jugueteria]> select * from didactico;
+-----+-----+
| IDjuguete | AreaDidactica |
+-----+-----+
|         4 | Matematicas   |
|         5 | Matematicas   |
|         6 | Ciencias      |
+-----+-----+
3 rows in set (0.000 sec)

MariaDB [Jugueteria]> select * from juguete_cliente;
+-----+-----+
| IDjuguete | CedulaCliente |
+-----+-----+
|         2 |              7 |
|         4 |             10 |
|         6 |             12 |
|         8 |             11 |
+-----+-----+
```

Consulta:

```
MariaDB [Jugueteria]> SELECT p.Nombre, c.Tipo
-> FROM Cliente c
-> JOIN Persona p ON c.Cedula = p.Cedula
-> WHERE c.Cedula = ANY (
->     SELECT jc.CedulaCliente
->     FROM Juguete_Cliente jc
->     JOIN Didactico d ON jc.IDjuguete = d.IDjuguete
-> );
+-----+-----+
| Nombre      | Tipo      |
+-----+-----+
| Nicolas garcia | Afiliado  |
| Maria jose   | Casual    |
+-----+-----+
2 rows in set (0.000 sec)
```

Como podemos ver nos muestra satisfactoriamente los clientes que han comprado juguetes didácticos.

Consulta EXISTS: Obtener todos los juguetes que han comprado al menos 1 juguetes, se desea ver el ID del juguete comprado, y los datos de su cliente (nombre, cedula y tipo).

Dato actuales tabla juguete_cliente:

```
MariaDB [Jugueteria]> select * from juguete_cliente;
```

IDjuguete	CedulaCliente
2	7
4	10
6	12
8	11

En este momento hay 4 clientes que han realizado alguna compra por lo que la consulta nos debe devolver 4 resultados.

Consulta:

```
MariaDB [Jugueteria]> SELECT j.IDjuguete AS ID_juguete_Comprado, p.Nombre AS Cliente, c.Cedula, c.tipo
-> FROM Juguete j
-> JOIN Juguete_Cliente jc ON j.IDjuguete = jc.IDjuguete
-> JOIN Cliente c ON jc.CedulaCliente = c.Cedula
-> JOIN Persona p ON p.Cedula = c.Cedula
-> WHERE EXISTS (
->   SELECT *
->   FROM Juguete_Cliente jc
->   WHERE jc.IDjuguete = j.IDjuguete
-> );
```

ID_juguete_Comprado	Cliente	Cedula	tipo
2	Karen sierra	7	Afiliado
4	Nicolas garcia	10	Afiliado
6	Maria jose	12	Casual
8	Maria Ardila	11	Afiliado

Como podemos ver los devuelve los datos de las personas que han comprado algún juguete y el ID del juguete comprado.

Consulta From: Se requiere conocer la clasificación de juguetes cuyo promedio del precio sea mayor de todos.

Antes hay que ver los promedios de valores por clasificación de juguete

```
MariaDB [Jugueteria]> SELECT TipoJuguete, AVG(Precio) AS PromedioValor
-> FROM (
->     SELECT j.Precio,
->         CASE
->             WHEN d.IDjuguete IS NOT NULL THEN "Didactico"
->             WHEN m.IDjuguete IS NOT NULL THEN "Mecanico"
->             WHEN di.IDjuguete IS NOT NULL THEN "Digital"
->         END AS TipoJuguete
->     FROM Juguete j
->     LEFT JOIN Didactico d ON j.IDjuguete = d.IDjuguete
->     LEFT JOIN Mecanico m ON j.IDjuguete = m.IDjuguete
->     LEFT JOIN Digital di ON j.IDjuguete = di.IDjuguete
-> ) AS Subconsulta
-> GROUP BY TipoJuguete;
```

TipoJuguete	PromedioValor
Didactico	20666.6667
Digital	21333.3333
Mecanico	17666.6667

Como podemos ver el tipo de juguete con mayor promedio de valor es el digital, por lo que ahora haremos la consulta para que solo nos devuelva aquel con valor promedio mas alto (MAX).

```
MariaDB [Jugueteria]> SELECT TipoJuguete, MAX(precioPreliminar) AS PromedioVal
-> FROM (
->     SELECT AVG(j.Precio) AS precioPreliminar,
->         CASE
->             WHEN d.IDjuguete IS NOT NULL THEN "Didactico"
->             WHEN m.IDjuguete IS NOT NULL THEN "Mecanico"
->             WHEN di.IDjuguete IS NOT NULL THEN "Digital"
->         END AS TipoJuguete
->     FROM Juguete j
->     LEFT JOIN Didactico d ON j.IDjuguete = d.IDjuguete
->     LEFT JOIN Mecanico m ON j.IDjuguete = m.IDjuguete
->     LEFT JOIN Digital di ON j.IDjuguete = di.IDjuguete
->     GROUP BY TipoJuguete
-> ) AS Subconsulta;
```

TipoJuguete	PromedioValor
Didactico	21333.3333

Como podemos ver nos devuelve el que tiene promedio mas alto. El código parece similar pero en el segundo estoy hallando el MAX al AVG realizado dentro del from anidado.