PRUEBA TEORICA

# PRUEBA GIT

1. ¿Cuál es el comando de git para descargar un repositorio completo de git?
2. Push
3. Fork
4. Clone
5. Commit
6. ¿Cuál es el comando para subir los cambios realizados en su repositorio local al repositorio central?
7. Push
8. Fork
9. Clone
10. Commit
11. ¿Como agrego un mensaje a un commit?
12. message “Mensage”
13. add “Mensage”
14. commit “Mensage”
15. commit -m “Mensage”
16. ¿Como cambio de rama en git?
17. Pull
18. Merge
19. checkout
20. rebase
21. ¿Con cuál comando mezclo los cambios de otra rama (Seleccione 2)?
22. checkout
23. merge
24. rebase
25. pull
26. ¿Como veo las modificaciones que he realizado a un archivo antes de hacer un commit?
27. Log
28. status
29. diff
30. tag
31. ¿ Cuál es la función principal del comando git stash?
32. Combinar dos ramas diferentes en un solo commit.
33. Guardar temporalmente los cambios no confirmados en el área de trabajo para poder aplicar otros cambios.
34. Eliminar todos los archivos no rastreados en el repositorio.
35. Crear una nueva rama a partir del último commit confirmado.

# PRUEBA SQL

Resalte la opción Correcta

1. ¿Qué significan las siglas SQL?
   1. Structured Query Language
   2. Structured Question Line
   3. Strong Question Language
2. ¿Qué comando extrae datos de la base de datos?
   1. GET
   2. OPEN
   3. SELECT
3. ¿Qué comando actualiza datos en la base de datos?
   1. SAVE AS
   2. SAVE
   3. UPDATE
4. ¿Qué comando borra datos de la base de datos?
   1. DELETE
   2. BLANK
   3. REMOVE
5. ¿Qué comando inserta datos en la base de datos?
   1. ADD
   2. NEW
   3. INSERT
6. ¿Cuál sería la sintaxis correcta para seleccionar la columna llamada "Apellidos" de una tabla llamada "Trabajadores"?
   1. EXTRACT Apellidos FROM Trabajadores
   2. SELECT Trabajadores, Apellidos
   3. SELECT Apellidos FROM Trabajadores
7. ¿Cuál es la sintaxis correcta para seleccionar todos los campos de una tabla llamada "Trabajadores"?
   1. SELECT \* FROM Trabajadores
   2. SELECT [all] FROM Trabajadores
   3. SELECT Trabajadores
8. ¿Cuál es la sintaxis correcta para seleccionar todos los campos de una tabla llamada Trabajadores, en los que la columna "Nombre" es "Raúl"?
   1. SELECT \* FROM Trabajadores WHERE Nombre: 'Raúl'
   2. SELECT \* FROM Trabajadores WHERE Nombre='Raúl'
   3. SELECT \* FROM Trabajadores LIKE 'Raúl'
9. Si se quisiera seleccionar todos los campos de una tabla llamada "Trabajadores", en los que el campo "Nombre" comienza con una "a", la consulta sería...
   1. SELECT \* FROM Trabajadores LIKE Nombre='%a'
   2. SELECT \* FROM Trabajadores WHERE Nombre LIKE 'a%'
   3. SELECT \* FROM Trabajadores WHERE Nombre
10. ¿Qué palabra clave se utiliza para ordenar el resultado?
    1. ORDER
    2. SORT
    3. ORDER BY
11. ¿Cuál es la sintaxis correcta si se desea insertar una nueva fila en la tabla "Trabajadores" (cuyos campos son "Nombre" y "Apellidos")?
    1. INSERT ('Carlos', 'Pérez') INTO Trabajadores
    2. INSERT INTO Trabajadores VALUES ('Carlos', 'Pérez')
    3. INSERT VALUES ('Carlos', 'Pérez') INTO Trabajadores
12. ¿Cómo cambiaría "Carlos" por "Javier" en la columna "Nombre" de la tabla "Trabajadores"?
    1. UPDATE Trabajadores SET Nombre = 'Carlos' INTO Nombre = 'Javier'
    2. UPDATE Trabajadores SET Nombre = 'Javier' WHERE Nombre = 'Carlos'
    3. SAVE Trabajadores SET Nombre = 'Carlos' INTO Nombre = 'Javier'
13. ¿Cuál es la sintaxis correcta para borrar los registros cuyo campo "Nombre" sea "Andrés"?
    1. DELETE FROM Trabajadores WHERE Nombre = 'Andrés'
    2. DELETE ROW Nombre='Andrés' FROM Trabajadores
    3. DELETE Nombre='Andrés' FROM Trabajadores
14. Si se quisiera contar el número de registros de la tabla "Trabajadores"...
    1. SELECT NUMBER FROM Trabajadores
    2. COUNT Trabajadores
    3. SELECT COUNT (\*) FROM Trabajadores
15. De acuerdo con las siguientes tablas

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ARTICULOS | | |  | VENDEDOR | | |
| CODIGO | DESCRIP | | CEDULA | | NOMBRE |
| T346 | SOPORTE | | 10678567 | | JUAN CARDONA |
| F435 | TERMO | | 42567890 | | CARLOS VASQUEZ |
| R234 | TINA | | 31234543 | | PEDRO GIRALDO |
|  | | | | | | |
| VENTAS | | | | | |  |
| MES | CODIGO | CEDULA | | | VENTA |
| 2 | T346 | 10678567 | | | 23'000 |
| 2 | F435 | 42567890 | | | 34'000 |
| 2 | R234 | 31234543 | | | 12'000 |
| 3 | T346 | 10678567 | | | 65'000 |
| 3 | F435 | 42567890 | | | 32'000 |
| 3 | R234 | 31234543 | | | 23'000 |
| 4 | T346 | 10678567 | | | 23'000 |
| 4 | F435 | 42567890 | | | 34'000 |
| 4 | R234 | 31234543 | | | 12'000 |

1. Construya una consulta SQL, que devuelva código de artículo, descripción, cedula, nombre vendedor, total de ventas(sumatoria) agrupando por articulo y vendedor

SELECT a.CODIGO, a.DESCRIP, v.CEDULA, v.NOMBRE, SUM(vt.VENTA) as TOTAL\_VENTAS

FROM ARTICULOS a

JOIN VENTAS vt ON a.CODIGO = vt.CODIGO

JOIN VENDEDOR v ON vt.CEDULA = v.CEDULA

GROUP BY a.CODIGO, a.DESCRIP, v.CEDULA, v.NOMBRE

1. Construya una consulta SQL que muestre el total (sumatoria) de ventas agrupando por vendedor (in- cluir nombre de vendedor) ordenando ventas de mayor a menor

SELECT v.CEDULA, v.NOMBRE, SUM(vt.VENTA) as TOTAL\_VENTAS

FROM VENDEDOR v

JOIN VENTAS vt ON v.CEDULA = vt.CEDULA

GROUP BY v.CEDULA, v.NOMBRE

ORDER BY TOTAL\_VENTAS DESC

1. Construya una consulta SQL que retorne la suma de ventas agrupadas por artículo, por mes ordenado ventas de menor a mayor

SELECT a.CODIGO, a.DESCRIP, vt.MES, SUM(vt.VENTA) as TOTAL\_VENTAS

FROM ARTICULOS a

JOIN VENTAS vt ON a.CODIGO = vt.CODIGO

GROUP BY a.CODIGO, a.DESCRIP, vt.MES

ORDER BY TOTAL\_VENTAS ASC

1. Construya una consulta que devuelva los vendedores que en la suma total ventas de los tres meses su- peraron los $ 100.000.

SELECT v.CEDULA, v.NOMBRE, SUM(vt.VENTA) as TOTAL\_VENTAS

FROM VENDEDOR v

JOIN VENTAS vt ON v.CEDULA = vt.CEDULA

GROUP BY v.CEDULA, v.NOMBRE

HAVING SUM(vt.VENTA) > 100000

# PRUEBA DE LOGICA DE PROGRAMACIÓN

1. ¿Qué imprime el siguiente fragmento de código? int x, y, z;

x = 2;

y = x+3; y = x + y;

z = x + x -2;

printf("%d, %d, %d.",x, y, z);

|  |
| --- |
| a) 2, 6, 3. |
| b) 2 6 3 |
| c) 2, 7, 2. |

d) ninguna de las anteriores

1. ¿Qué imprime el siguiente fragmento de código? int a= 67, b= 77, c=88;

if(a>b) putchar('a'); putchar('e');

if(b>c) putchar('b'); else putchar('c'); putchar('d');

|  |
| --- |
| a) e c d |
| b) c |
| c) a c d |
| d) ninguna de las anteriores |

1. ¿Qué imprime el siguiente fragmento de código? int i=3;

for(i=1;i<5;i--)

{

i=3\*i;

printf("%d\n",i);

}

|  |
| --- |
| a) 1 2 3 |
| b) 3 6 9 |
| c) 3 6 |
| d) ninguna de las anteriores |

1. ¿Qué imprime el siguiente fragmento de código? int i=1, j;

while(i<=6)

{

j=1;

while(j<=3)

{

printf("%d ",i); i++; j++;

}

putchar('/');

}

|  |
| --- |
| a) 3 / 4 / 5 / 6 |
| b) 1 2 3 /4 5 6 |
| c) 1 2 3 /4 5 6 /7 8 9 / |
| d) ninguna de las anteriores |

1. Partiendo de una captura de tres números en las variables A, B, C

# por ejemplo:

A=7; B=5; C=9;

Construya en pseudocódigo un algoritmo que imprima los números antes mencionados ordenados de mayor a menor, teniendo en cuenta que la captura toma cualquier número en cualquier Orden. Utilice las senten- cias, ciclos u operadores que requiera sin importar la sintaxis.

Captura de los números A, B y C (por ejemplo: A=7; B=5; C=9;)

1. Inicializa tres variables para almacenar los números ordenados: mayor, medio y menor.

2. Compara A con B y C para determinar el mayor, el medio y el menor.

- Si A es mayor que B y A es mayor que C:

- mayor = A

- Si B es mayor que C:

- medio = B

- menor = C

- Sino:

- medio = C

- menor = B

- Si B es mayor que A y B es mayor que C:

- mayor = B

- Si A es mayor que C:

- medio = A

- menor = C

- Sino:

- medio = C

- menor = A

- Sino (C es mayor que A y C es mayor que B):

- mayor = C

- Si A es mayor que B:

- medio = A

- menor = B

- Sino:

- medio = B

- menor = A

3. Imprime los números ordenados de mayor a menor: mayor, medio, menor.

**PRUEBA PRACTICA**

# Se debe crear un CRUD para una tienda online:

✔ Debe tener las siguientes entidades base: Product, Customer, Sale.

✔ Puede agregar las entidades adicionales que considere conveniente.

✔ Versionado en git.

✔ Puede utilizar cualquier framework PHP orientado a objetos, o si desea con PHP puro. (Preferible Symfony 4 o superior)

✔ Puede utilizar cualquier sistema de base de datos SQL relacional.

# Reglas de negocio.

✔ La tienda puede tener muchos productos

✔ La tienda puede tener muchos Clientes

✔ Los clientes pueden comprar 1 ó muchos productos

✔ Si compra más de 5 productos de la misma referencia se aplica un 10% de descuento al precio de venta del producto.

✔ Debe simular un stock, es decir no puede vender más productos de los que hay en existencia.

✔ Agregar validaciones con javascript a los formularios.

# Informes

✔ Clasificación de los primeros 5 clientes con más compras en el mes.

✔ Clasificación primeros 5 productos más vendidos en el mes

✔ Mostrar el valor total por producto y por cliente comprados en un rango de fechas.

✔ Ej:

|  |  |  |
| --- | --- | --- |
| Cliente | Producto | Total |
| Pepito Perez | Producto 1 | $1200000 |
| Pepito Perez | Producto 2 | $5200000 |
| Paco Lopez | Producto 3 | $530000 |
| Paco Lopez | Producto 1 | $230000 |

# API REST

✔ Debe crear un endpoint que me devuelva la lista de productos en stock.

✔ Crear otra vista que consuma este endpoint y muestra la lista de de productos

✔ Crear un formulario para filtrar por código de producto, y me devuelva la información del producto y el stock sin recargar la página utilizando javascript. *Ej endpoint:*

*api/product/77023445323*

# GIT

✔ Debe crear un repositorio en GITLAB o GITHUB y subir el código versionado.