

Primer entrega Proyecto Final SQL

CODERHOUSE

SQL Comisión 34950

Manuel Marchena

Índice

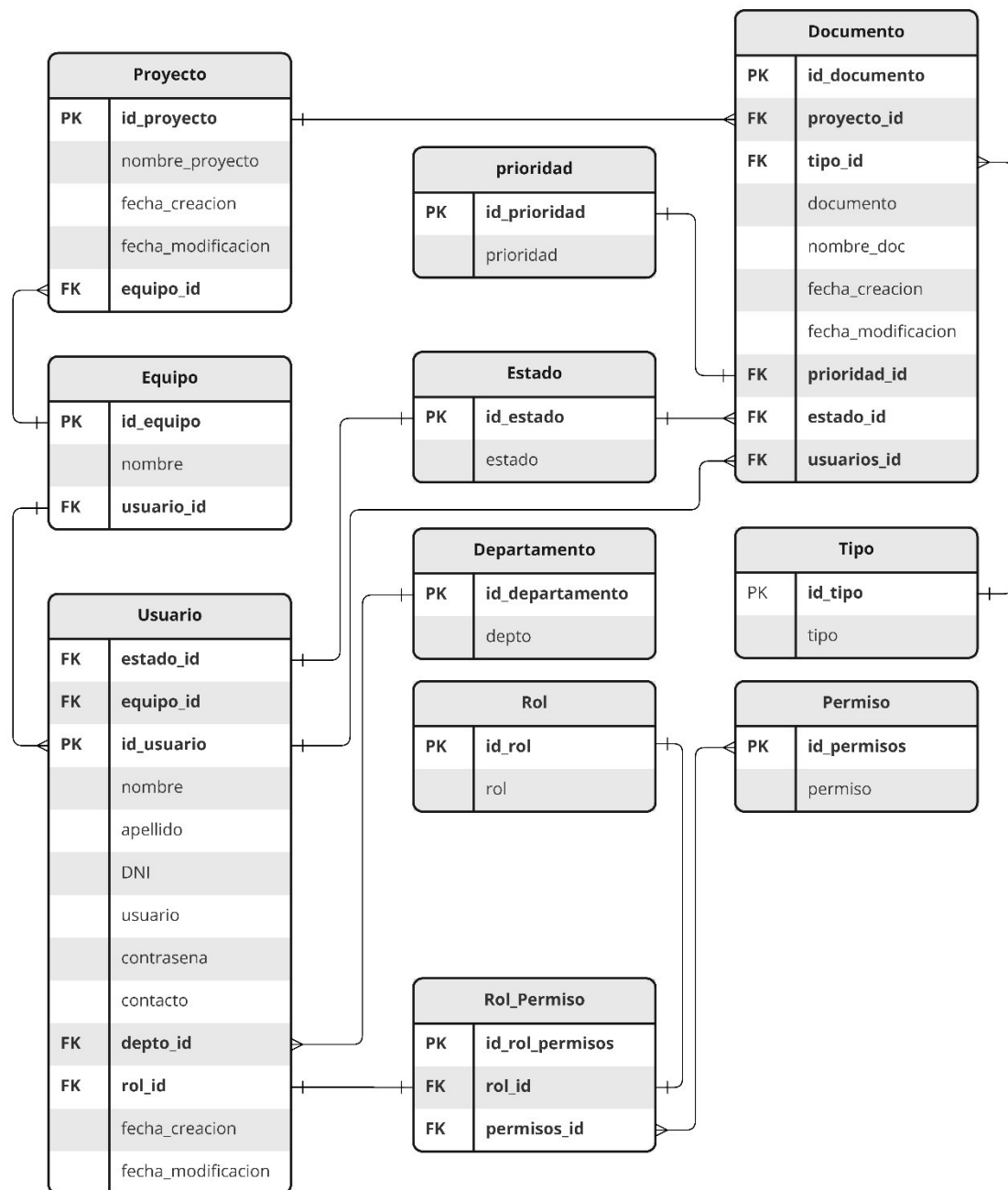
Contenido	Pág
I. Diagrama Entidad Relación	3
1. Descripción teórica del proyecto	3
2. Definición de la DB	4
3. Tablas contenidas en la Base de Datos	4
4. CREATE TABLE	5
4.1. departamento	5
4.2. estado	5
4.3. permisos	5
4.4. prioridad	6
4.5. rol	6
4.6. equipos	7
4.7. usuarios	7
4.8. tipo	8
4.9. proyecto	8
4.10. documento	9
5.- INSERT INTO	9
5.1.- Por script	9

