Guía 4

1. Insertar tres registros en la tabla clientes:

INSERT INTO clientes (nombre, email, telefono) VALUES

('Juan Perez', 'juan.perez@example.com', '123456789'),

('Maria Lopez', 'maria.lopez@example.com', '987654321'),

('Carlos Gomez', 'carlos.gomez@example.com', '555555555');

1. Insertar cinco registros en la tabla productos:

sql

Copy

INSERT INTO productos (nombre, descripcion, precio, stock) VALUES

('Laptop', 'Laptop de alta gama', 1500.00, 10),

('Mouse', 'Mouse inalámbrico', 25.00, 50),

('Teclado', 'Teclado mecánico', 75.00, 30),

('Monitor', 'Monitor 24 pulgadas', 200.00, 20),

('Impresora', 'Impresora multifuncional', 120.00, 15);

1. Insertar dos registros en la tabla pedidos:

sql

Copy

INSERT INTO pedidos (cliente\_id, total) VALUES

(1, 1525.00),

(2, 295.00);

1. Insertar detalles para los pedidos anteriores en la tabla detalle\_pedidos:

sql

Copy

INSERT INTO detalle\_pedidos (pedido\_id, producto\_id, cantidad, precio\_unitario) VALUES

(1, 1, 1, 1500.00),

(1, 2, 1, 25.00),

(2, 3, 1, 75.00),

(2, 4, 1, 200.00),

(2, 2, 1, 20.00);

1. Actualizar el teléfono de Juan Perez:

sql

Copy

UPDATE clientes SET telefono = '111111111' WHERE nombre = 'Juan Perez';

1. Incrementar el precio de todos los productos en un 10%:

sql

Copy

UPDATE productos SET precio = precio \* 1.1;

1. Eliminar el producto Impresora:

sql

Copy

DELETE FROM productos WHERE nombre = 'Impresora';

1. Insertar un nuevo pedido para Carlos Gomez:

sql

Copy

INSERT INTO pedidos (cliente\_id, total) VALUES

((SELECT id FROM clientes WHERE nombre = 'Carlos Gomez'), 275.00);

1. Insertar detalles para el nuevo pedido:

sql

Copy

INSERT INTO detalle\_pedidos (pedido\_id, producto\_id, cantidad, precio\_unitario) VALUES

((SELECT MAX(id) FROM pedidos), (SELECT id FROM productos WHERE nombre = 'Teclado'), 1, 75.00),

((SELECT MAX(id) FROM pedidos), (SELECT id FROM productos WHERE nombre = 'Monitor'), 1, 200.00);

1. Eliminar el cliente Maria Lopez:

sql

Copy

DELETE FROM clientes WHERE nombre = 'Maria Lopez';

Guía 5

1. Lista el primer apellido de todos los empleados:

SELECT apellido1 FROM empleado;

1. Lista el primer apellido de los empleados eliminando los apellidos que estén repetidos:

SELECT DISTINCT apellido1 FROM empleado;

1. Lista todas las columnas de la tabla empleado:

SELECT \* FROM empleado;

1. Lista el nombre y los apellidos de todos los empleados:

SELECT nombre, apellido1, apellido2 FROM empleado;

1. Lista el código de los departamentos de los empleados que aparecen en la tabla empleado:

SELECT departamento\_id FROM empleado;

1. Lista el código de los departamentos de los empleados que aparecen en la tabla empleado, eliminando los códigos que aparecen repetidos:

SELECT DISTINCT departamento\_id FROM empleado;

1. Lista el nombre y apellidos de los empleados en una única columna:

SELECT CONCAT(nombre, ' ', apellido1, ' ', IFNULL(apellido2, '')) AS nombre\_completo FROM empleado;

1. Lista el nombre de cada departamento y el valor del presupuesto actual del que dispone:

SELECT nombre, (presupuesto - gastos) AS presupuesto\_actual FROM departamento;

1. Lista el nombre de los departamentos y el valor del presupuesto actual ordenado de forma ascendente:

SELECT nombre, (presupuesto - gastos) AS presupuesto\_actual

FROM departamento

ORDER BY presupuesto\_actual ASC;

1. Lista el nombre de todos los departamentos ordenados de forma ascendente:

SELECT nombre FROM departamento ORDER BY nombre ASC;

