**Documentación SQL ICCP.Net**

**INTEGRATED CUSTOMER COMPLAINTS PLATFORM DEVELOPED IN ASP.NET**

# Introducción

En la presente documentación se abordará el tema de la base de datos creada para el proyecto Integrated Customer Complaints Platform Developed in .Net (ICCP.Net por sus siglas en inglés) bajo análisis del equipo de trabajo Robotech Inc. para los requisitos tomados desde el <cliente>.

Un Sistema de Bases de Datos es un conjunto de registros, datos y relaciones que refleja las necesidades de información de una empresa. Estos registros datos y relaciones se agrupan en unidades independientes que satisfacen los requerimientos de información de la empresa, además de funcionar como respaldo para la misma.

Todos los datos pertenecientes a una misma Base de Datos están interrelacionados:

• De manera directa, por relaciones establecidas durante el diseño

• De manera indirecta, por el uso que se les dé mediante aplicaciones básicas, así llamadas si el uso de estos datos es primordial en ellas.

Una de las principales ventajas de realizar un diseño lógico y físico de la base de datos para la solución, consiste en:

• Identificar las distintas entidades que pertenecerán a la BD en conjunto con sus atributos.

• Identificar de manera apropiada las relaciones entre dichas entidades y sus jerarquías.

• Identificar los datos críticos que se emplearán en el futuro sistema.

# Diagramas de Clase

# Modelo Entidad-Relación

# Diccionario de Datos

**Usuarios**: Tabla que registra a los usuarios clientes que dejan su reclamo (o se les ha tomado el reclamo) en el sistema

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Columna** | **Tipo** | **Nulo** | **Predeterminado** | **Comentarios** |
| Rut *(primaria)* | Varchar(12) | No |  | Rut de una persona Chilena |
| Nombre | Varchar(50) | No |  | Nombre del cliente |
| Apellido | Varchar(50) | No |  | Apellido del cliente |
| Correo | Varchar(100) | No |  | Correo electrónico del cliente |
| Teléfono | Integer | Si | Null | Teléfono del cliente |

**EstadoReclamo**: Tabla que representa los distintos tipos de estados que puede tener un reclamo en cierto punto de su vida.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Columna** | **Tipo** | **Nulo** | **Predeterminado** | **Comentarios** |
| Id *(primaria)* | Integer | No |  | Identificación del estado |
| Descripcion | Varchar(50) | No |  | Describe el estado |

**TipoReclamo**: Tabla que representa los distintos tipos de reclamos que pueden existir, divididos por áreas de especialidad

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Columna** | **Tipo** | **Nulo** | **Predeterminado** | **Comentarios** |
| Id *(primaria)* | Integer | No |  | Identificación del área o tipo |
| Nombre | Varchar(50) | No |  | Nombre del área o tipo |

**SLAReclamo**: Tabla que representa el Service Level Agreement (SLA) por cada reclamo.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Columna** | **Tipo** | **Nulo** | **Predeterminado** | **Comentarios** |
| Id *(primaria)* | Integer | No |  | Identificación del SLA para un reclamo |
| Fecha\_sla | DateTime | Si | Null | Fecha y hora límite |

**Reclamo**: Tabla que representa el reclamo per-se generado por un cliente mismo o por un operador de la empresa.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Columna** | **Tipo** | **Nulo** | **Predeterminado** | **Comentarios** |
| Id *(primaria)* | Integer | No |  | Identificación del reclamo |
| Área *(foránea)* | Integer | No |  | Área o tipo del reclamo referenciada a TipoReclamo |
| Rut *(foránea)* | Varchar(12) | No |  | Rut del cliente que realiza el reclamo |
| Fecha | DateTime | No |  | Fecha en la cuál fue tomado el reclamo |
| Estado *(foránea)* | Integer | No |  | Estado actual del reclamo |
| PDF | Varchar(100) | Si | Null | Nombre del archivo guardado en el servidor con el PDF del reclamo |
| Comentarios | Text | No |  | Comentarios del cliente sobre su situación |
| Operador *(foránea)* | Varchar(12) | Si | Null | Operador que tomó el reclamo (en caso de ser tomado) |

**Logs**: Tabla que representa el actuar de los distintos actores dentro de la plataforma para su futura auditoría o posible investigación de movimientos

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Columna** | **Tipo** | **Nulo** | **Predeterminado** | **Comentarios** |
| Id\_log *(primaria)* | Integer | No |  | Identificación del registro. Posee auto-incremento de 1 en 1 |
| Rut *(foránea)* | Varchar(12) | No |  | Actor involucrado en la acción |
| Tiempo\_log | DateTime | No |  | Fecha y hora de la acción realizada |
| Mensaje | Text | No |  | Acción realizada dentro del sistema |
| IP | Varchar(100) | No |  | Dirección lógica de la máquina donde se realizó la acción. Admite valores IPV4 e IPV6 |

**Logins**: Tabla en la que se guardan los datos de los usuarios con accesos administrativos u operativos dentro de la plataforma.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Columna | Tipo | Nulo | Predeterminado | Comentarios |
| UserId *(primaria)* | Integer | No |  | Identificación de cada usuario |
| Username | Varchar(50) | No |  | Distinta a la identificación, éste es el nombre de usuario |
| Pass | Varchar(50) | No |  | Representa la contraseña del usuario |
| UserGuid | UniqueIdentifier | No |  | Representa el SALT para proteger las contraseñas frente a posibles ataques de diccionarios |
| Permisos | Integer | No |  | Representa los permisos que va a tener el actor dentro del sistema |

# Conclusión

El diseño lógico y físico de una base de datos, facilita y prepara el trabajo previo a la programación de un sistema dentro de un proyecto, ya que permite identificar de manera adecuada las tablas y sus relaciones, permitiendo reflejar dicha identificación en la etapa de codificación y consultas realizadas hacia la misma base de datos. Es demasiado importante tener un buen diseño de cómo va a estar estructurada la información para llegar a un buen resultado final.

# Script Base de Datos (sin DATA)

use master

drop database ICCP\_Final;

create database ICCP\_Final;

use ICCP\_Final;

create table estadoReclamo(

id int not null primary key,

descripcion varchar(100) not null

);

create table tipoReclamo(

id int not null primary key,

nombre varchar(50) not null

);

create table usuarios(

rut varchar(12) not null primary key,

nombre varchar(50) not null,

apellido varchar(50)not null,

correo varchar(100) not null,

telefono int

);

create table slareclamo(

id int not null primary key IDENTITY(1,1),

fecha\_sla datetime not null

);

create table reclamo(

id int not null primary key,

area int foreign key references tipoReclamo(id) not null,

rut varchar(12) foreign key references usuarios(rut),

fecha datetime not null,

estado int foreign key references estadoReclamo(id),

pdf varchar(100),

comentarios text not null,

sla\_id int foreign key references slareclamo(id),

operador int foreign key references logins(userId)

);

create table logs(

id\_log int not null primary key IDENTITY(1,1),

rut varchar(12) foreign key references usuarios(rut),

tiempo\_log datetime not null,

mensaje text not null,

ip varchar(100) not null

);

create table logins(

userId int IDENTITY(1,1) not null primary key,

username varchar(50) not null,

pass varchar(50) not null,

userGuid UniqueIdentifier,

permisos int not null

);

El anterior script ha sido desarrollado y probado bajo entorno Windows y el motor de bases de datos Microsoft SQL Server Express 2014.

# Script Procedimiento Almacenado “InsertarReclamo”

El siguiente script tiene como finalidad insertar un registro como reclamo en la tabla reclamos, actualizar (si no existe, crear) un registro en la tabla usuarios para identificar a la persona que reclama, un registro en la tabla logs para guardar la información de auditoría y un registro en la tabla SLAReclamos para guardar su fecha y hora de Service Level Agreement.

use ICCP\_Final

create procedure IngresoReclamo

(

@id INTEGER,

@nombre VARCHAR(20),

@apellido VARCHAR(20),

@rut VARCHAR(12),

@email VARCHAR(100),

@telefono INTEGER,

@area INTEGER,

@comentarios text,

@fecha DATETIME,

@pdf VARCHAR(100),

@sla DATETIME,

@ip VARCHAR(100)

)

AS

BEGIN

IF NOT EXISTS (Select \* from usuarios WHERE rut = @rut) INSERT INTO usuarios(rut,nombre,apellido,correo,telefono,permisos) values (@rut,@nombre,@apellido,@email,@telefono,1)

ELSE

UPDATE usuarios SET nombre=@nombre, apellido=@apellido, correo=@email, telefono=@telefono where rut=@rut

END

BEGIN

DECLARE @sla\_id INT;

insert into slareclamo(fecha\_sla) values (@sla)

SET @sla\_id = SCOPE\_IDENTITY()

insert into reclamo(id,area,rut,fecha,estado,pdf,comentarios,sla\_id) values(@id,@area,@rut,@fecha,1,@pdf,@comentarios,@sla\_id);

END

BEGIN

insert into logs(rut,tiempo\_log,mensaje,ip) values(@rut,@fecha,'Se agregó un reclamo',@ip);

END

exec IngresoReclamo @id = 11,

@nombre ='Nombre',

@apellido ='Apellido',

@rut ='19',

@email ='usuario@dominio.com',

@telefono =12345678,

@area =2,

@comentarios ='Lorem ipsum',

@fecha ='2017-11-30T08:38:07',

@pdf ='itsec.pdf',

@sla = '2017-11-30T10:38:07',

@ip ='127.0.0.1'