5.2.- Por importación	10
Paso 1. – Creación de los datos en archivo Excel	11
Paso 2. – Conversión de archivo de .xlsx a .csv	12
Paso 3. – Importación de datos por Wizard de Workbench	12
Paso 4. – Selección de Archivo	13
Paso 5. – Selección Destino	13
Paso 6. – Verificación de configuración	14
Paso 7. – Ejecución de Importación	14
Paso 8. – Verifica INSERT en la tabla	15
6.- Vistas	15
6.1.- Vista 1, Usuarios por equipo	18
6.2.- Vista 2, Permisos por cargo	19
6.3.- Vista 3, Equipos por proyecto	20
6.4.- Documentos prioridad estado	21
6.5.- Vista 5, Documento prioridad baja	22
7.- Funciones	22
7.1.- Documento demorado	22
7.2.- Usuarios Inactivos	23
8.- Procedimientos	23
8.1.- Cambio de estado	23
8.2.- Order by	24
9.- Repositorio GitHub	24

I. Diagrama Entidad Relación

[\[Volver\]](#)

Entidades de proyecto Control de Documentacion



1. DESCRIPCIÓN TEÓRICA DEL POYECTO

[\[Volver\]](#)

La base de datos estará estructurada de manera que podrá almacenar la documentación de los proyectos (definición funcional de Backend, Frontend, Manuales de usuario, entre otros...), los usuarios que forman parte de los equipos tendrán acceso a los proyectos y a la documentación, dependiendo del rol de estos podrán tener permisos a realizar gestiones sobre la documentación y garantizar la confiabilidad de los datos.

2. DEFINICIÓN DE LA BASE DE DATOS

La base de datos esta identificada como "*proyecto_final_SQL*" en esta se creará todas las tablas del proyecto, consultas y demás.

3. TABLAS QUE CONTENIDAS EN LA BASE DE DATOS

La base de datos esta compuesta por las siguientes tablas:

- Departamento
- Documentos
- Estado
- Equipos
- Permisos
- Prioridad
- Proyectos
- Rol
- Tipo
- Usuarios

4. CREATE TABLE

[\[Volver\]](#)

4.1. departamento

Departamento se refiere a las diferentes unidades organizacionales como esta dividida la empresa, para efectos de garantizar la homogeneidad de los datos se le asigno como data type un ENUM.

Departamento				
Field	Data type	Null	Key	Extra
Id_dpto	INT	NO	PK	AI
depto	ENUM	NO		

4.2.- estado

Los estados son para definir si el documento se encuentra vigente aun, tiene validez o no.

Estado				
Field	Data type	Null	Key	Extra
Id_estado	INT	NO	PK	AI
estado	ENUM	NO		

4.3.- permisos

Son las diferentes acciones a las que tendrá acceso un usuario, dependiendo de su rol o permisos concedidos podrá eliminar, agregar, modificar, crear usuarios, roles, permisos o documentos.

Permisos				
Field	Data type	Null	Key	Extra
Id_permiso	INT	NO	PK	AI
permiso	VERCHAR(60)	NO		

4.4.- prioridad

[\[Volver\]](#)

La prioridad define que la urgencia con la que se debe finalizar el documento.

Prioridad				
Field	Data type	Null	Key	Extra
Id_prioridad	INT	NO	PK	AI
prioridad	ENUM	NO		

4.5.- rol

Dependiendo del rol que tengan dentro del equipo podrán realizar diferentes acciones o verificar la información.

Rol				
Field	Data type	Null	Key	Extra
Id_rol	INT	NO	PK	AI
nombre_rol	ENUM	NO		

4.7.- rol_permisos

Tabla intermedia para asignar permisos a los roles, de manera de tener un rol con diferentes permisos.

Rol_permisos				
Field	Data type	Null	Key	Extra
Id_rol_permisos	INT	NO	PK	AI
Id_rol	INT	NO	FK	
permisos_id	INT	NO	FK	

4.7.- equipos

[\[Volver\]](#)

Son los grupos de personas responsables del desarrollo y soporte compuesto por desarrolladores BackEnd, FrontEnd, Tech Leader y Project Manager.

Equipo				
Field	Data type	Null	Key	Extra
Id_equipo	INT	NO	PK	AI
nombre	VARCHAR(100)	NO		
usuario_id	INT	NO	FK	

4.7.- Usuarios

Son los usuarios que participan en los proyectos y tienen acceso a los documentos.

Usuarios				
Field	Data type	Null	Key	Extra
Id_usuario	INT	NO	PK	AI
nombre	VARCHAR(100)	NO		
apellido	VARCHAR(100)	NO		
DNI	BIGINT	NO		UNIQUE
contrasena	VARCHAR(20)	NO		
usuario	VARCHAR(20)	NO		UNIQUE
contacto	VARCHAR(20)	NO		UNIQUE
fecha_creacion	DATETIME			
fecha_modificacion	DATETIME			
estado_id	INT	NO	FK	
rol_id	INT	NO	FK	
equipo_id	INT	NO	FK	
depto_id	INT	NO	FK	

4.8.- tipo

[\[Volver\]](#)

Define que tipo de documento se está ingresando en la DB, definición funcional, manual de usuario, entre otros...

Tipo				
Field	Data type	Null	Key	Extra
Id_tipo	INT	NO	PK	AI
tipo	VARCHAR(100)	NO		

4.9.- proyecto

Cada proyecto es asignado a un equipo que será responsable de su desarrollo y soporte, estos son los que tienen acceso al proyecto y su documentación.

Proyecto				
Field	Data type	Null	Key	Extra
Id_proyecto	INT	NO	PK	AI
Nombre_proyecto	VARCHAR(100)	NO		
fecha_creacion	DATETIME	NO		
fecha_modificacion	DATETIME	NO		
equipo_id	INT	NO	FK	

4.10.- Documentos

[\[Volver\]](#)

Los documentos para almacenar serán las definiciones funcionales, manuales de usuarios y el documento que se realizó, la estructura de la tabla se espera contenga los datos necesarios para identificar el documento.

Documento				
Field	Data type	Null	Key	Extra
Id_documento	INT	NO	PK	AI
nombre_doc	VARCHAR(100)	NO		
documento	BLOB	SI		
fecha_creacion	DATETIME	NO		
fecha_modificacion	DATETIME	NO		
prioridad_id	INT			
estado_id	INT	NO	FK	
usuario_id	INT	NO	FK	
proyecto_id	INT	NO	FK	
tipo_id	INT	NO	FK	

5.- INSERT INTO

5.1.- Por script

Para poblar las tablas a través de script se escogieron las tablas:

- Prioridad.
- Tipo.
- Departamento.
- Estado.
- Permisos.

En el Script del proyecto a partir de la línea 182 se pueden observar los INSERT INTO

5.2.- Por importación

[\[Volver\]](#)

A través de Mockaroo se generaron los datos que fueron insertados por importación, las tablas seleccionadas para importación fueron:

- Rol.
- Equipos.
- Usuarios.
- Proyecto
- Documento

Los pasos seguidos para realizar la importación fueron los siguientes:

Paso 1.– Creación de los datos en archivo Excel

[\[Volver\]](#)

Autoguardado Data Import [Roles].csv

Archivo Inicio Insertar Disposición de página Fórmulas

Pegar

Calibri 11 A A

N K S A

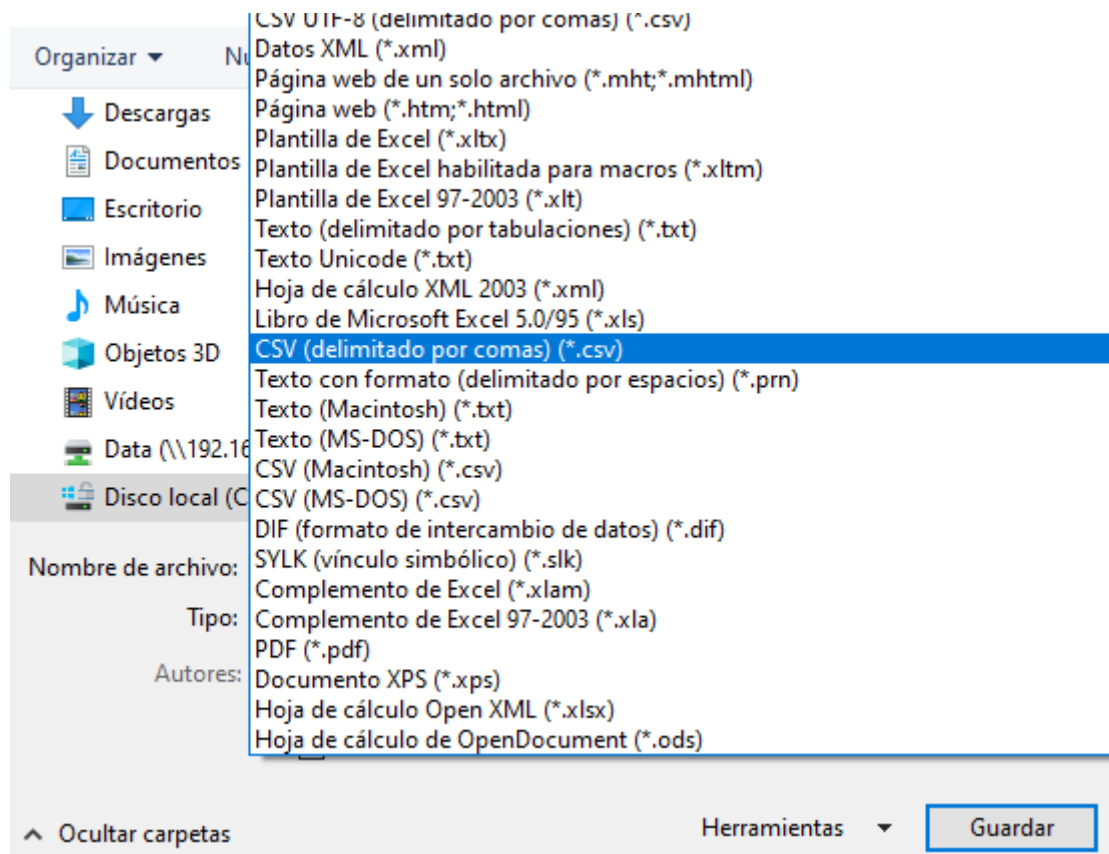
Portapapeles Fuente Alineación

A1 id_rol

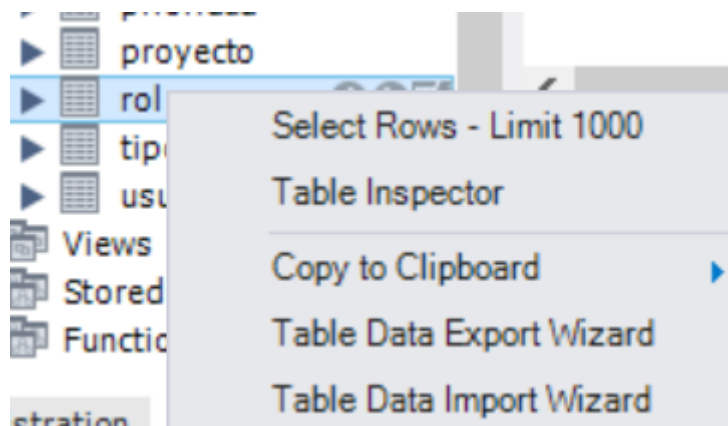
	A	B	C	D	E	F
1	id_rol	nombre_rol				
2		1 Lider Tecnico				
3		2 Desarrollador				
4		3 DBA				
5		4 DevOps				
6		5 Gerente Desarrollo				
7		6 Analista Funcional				
8						
9						
10						

Paso 2.– Conversión de archivo de .xlsx a .csv

[\[Volver\]](#)

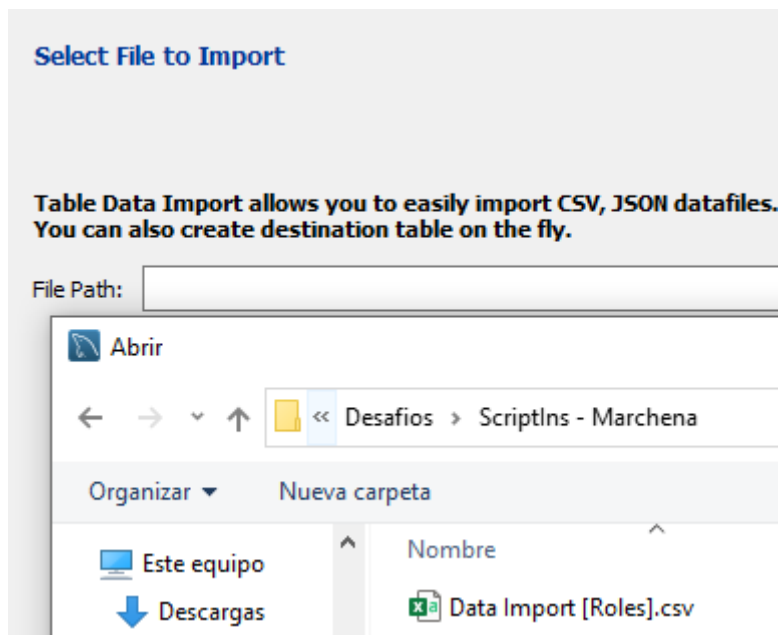


Paso 3.– Importación de datos por Wizard de Workbench



Paso 4.– Selección de Archivo

[\[Volver\]](#)



Paso 5.– Selección Destino

Select Destination

Select destination table and additional options.

☒ Use existing table:


☐ Create new table: .

☐ Truncate table before import

Paso 6.– Verificación de configuración




[\[Volver\]](#)

Configure Import Settings

Detected file format: csv 

Encoding:

Columns:

<input checked="" type="checkbox"/> Source Column	Dest Column
<input checked="" type="checkbox"/> id_rol	<input type="text" value="id_rol"/> 
<input checked="" type="checkbox"/> nombre_rol	<input type="text" value="nombre_ro"/> 
<input checked="" type="checkbox"/> permisos_id	<input type="text" value="permisos_j"/> 

id_rol	nombre_rol	permisos_id
1	Lider Tecnico	2
1	Lider Tecnico	4
1	Lider Tecnico	3
2	Desarrollador	3

Paso 7.– Ejecución de Importación

Import Data

The following tasks will now be performed. Please monitor the execution.

- ☒ Prepare Import
- ☒ Import data file

Finished performing tasks. Click [Next >] to continue.

Paso 8.– Verifica INSERT en la tabla.

[\[Volver\]](#)



	id_rol	nombre_rol
▶	1	Lider Tecnico
	2	Desarrollador
	3	DBA
	4	DevOps
	5	Gerente Desarrollo
	6	Analista Funcional
✱	NULL	NULL

6.- Vistas

Se realizaron 5 vistas, estas trataron de abarcar consultas que podrían ser las mas requeridas. A continuación se transcriben las vistas generadas, estas igualmente están en el repositorio de GitHub

6.1.- [Vista 1, Usuarios por equipo](#)

Objetivo

Muestra los integrantes de cada uno de los equipos de desarrollo, el rol que tiene cada integrante.

Los equipos son:

- Guemes 1
- Guemes 2
- Guemes 3
- Parque patricio

Los roles

- DevOps
- Gerente de Desarrollo
- Líder Técnico
- Desarrollador
- DBA

Tablas involucradas

- Equipo
- Rol
- usuarios

6.2.- [Vista 2, Permisos por cargo](#)

Objetivo

Mostrar los permisos con los que cuenta cada rol dentro de la DB, cada rol tiene una serie de permisos asignados, de forma que sea mas fácil conceder permisos dentro de la DB.

Los permisos creados son:

- Asignar Equipo
- Crear Documento
- Crear proyecto
- Crear Usuario
- Editar Documento
- Eliminar Proyecto
- Eliminar Usuario

Tablas involucradas

- Permisos
- Rol
- Rol_permisos

6.3.- [Vista 3, Equipos por proyecto](#)

Objetivo

Muestra los proyectos de los que cada equipo es responsable de su desarrollo, mantenimiento y soporte

Tablas involucradas

- Proyecto
- Equipo
- Rol
- Usuario

6.4.- [Documentos prioridad estado](#)

Objetivo

Muestra cada uno de los proyectos registrados en la DB, la prioridad de estos y el estado del documento

La prioridad puede ser:

- Alta
- Media
- Baja

El estado:

- Activo
- Inactivo

Tablas involucradas

- Documento
- Prioridad
- Estado

6.5.- Vista 5, Documento prioridad baja

Objetivo

Muestra los documentos referidos al rol DBA que se encuentran activos y tienen prioridad baja

Tablas involucradas

- Documento
- Prioridad
- Estado
- Usuario
- Rol

6.1.- Vista 1, Usuarios por equipo

[\[Volver\]](#)

```
CREATE
  ALGORITHM = UNDEFINED
  DEFINER = `root`@`localhost`
  SQL SECURITY DEFINER
VIEW `proyecto_final_sql`.`usuarios_por_equipo` AS
  (SELECT
    `equ`.`nombre` AS `Equipo`,
    `proyecto_final_sql`.`rol`.`nombre_rol` AS `Puesto`,
    CONCAT(`usu`.`nombre`, ' ', `usu`.`apellido`) AS `Nombre`
  FROM
    ((`proyecto_final_sql`.`usuario` `usu`
    JOIN `proyecto_final_sql`.`equipo` `equ` ON (`equ`.`id_equipo` =
`usu`.`equipo_id`))
    JOIN `proyecto_final_sql`.`rol` ON
    (`proyecto_final_sql`.`rol`.`id_rol` = `usu`.`rol_id`)))
```

6.2.- Vista 2, Permisos por cargo

[\[Volver\]](#)

```
CREATE
  ALGORITHM = UNDEFINED
  DEFINER = `root`@`localhost`
  SQL SECURITY DEFINER
VIEW `proyecto_final_sql`.`permisos_por_cargo` AS
  (SELECT
    `proyecto_final_sql`.`rol`.`nombre_rol` AS `Cargo`,
    `per`.`permiso` AS `permiso`
  FROM
    ((`proyecto_final_sql`.`permisos` `per`
    JOIN `proyecto_final_sql`.`rol_permisos` `rolp` ON
    (`rolp`.`permisos_id` = `per`.`id_permisos`))
    JOIN `proyecto_final_sql`.`rol` ON
    (`proyecto_final_sql`.`rol`.`id_rol` = `rolp`.`rol_id`)))
```

6.3.- Vista 3, Equipos por proyecto

[\[Volver\]](#)

```
CREATE
  ALGORITHM = UNDEFINED
  DEFINER = `root`@`localhost`
  SQL SECURITY DEFINER
VIEW `proyecto_final_sql`.`equipos_por_proyecto` AS
  (SELECT
    `pro`.`nombre_proyecto` AS `nombre_proyecto`,
    `equ`.`nombre` AS `nombre`,
    CONCAT(`usu`.`nombre`, ' ', `usu`.`apellido`) AS `Nombre_apellido`,
    `proyecto_final_sql`.`rol`.`nombre_rol` AS `nombre_rol`
  FROM
    (((`proyecto_final_sql`.`proyecto` `pro`
    JOIN `proyecto_final_sql`.`equipo` `equ` ON (`pro`.`equipo_id` =
    `equ`.`id_equipo`))
    JOIN `proyecto_final_sql`.`usuario` `usu` ON (`equ`.`id_equipo` =
    `usu`.`equipo_id`))
    JOIN `proyecto_final_sql`.`rol` ON
    (`proyecto_final_sql`.`rol`.`id_rol` = `usu`.`rol_id`)))
```