1. Lista el nombre de todos los departamentos ordenados de forma descendente:

SELECT nombre FROM departamento ORDER BY nombre DESC;

1. Lista los apellidos y el nombre de todos los empleados, ordenados de forma alfabética tendiendo en cuenta en primer lugar sus apellidos y luego su nombre:

SELECT apellido1, apellido2, nombre

FROM empleado

ORDER BY apellido1, apellido2, nombre;

1. Devuelve una lista con el nombre y el presupuesto, de los 3 departamentos que tienen mayor presupuesto:

SELECT nombre, presupuesto

FROM departamento

ORDER BY presupuesto DESC

LIMIT 3;

1. Devuelve una lista con el nombre y el presupuesto, de los 3 departamentos que tienen menor presupuesto:

SELECT nombre, presupuesto

FROM departamento

ORDER BY presupuesto ASC

LIMIT 3;

1. Devuelve una lista con el nombre de los departamentos y el presupuesto, de aquellos que tienen un presupuesto mayor o igual a 150000 euros:

SELECT nombre, presupuesto

FROM departamento

WHERE presupuesto >= 150000;

1. Devuelve una lista con el nombre de los departamentos y el gasto, de aquellos que tienen menos de 5000 euros de gastos:

SELECT nombre, gastos

FROM departamento

WHERE gastos < 5000;

1. Devuelve una lista con el nombre de los departamentos y el presupuesto, de aquellos que tienen un presupuesto entre 100000 y 200000 euros. Sin utilizar el operador BETWEEN:

SELECT nombre, presupuesto

FROM departamento

WHERE presupuesto >= 100000 AND presupuesto <= 200000;

1. Devuelve una lista con el nombre de los departamentos que tienen un presupuesto entre 100000 y 200000 euros. Utilizando el operador BETWEEN:

SELECT nombre

FROM departamento

WHERE presupuesto BETWEEN 100000 AND 200000;

1. Devuelve una lista con el nombre de los departamentos, gastos y presupuesto, de aquellos departamentos donde los gastos sean mayores que el presupuesto del que disponen:

SELECT nombre, gastos, presupuesto

FROM departamento

WHERE gastos > presupuesto;

1. Devuelve una lista con el nombre de los departamentos, gastos y presupuesto, de aquellos departamentos donde los gastos sean menores que el presupuesto del que disponen:

SELECT nombre, gastos, presupuesto

FROM departamento

WHERE gastos < presupuesto;

1. Devuelve una lista con el nombre de los departamentos, gastos y presupuesto, de aquellos departamentos donde los gastos sean iguales al presupuesto del que disponen:

SELECT nombre, gastos, presupuesto

FROM departamento

WHERE gastos = presupuesto;

1. Lista todos los datos de los empleados cuyo segundo apellido sea NULL:

SELECT \* FROM empleado WHERE apellido2 IS NULL;

1. Lista todos los datos de los empleados cuyo segundo apellido no sea NULL:

SELECT \* FROM empleado WHERE apellido2 IS NOT NULL;

1. Lista todos los datos de los empleados cuyo segundo apellido sea López:

SELECT \* FROM empleado WHERE apellido2 = 'López';

1. Lista todos los datos de los empleados cuyo segundo apellido sea Díaz o Moreno. Sin utilizar el operador IN:

SELECT \* FROM empleado WHERE apellido2 = 'Díaz' OR apellido2 = 'Moreno';

1. Lista todos los datos de los empleados cuyo segundo apellido sea Díaz o Moreno. Utilizando el operador IN:

SELECT \* FROM empleado WHERE apellido2 IN ('Díaz', 'Moreno');

1. Lista los nombres, apellidos y nif de los empleados que trabajan en el departamento 3:

SELECT nombre, apellido1, apellido2, nif

FROM empleado

WHERE departamento\_id = 3;

1. Lista los nombres, apellidos y nif de los empleados que trabajan en los departamentos 2, 4 o 5:

SELECT nombre, apellido1, apellido2, nif

FROM empleado

WHERE departamento\_id IN (2, 4, 5);

1. Lista los nombres, apellidos de los empleados cuyo nif contengan "GE":

SELECT nombre, apellido1, apellido2

FROM empleado

WHERE nif LIKE '%GE%';

Ahora, procederé a resolver las tareas del archivo "Guia Practica Adicional 4.pdf". Primero, necesitamos crear las tablas necesarias: