

Manual de Usuario *Real Estate*

Andrea Gabriela Zelaya Flores 12241001

Dariel Emilio Sevilla Bueso 12241006

Josué Andrés Ham Álvarez 12141190

Víctor Isaías Romero Núñez 12211079

Ing. Julio César Sandoval



Índice

Índice	1
Introducción al Programa	4
Especificación	5
Diseño lógico del modelo relacional en MariaDB (DDL)	5
1. Instalación del Programa	8
2.1 Instrucciones de Acceso:.....	8
1.1.1. Descarga la Aplicación en Java	8
1.1.2. Base de Datos MariaDB	8
1.1.3. Clona el Repositorio	8
1.1.4. Explora y Ejecuta la Aplicación	8
2. Navegación y Diseño de la Interfaz.....	9
2.1 Importación del Programa	9
2.1.1. Importa el Programa y Ejecuta la Aplicación.....	9
2.1.2. Busca el Programa en la Carpeta Guardada.....	9
2.1.3. Explora y Analiza el Código (Opcional)	10
2.2 Ejecución del Programa	11
2.1.1 Ejecución del Programa desde Java/NetBeans	11
2.1.2 Programa en Ejecución	11
3. Funcionalidades Principales	12
3.1 Log-In Fácil e Intuitivo	12
3.1.1 Ingresar a la Sesión	12
3.1.2 Ingresado de Usuario y Contraseña	12
3.1.3 Presionar el Botón de Log-In	13
3.2 Ventanas de Administrador	13
3.2.1 Ventana Principal (Admin_MainScreen).....	13
3.2.1.1 Botón de Opciones	14
3.2.1.2 Botón de Mantenimiento	14
3.2.1.3 Botón de Bitácora	15
3.2.1.4 Botón de Reportes.....	15
3.2.1.5 Botón de Salida	16
3.2.2 Ventana de Mantenimiento	16
3.2.2.1 Botón de Administradores.....	17
3.2.2.2 Botón de Añadir Administrador	18

3.2.2.3	Botón de Compradores	19
3.2.2.4	Botón de Añadir Clientes (Compradores).....	20
3.2.2.5	Botón de Vendedores	21
3.2.2.6	Botón de Añadir Clientes (Vendedores).....	22
3.2.2.7	Botón de Agentes	23
3.2.2.8	Botón de Añadir Agentes	24
3.2.2.9	Botón de Propiedades Vendidas	25
3.2.2.10	Botón de Añadir Propiedades Vendidas	26
3.2.2.11	Botón de Propiedades en Mercado.....	27
3.2.3	Ventana de Bitácora	29
3.2.4	Ventana de Reportes	30
3.2.4.1	Botón de Ventas por Ubicación	31
3.2.4.2	Botón de Ventas por Precio de Propiedad	31
3.2.4.3	Botón de Características.....	32
3.2.4.4	Botón de Año de Valor Total	32
3.2.4.5	Botón de Ventas Promedio	33
3.2.4.6	Botón de Ventas por Agente	33
3.2.4.7	Botón de Ventas por Vendedor.....	34
3.2.4.8	Botón de Cantidad de Propiedades Compradas por Comprador	34
3.3	Ventanas de Vendedor	35
3.3.1	Ventana Principal (Vendedor_MainScreen).....	35
3.3.1.1	Botón de Opciones	35
3.3.1.2	Botón de Propiedades Asignadas al Vendedor.....	36
3.3.1.3	Botón de Propiedades Vendidas por el Vendedor	36
3.3.1.4	Botón de Salida	37
3.3.2	Ventana de Propiedades Asignadas al Vendedor.....	37
3.3.3	Ventana de Propiedades Vendidas por el Vendedor	37
3.4	Ventanas de Comprador	38
3.4.1	Ventana Principal (Comprador_MainScreen)	38
3.4.1.1	Botón de Opciones	38
3.4.1.2	Botón de Propiedad en Mercado	39
3.4.1.3	Botón de Salida	39
4.	Triggers.....	40
4.1	Registrar Agente.....	40

4.2	Registrar Comprador	40
4.3	Registrar Propiedad en Mercado.....	41
4.4	Registrar Propiedad Vendida	41
4.5	Registrar Usuario	42
4.6	Registrar Vendedor	42
4.7	Registrar Agente.....	43
4.8	Registrar Comprador	43
4.9	Registrar Propiedades en Mercado.....	44
4.10	Registrar Propiedades Vendidas.....	44
4.11	Registrar usuarios.....	45
4.12	Registrar Vendedores	45
4.13	Registrar Agente.....	46
4.14	Registrar Compradores.....	46
4.15	Registrar Propiedad en Mercado.....	47
4.16	Registrar Propiedad Vendida	47
4.17	Registrar Usuario	48
4.18	Registrar Vendedor	48
5.	Views	49
5.1	Ventas de Agente por Promedio Anual.....	49
5.2	Valor de Venta por Agente.....	49
5.3	Agentes por Fecha de Venta.....	50
5.4	Agente por Año de Venta	50
5.5	Compras por Comprador	51
5.6	Ventas por Agente	51
5.7	Ventas por Característica	52
5.8	Ventas por Precio.....	52
5.9	Ventas por Ubicación.....	53
5.10	Ventas por Vendedor	53

Introducción al Programa

Bienvenido a la Oficina de REALESTATE... *Simplificando tus Transacciones Inmobiliarias*

En el acelerado mundo del mercado inmobiliario, donde cada transacción cuenta y la eficiencia es clave, damos la bienvenida a nuestra aplicación de la Oficina de **RealEstate**. Diseñada para brindar una solución integral a tus necesidades de gestión y seguimiento, nuestra aplicación está diseñada para hacer que tu experiencia en el mercado inmobiliario sea más fluida, eficiente y efectiva.

➤ Una Solución Integral para Agentes, Compradores y Vendedores

Nuestra aplicación está diseñada para satisfacer las necesidades de una amplia gama de usuarios, desde agentes de bienes raíces hasta compradores y vendedores de propiedades. Con una interfaz intuitiva y funciones potentes, ofrecemos herramientas avanzadas para facilitar cada paso del proceso, desde la búsqueda y la presentación de propiedades hasta el cierre de transacciones exitosas.

➤ Gestión de Propiedades Simplificada

Ya sea que estés buscando comprar o vender una propiedad siendo un agente, comprador, vendedor, o Administrador nuestra aplicación te ofrece todas las herramientas que necesitas para tener éxito en el mercado inmobiliario. Con funciones de búsqueda avanzadas, seguimiento de propiedades en el mercado y generación de informes detallados, bitácoras en tiempo real, y vistas personalizadas a tu medida estamos aquí para ayudarte en cada paso del camino de Bienes Raíces.

➤ Estadísticas y Análisis Avanzados

Nuestra aplicación no solo te ayuda a gestionar tus propiedades de manera eficiente, sino que también te brinda información valiosa sobre el rendimiento del mercado y el éxito de tus transacciones. Con funciones de análisis avanzado, puedes obtener estadísticas detalladas sobre las tendencias del mercado, propiedades disponibles en el mercado, del mismo modo las propiedades venidas y sobre todo el rendimiento de los agentes, y mucho más... esto le permite tomar decisiones informadas y estratégicas.

➤ Soporte Expertos y Actualizaciones Continuas

En la Oficina de RealEstate, nos enorgullecemos de ofrecer un servicio excepcional a nuestros usuarios. Nuestro equipo de expertos está siempre disponible para brindarte soporte y asistencia cuando lo necesites contando con un grupo de expertos en el área de Front-End, Back-End y Data Base Administrators. Además, estamos comprometidos a mejorar continuamente nuestra aplicación con actualizaciones regulares y nuevas características para satisfacer tus necesidades cambiantes en el mercado.

Especificación

La oficina de bienes raíces enfrenta un desafío crítico en la gestión descentralizada de datos, lo que ha llevado a ineficiencias operativas y a una pérdida de oportunidades clave en el mercado inmobiliario. Actualmente, la información relativa a agentes, compradores, vendedores, propiedades y transacciones se encuentra dispersa en diversos formatos y plataformas, dificultando su acceso eficiente y su análisis. La implementación de una base de datos relacional utilizando MariaDB busca abordar esta problemática y establecer una infraestructura robusta para el manejo integral de datos.

Diseño lógico del modelo relacional en MariaDB (DDL)

A continuación, se presenta la creación de casa una de las tablas usadas en la base de datos.

Tabla Usuarios:

```
CREATE TABLE IF NOT EXISTS USUARIOS(
    username VARCHAR(20) PRIMARY KEY,
    passwrd VARCHAR(10) NOT NULL,
    noldentidad NUMERIC(8) NOT NULL,
    activo DECIMAL(1) NOT NULL,
    rol VARCHAR(9) NOT NULL
);
```

Tabla Agentes:

```
CREATE TABLE IF NOT EXISTS AGENTES(
    noldentidad NUMERIC(8) PRIMARY KEY,
    nombre VARCHAR(20) NOT NULL,
    direccion VARCHAR(30),
    celular NUMERIC(8),
    telefonoOficina NUMERIC(8) NOT NULL
);
```

Tabla Compradores:

```
CREATE TABLE COMPRADORES(
    noldentidad NUMERIC(8) PRIMARY KEY,
```

```

nombre VARCHAR(20) NOT null,
direccion VARCHAR(30),
celular NUMERIC(8)
);

```

Tabla Vendedores:

```

CREATE TABLE VENDEDORES(
noldentidad NUMERIC(8) PRIMARY KEY,
nombre VARCHAR(20) NOT null,
direccion VARCHAR(30),
celular NUMERIC(8)
);

```

Tabla Propiedades_en_mercado:

```

CREATE TABLE PROPIEDADES_EN_MERCADO(
idPropiedad INT PRIMARY KEY,
nombre VARCHAR(30) NOT NULL,
ciudad VARCHAR(20) NOT NULL,
direccion VARCHAR(30) NOT NULL,
cantidadDormitorios INT NOT NULL,
caracteristicas VARCHAR(100) NOT NULL,
precio NUMERIC(13, 2) NOT NULL,
fechaPublicacion DATE NOT NULL,
noldentidad_Agente NUMERIC(8) NOT NULL,
noldentidad_Vendedor NUMERIC(8) NOT NULL,
imgPath VARCHAR(200),
FOREIGN KEY (noldentidad_Agente) REFERENCES AGENTES(noldentidad),
FOREIGN KEY (noldentidad_Vendedor) REFERENCES VENDEDORES(noldentidad)
);

```

Tabla Propiedades_vendidas:

```
CREATE TABLE PROPIEDADES_VENDIDAS(
    idPropiedad INT PRIMARY KEY,
    nombre VARCHAR(30) NOT NULL,
    ciudad VARCHAR(20) NOT NULL,
    direccion VARCHAR(30) NOT NULL,
    cantidadDormitorios INT NOT NULL,
    caracteristicas VARCHAR(100) NOT NULL,
    precio NUMERIC(13, 2) NOT NULL,
    fechaPublicacion DATE NOT NULL,
    fechaVenta DATE NOT NULL,
    noldentidad_Agente NUMERIC(8) NOT NULL,
    noldentidad_Vendedor NUMERIC(8) NOT NULL,
    noldentidad_Comprador NUMERIC(8) NOT NULL,
    comisionVenta NUMERIC(13, 2) NOT NULL,
    imgPath VARCHAR(200),
    FOREIGN KEY (noldentidad_Agente) REFERENCES AGENTES(noldentidad),
    FOREIGN KEY (noldentidad_Vendedor) REFERENCES VENDEDORES(noldentidad),
    FOREIGN KEY (noldentidad_Comprador) REFERENCES COMPRADORES(noldentidad)
);
```

Tabla Bitacora:

```
CREATE TABLE bitacora (
    idOperacion INT(11) NOT NULL,
    username VARCHAR(20) NOT NULL,
    operacion VARCHAR(10) NOT NULL,
    tabla VARCHAR(25) NOT NULL,
    id INT(11) NOT NULL,
    fecha DATE NOT NULL,
    hora TIME NOT NULL,
);
```

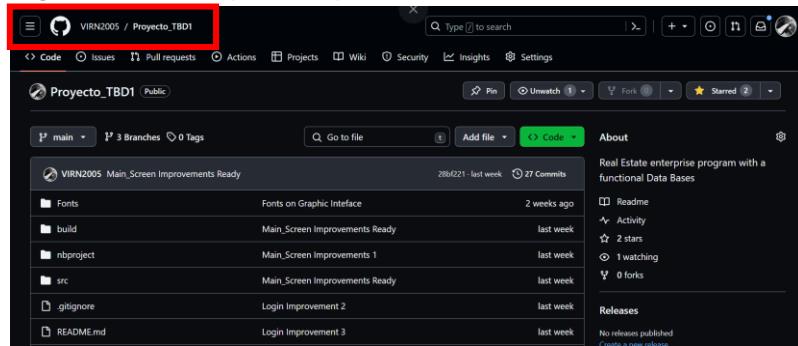
1. Instalación del Programa

Para acceder a nuestra aplicación de la Oficina de Bienes Raíces, simplemente descarga la aplicación en Java desde nuestro repositorio en GitHub y asegúrate de tener la base de datos MariaDB lista para su uso.

2.1 Instrucciones de Acceso:

1.1.1. Descarga la Aplicación en Java

Dirígete a nuestro repositorio en GitHub en el siguiente enlace: GitHub - Proyecto_TBD1 (https://github.com/VIRN2005/Proyecto_TBD1/tree/master). Desde allí, puedes clonar el repositorio o descargar el archivo zip de la aplicación.



1.1.2. Base de Datos MariaDB

Nuestra aplicación utiliza una base de datos MariaDB para almacenar y gestionar los datos de bienes raíces. Asegúrate de tener instalado MariaDB en tu sistema y preparado para su uso.

1.1.3. Clona el Repositorio

Si prefieres clonar el repositorio, puedes hacerlo ejecutando el siguiente comando en tu terminal:

```
Victor@Victor MINGW64 /c/- UNITEC/Ing. Sistemas/2024 - Periodo 1/Teoría de Base de Datos I/- Proyect/Proyecto_TBD1 (master)
$ git clone -b master https://github.com/VIRN2005/Proyecto_TBD1
```

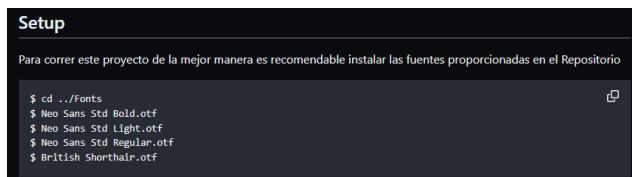
Esto descargará el código fuente de nuestra aplicación en tu máquina local.

1.1.4. Explora y Ejecuta la Aplicación

Una vez que hayas descargado la aplicación, explora el código fuente y asegúrate de cumplir con los requisitos necesarios para ejecutarla.

Abre la aplicación en tu entorno de desarrollo Java y ejecútala para comenzar a utilizarla.

Recuerda seguir el README.md de GitHub



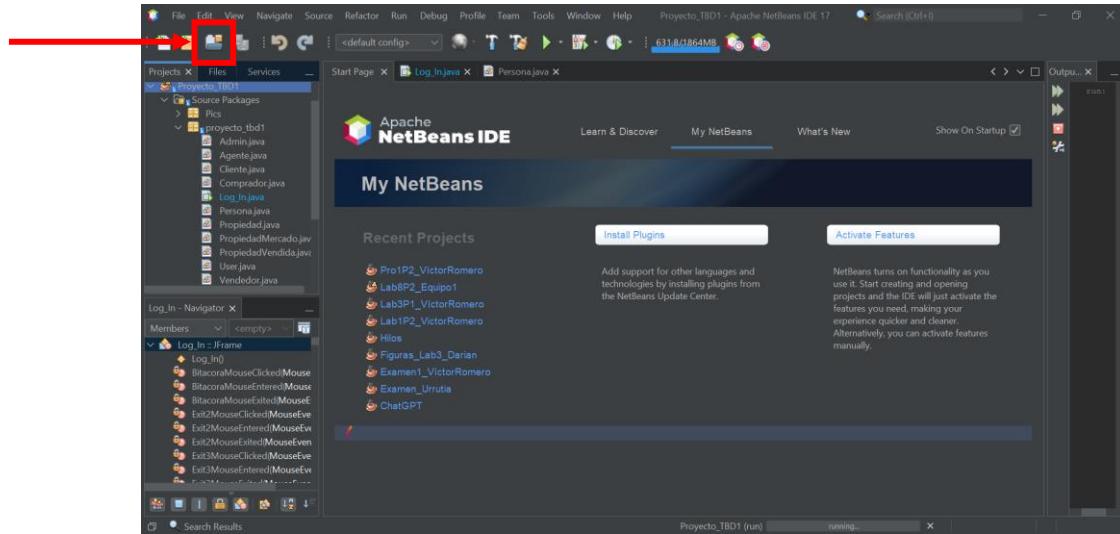
2. Navegación y Diseño de la Interfaz

Luego de haber instalado y ejecutado la aplicación de la Oficina de RealEstate, serás recibido con una pantalla de inicio de sesión. Aquí es donde podrás ingresar tus credenciales para acceder a las funciones de la aplicación.

2.1 Importación del Programa

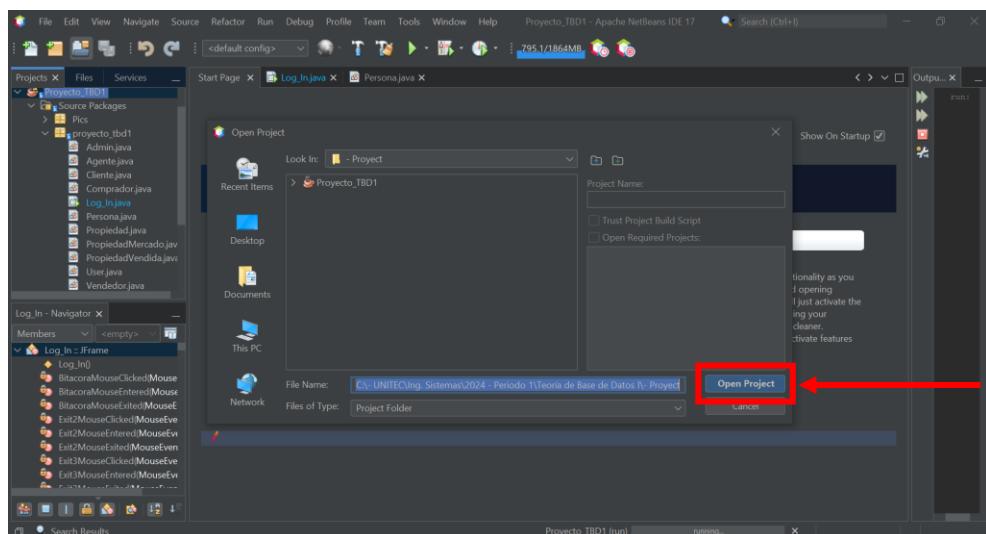
2.1.1. Importa el Programa y Ejecuta la Aplicación

Toca el Tercer Botón en la parte superior derecha (Se representa en la Imagen) para abrir el proyecto.