6.4.- Documentos prioridad estado

[\[Volver\]](#)

```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `proyecto_final_sql`.`documentos_prioridad_estado` AS
    SELECT
        `doc`.`nombre_doc` AS `nombre_doc`,
        `pri`.`prioridad` AS `prioridad`,
        `est`.`estado` AS `estado`
    FROM
        ((`proyecto_final_sql`.`documento` `doc`
        JOIN `proyecto_final_sql`.`prioridad` `pri` ON (`pri`.`id_prioridad`
        = `doc`.`prioridad_id`))
        JOIN `proyecto_final_sql`.`estado` `est` ON (`doc`.`estado_id` =
        `est`.`id_estado`))
```

6.5.- Vista 5, Documento prioridad baja

[\[Volver\]](#)

```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `proyecto_final_sql`.`documento_prioridad_baja_dba` AS
    (SELECT
        `doc`.`nombre_doc` AS `nombre_doc`,
        `pri`.`prioridad` AS `prioridad`,
        `est`.`estado` AS `estado`,
        CONCAT(`usu`.`nombre`, ' ', `usu`.`apellido`) AS `Nombre`,
        `proyecto_final_sql`.`rol`.`nombre_rol` AS `nombre_rol`
    FROM
        (((((`proyecto_final_sql`.`documento` `doc`
        JOIN `proyecto_final_sql`.`prioridad` `pri` ON (`pri`.`id_prioridad` =
        `doc`.`prioridad_id`))
        JOIN `proyecto_final_sql`.`estado` `est` ON (`doc`.`estado_id` =
        `est`.`id_estado`))
        JOIN `proyecto_final_sql`.`usuario` `usu` ON (`usu`.`id_usuario` =
        `doc`.`usuario_id`))
        JOIN
            `proyecto_final_sql`.`rol`
        ON
        (`proyecto_final_sql`.`rol`.`id_rol` = `usu`.`rol_id`))
    WHERE
        `proyecto_final_sql`.`rol`.`nombre_rol` = 'DBA'
        AND `est`.`estado` = 'Activo'
        AND `pri`.`prioridad` = 'Baja')
```

7.- Funciones

7.1.- Documento demorado

Objetivo

Esta función tiene por objetivo determinar si un documento se encuentra demorado, recibiendo como argumento el nombre del documento, en caso de encontrarse demorado arroja la fecha de creación del documento, caso contrario *'null'* de manera que, si han pasado 180 días desde su creación, su prioridad es alta y fue creado antes de julio del 2022 arroja la fecha de creación del documento.

Tablas

Documento

Prioridad

7.2.- Usuarios Inactivos

[\[Volver\]](#)

Objetivo

La función tiene por objetivo determinar si un usuario se encuentra activo o inactivo, para ello recibe dos argumentos, el DNI del usuario y el equipo al que pertenece

Los argumentos validos par equipo son

- Guemes 1
- Guemes 2
- Guemes 3
- Parque Patricios

Cualquier valor diferente arroja *'null'*.

8.- Procedimientos

8.1.- Cambio de estado

Objetivo

Este procedimiento tiene como finalidad cambiar la prioridad de un documento a partir de los 3 argumentos que recibe p_prioridad, p_nombre, p_prioridad_nueva,

El parámetro p_nombre recibe el nombre del documento como se encuentra registrado en la tabla documento

El parámetro p_prioridad recibe los valores

- Alta
- Media
- Baja

El parámetro p_prioridad_nueva recibe el cambio de prioridad, donde se alterará en la tabla documento, y puede recibir los parámetros

- 1 para Alta
- 2 para Media
- 3 para Baja

Tablas

8.2.- Order by

Objetivo

Este procedimiento tiene por objeto ordenar una tabla en función de la columna y orden que seleccione el usuario,

Primero recibe el nombre de la tabla luego la columna por la que quiere ordenar la tabla y finalmente el ascendente o descendente, es decir

p_tabla: Nombre de la tabla que desea ordenar

p_columna: Por cual columna de la tabla desea ordenar

p_orden: desea ordenar ascendente, entonces ASC caso contrario DESC

9.- Repositorio GitHub

[\[Volver\]](#)

<https://github.com/manuelmarchena/CoderHouseSQL/tree/manuelmarchena-finalSQL>