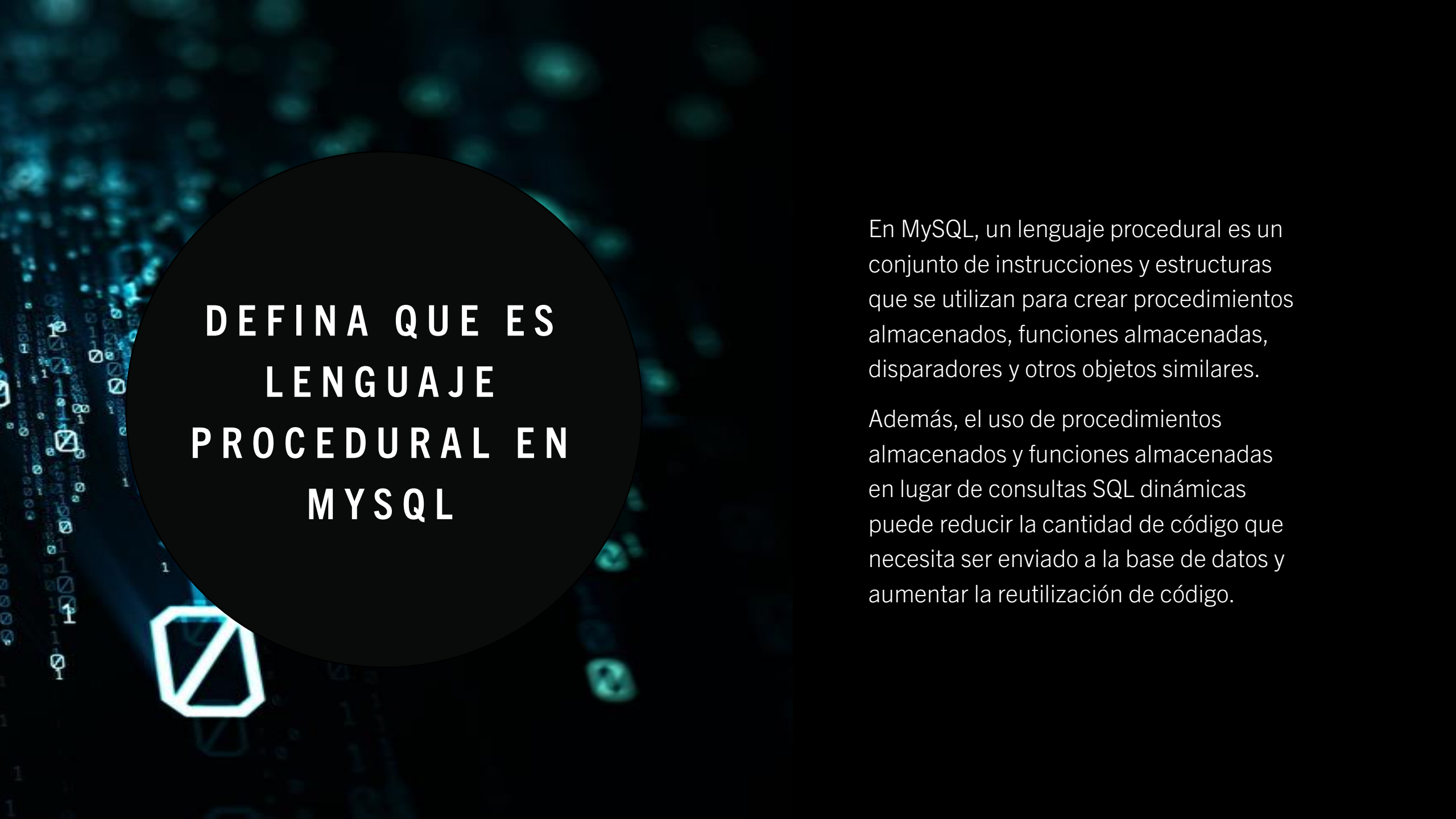




BASE DE DATOS II EVALUACION PROCESUAL- H3

ESTUDIANTE

- HEBER MOLLERICONA MIRANDA



DEFINA QUE ES LENGUAJE PROCEDURAL EN MYSQL

En MySQL, un lenguaje procedural es un conjunto de instrucciones y estructuras que se utilizan para crear procedimientos almacenados, funciones almacenadas, disparadores y otros objetos similares.

Además, el uso de procedimientos almacenados y funciones almacenadas en lugar de consultas SQL dinámicas puede reducir la cantidad de código que necesita ser enviado a la base de datos y aumentar la reutilización de código.



DEFINA QUE ES UNA FUNCIÓN EN MYSQL

Una función es un conjunto de instrucciones que realizan una tarea específica y que pueden ser reutilizadas en diferentes partes de una consulta o procedimiento almacenado. Una función se puede utilizar para realizar cálculos, manipular cadenas de texto o fechas, realizar operaciones matemáticas y más.

¿QUÉ COSAS CARACTERÍSTICAS DEBE DE TENER UNA FUNCIÓN? EXPLIQUE SOBRE EL NOMBRE, EL RETURN, PARAMETROS, ETC

- **Nombre:** Una función debe tener un nombre único y descriptivo que indique claramente su propósito y funcionalidad.
- **Parámetros:** Una función puede aceptar uno o más parámetros de entrada que se utilizan para realizar cálculos y operaciones dentro de la función.
- **Return:** Una función debe tener una declaración RETURN que indique qué valor se debe devolver al llamador de la función.
- **Cuerpo de la función:** Es el conjunto de instrucciones debe ser válido en términos de sintaxis y semántica.
- **Visibilidad:** Las funciones pueden ser públicas o privadas. Las funciones públicas son accesibles desde cualquier parte del sistema, mientras que las funciones privadas solo son accesibles dentro del contexto en el que se definen.

¿CÓMO CREAR, MODIFICAR Y CÓMO ELIMINAR UNA FUNCIÓN?

```
create function edad_minima()  
returns int  
begin  
    declare obt_edad int default 0;  
  
    set obt_edad = ( select min(est.edad)  
                    from estudiantes as est);  
  
    return obt_edad;  
  
end;
```

```
create or replace function edad_minima()  
returns int  
begin  
    declare obt_edad int default 0;  
  
    set obt_edad = ( select min(est.edad)  
                    from estudiantes as est);  
  
    return obt_edad;  
  
end;
```

```
drop function edad_minima;
```

PARA QUÉ SIRVE LA FUNCIÓN CONCAT Y COMO FUNCIONA EN MYSQL

La función CONCAT en MySQL se utiliza para concatenar dos o más cadenas de texto y devolver una cadena resultante que contiene todas las cadenas concatenadas.

```
set resp = concat(resp, numFibo, ' - ');
```

```
`fibonaci(8)`
```

```
0 - 1 - 1 - 2 - 3 - 5 - 8 - 13 -
```

PARA QUÉ SIRVE LA FUNCIÓN SUBSTRING Y COMO FUNCIONA EN MYSQL

- La función SUBSTRING en MySQL se utiliza para extraer una subcadena de una cadena de texto más grande, utilizando un índice de inicio y una longitud. Esto permite obtener una parte específica de la cadena original, en lugar de la cadena completa.

```
select substr('BASE DE DATOS II', 9);
```

```
mysql> `substr('BASE DE DATOS II', 9)`  
DATOS II
```

PARA QUÉ SIRVE LA FUNCIÓN STRCMP Y COMO FUNCIONA EN MYSQL

- La función STRCMP en MySQL se utiliza para comparar dos cadenas de texto y devuelve un valor entero que indica si las cadenas son iguales o diferentes

```
select strcmp('DBA', 'DBA');
```

```
`strcmp('DBA', 'DBA')`
```

```
0
```


PARA QUÉ SIRVE LA
FUNCIÓN
CHAR_LENGTH Y
LOCATE Y COMO
FUNCIONA EN
MYSQL

La función CHAR_LENGTH en MySQL se utiliza para determinar la longitud de una cadena de texto, es decir, el número de caracteres que contiene.

```
select char_length('DBA');
```

```
`char_length('DBA')` ÷  
3
```

La función LOCATE en MySQL se utiliza para encontrar la posición de una subcadena dentro de una cadena de texto.

```
select locate('2', 'DBA II 2023');
```

```
`locate('2', 'DBA II 2023')` ÷  
8
```

¿CUAL ES LA DIFERENCIA ENTRE LAS FUNCIONES DE AGREGACIÓN Y FUNCIONES CREADOS POR EL DBA?

Las funciones de agregación en MySQL son funciones que se utilizan en las consultas SELECT para realizar cálculos sobre un conjunto de valores y devolver un único valor resultante.

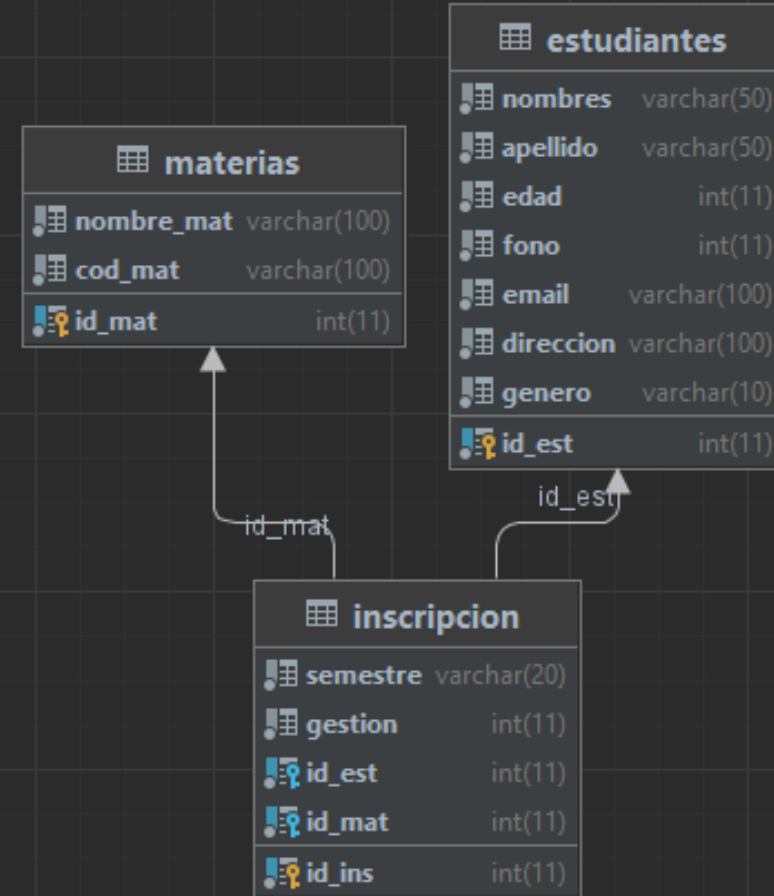
Ejemplos de funciones de agregación en MySQL son SUM, COUNT, AVG, MAX y MIN.

Las funciones creadas por el DBA en MySQL son funciones que se crean utilizando la sentencia CREATE FUNCTION y que permiten a los usuarios definir sus propias funciones personalizadas.

Las funciones creadas por el DBA pueden aceptar uno o varios parámetros y devolver un valor.



CREAR LA SIGUIENTE BASE DE DATOS Y SUS REGISTROS.



```
create or replace function fibonaci(limite int)
  returns text
begin
  declare cont int default 0;
  declare numero int default 1;
  declare numFibo int default 0;
  declare resp text default '';

  while cont < limite do
    set resp = concat(resp, numFibo, ' - ');

    set numero = numFibo + numero;
    set numFibo = numero - numFibo;

    set cont = cont+1;
  end while;
  return resp;
end;

select fibonaci( limite: 8);
```

CREAR UNA FUNCIÓN QUE GENERE LA SERIE FIBONACCI

```
mysql> `fibonaci(8)`
```

```
0 - 1 - 1 - 2 - 3 - 5 - 8 - 13 -
```

CREAR UNA VARIABLE GLOBAL A NIVEL BASE DE DATOS, GENERAR SERIE FIBONACCI

```
▣ `fibonacci_var_global()`
```

```
0 - 1 - 1 - 2 - 3 - 5 - 8 - 13 - 21 - 34 - 55 - 89 - 144 -
```

```
set @limit = 13;

create function fibonacci_var_global()
returns text
begin
    declare numero integer default 1;
    declare fibo integer default 0;
    declare cont int default 0;
    declare resp text default '';

    while cont < @limit do
        set resp = concat(resp, fibo, ' - ');

        set numero = numero + fibo;
        set fibo = numero - fibo;

        set cont = cont+1;
    end while;

    return resp;
end;

select fibonacci_var_global();
```


CREAR UNA FUNCIÓN NO
RECIBE PARÁMETROS
(UTILIZAR WHILE,
REPEAT O LOOP).
OBTENER EDAD MÍNIMA Y
GENERAR NÚMEROS
PARES O IMPARES

```
set @edadMinima_limite = edad_minima();

create or replace function numero_Par_Impar()
returns text
begin
    declare cont integer default 0;
    declare resp text default '';

    if @edadMinima_limite%2 = 0 then

        while cont < @edadMinima_limite do
            set resp = concat(resp, cont, ', ');

            set cont = cont + 2;
        end while;
    else
        while @edadMinima_limite >= 0 do

            set resp = concat(resp, @edadMinima_limite, ', ');
            set @edadMinima_limite = @edadMinima_limite - 2;

        end while;
    end if;
    return resp;
end;
```

■ `numero_Par_Impar()`

0, 2, 4, 6, 8, 10, 12, 14, 16, 18,

```
create or replace function edad_minima()
returns int
begin
    declare obt_edad int default 0;

    set obt_edad = ( select min(est.edad)
from estudiantes as est);

    return obt_edad;

end;
```

CREAR UNA FUNCIÓN QUE DETERMINA CUANTAS VECES SE REPITE LAS VOCALES.

```
SQL> `contar_Vocales('taller de base de datos')`  
a: 3, e: 4, i: 0, o: 1, u: 0
```

```
create or replace function contar_Vocales(cadena text)  
returns text  
begin  
    declare resp text default '';  
    declare a, e, i, o, u integer default 0;  
    declare posicion integer default 1;  
    declare substring text default '';  
  
    while posicion <= char_length(cadena) do  
  
        set substring = substring(cadena, posicion, 1);  
  
        if substring = 'a' then  
            set a = a+1;  
        end if;  
        if substring = 'e' then  
            set e = e+1;  
        end if;  
        if substring = 'i' then  
            set i = i+1;  
        end if;  
        if substring = 'o' then  
            set o = o+1;  
        end if;  
        if substring = 'u' then  
            set u = u+1;  
        end if;  
  
        set posicion = posicion+1;  
  
    end while;  
    set resp = concat('a: ', a, ', e: ', e, ', i: ', i, ', o: ', o, ', u: ', u);  
  
    return resp;  
end;
```

```
create function identificar_(credit_number int)
  returns text
begin
  declare resp text default '';

  case
    when credit_number > 50000 then set resp = 'PLATINIUM';
    when credit_number >= 10000 and credit_number <= 50000 then set resp = 'GOLD';
    when credit_number < 10000 then set resp = 'SILVER';
  end case;

  return resp;
end;

select identificar_( credit_number: 5200000);
```

CREAR UNA FUNCIÓN
QUE RECIBE UN
PARÁMETRO INTEGER

```
`identificar_(5200000)`
1 PLATINIUM
```

CREAR UNA FUNCIÓN QUE RECIBE 2 PARÁMETROS VARCHAR(20), VARCHAR(20). ELIMINAR LAS VOCALES

```
❏ `eliminar_vocales('TALLER DBA II', 'GESTION 2023')`  
TLLR DB - GSTN 2023
```

```
create or replace function eliminar_vocales(cad_1 varchar(20), cad_2 varchar(20))  
returns text  
begin  
    declare posicion int default 1;  
    declare borrar, substri text default '';  
    declare concate text default '';  
    declare resp text default '';  
    set concate = concat(cad_1, ' - ', cad_2);  
  
    while (posicion <= char_length(concate)) do  
        set substri = substr(concate, posicion, 1);  
  
        if substri = 'a' or substri = 'e' or substri = 'i' or substri = 'o' or substri = 'u' then  
            set borrar = substri;  
        else  
            set resp = concat(resp, substri);  
        end if;  
  
        SET posicion = posicion+1;  
    end while;  
  
    return resp;  
end;
```

```
select eliminar_vocales( cad_1: 'TALLER DBA II', cad_2: 'GESTION 2023');
```

CREAR UNA FUNCIÓN QUE RECIBA UN PARÁMETRO TEX EN DONDE ESTE PARÁMETRO DEBERÁ DE RECIBIR UNA CADENA

```
create or replace function reducir_cadena(cadena text)
returns text
begin
    declare resp text default '';
    declare posicion int default 0;
    declare cad_substr text default '';
    declare tam int default 0;

    set posicion = char_length(cadena);
    repeat
        set cad_substr = subStr(cadena, -posicion, posicion);

        set resp = concat(resp, cad_substr, ', ');

        set posicion = posicion-1;
    until posicion = 0 end repeat;

    return resp;
end;
```

```
select reducir_cadena( cadena: 'BASE DE DATOS II');
```

```
└─ `reducir_cadena('BASE DE DATOS II')`
```

```
BASE DE DATOS II, ASE DE DATOS II, SE DE DATOS II, E DE DATOS II, DE DATOS II, DE DATOS II, E DATOS II, DATOS II, DATOS II, ATOS II, TOS II, OS II, S II, II, II, I,
```