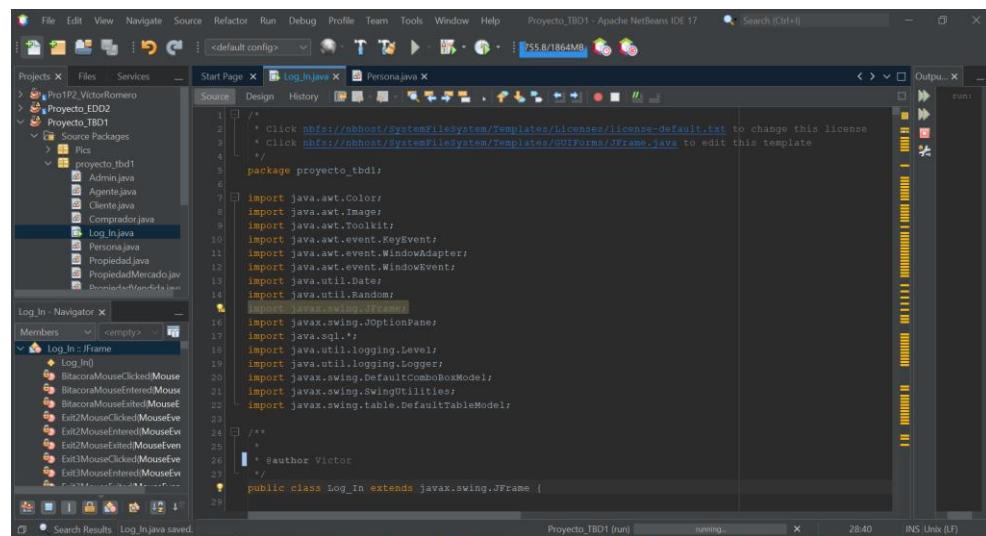
2.1.2. Busca el Programa en la Carpeta Guardada

Ya encontrando la ubicación del archivo presionamos el botón de **Open Project** o Abrir Proyecto para poder importar los datos Clonados desde GitHub.



2.1.3. Explora y Analiza el Código (Opcional)

Luego de haber importado el proyecto puedes revisar si el código se acopla a tu necesidad al momento de administrar las propiedades dentro de la Base de datos



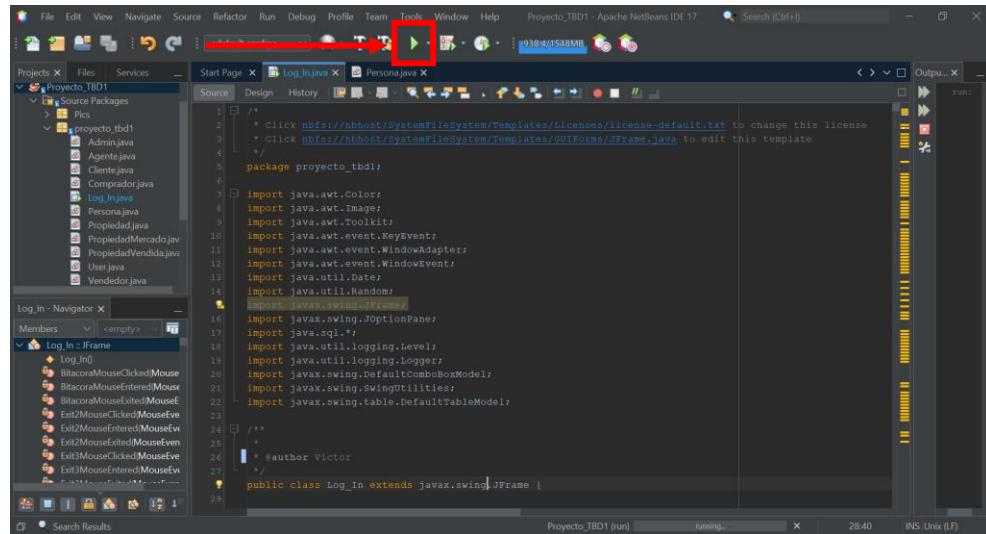
The screenshot shows the Apache NetBeans IDE interface. The title bar reads "Proyecto_TBD1 - Apache NetBeans IDE 17". The left sidebar displays the "Projects" tree, which includes "Proyecto_EDD2", "Proyecto TBD1" (selected), and "Pics". Inside "Proyecto TBD1", there are several source packages: "Admin.java", "Agente.java", "Cliente.java", "Comprador.java", "Log_In.java" (selected), "Persona.java", "Propiedad.java", and "PropiedadMercado.java". The main editor area shows the Java code for "Log_In.java". The code imports various Java packages such as java.awt, javax.swing, and java.util. It defines a class "Log_In" that extends "javax.swing.JFrame". The code includes several mouse event listeners (MouseAdapter, MouseListener) and a constructor. The right side of the interface has a vertical toolbar and a status bar at the bottom indicating "Projeto_TBD1 (run)" and "running...".

```
1  /*
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3  * Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to edit this template
4  */
5  package proyecto_tbd1;
6
7  import java.awt.Color;
8  import java.awt.Image;
9  import java.awt.Toolkit;
10 import java.awt.event.KeyEvent;
11 import java.awt.event.WindowAdapter;
12 import java.awt.event.WindowEvent;
13 import java.util.Date;
14 import java.util.Random;
15
16 import javax.swing.JOptionPane;
17 import java.sql.*;
18 import java.util.logging.Level;
19 import java.util.logging.Logger;
20 import javax.swing.DefaultComboBoxModel;
21 import javax.swing.SwingUtilities;
22 import javax.swing.table.DefaultTableModel;
23
24 /**
25  * @author Victor
26  */
27 public class Log_In extends javax.swing.JFrame {
28
29 }
```

2.2 Ejecución del Programa

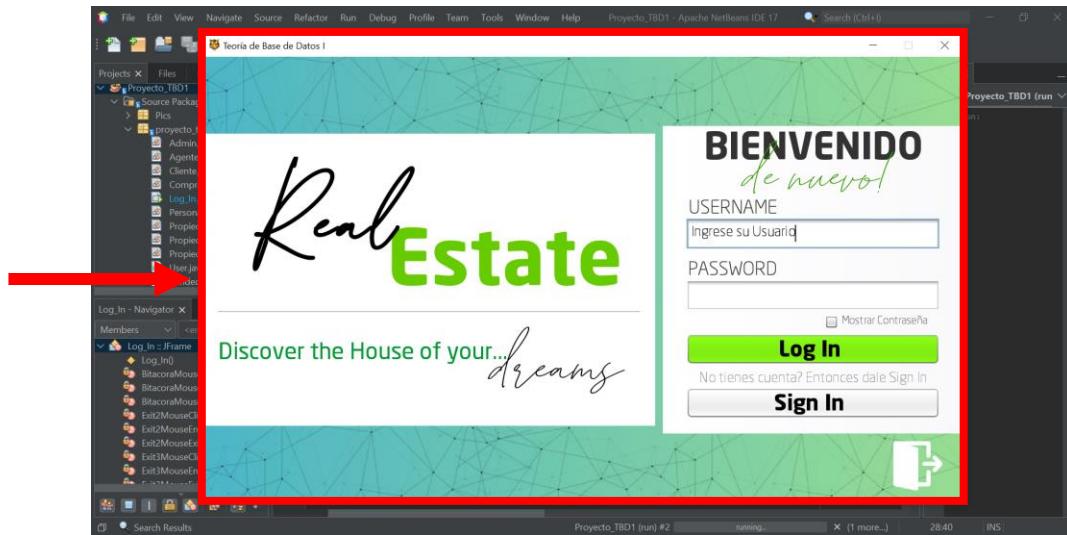
2.1.1 Ejecución del Programa desde Java/NetBeans

Presiona el Botón donde de Corre el Programa o simplemente toca la tecla F6 para ejecutar el programa en pantalla



2.1.2 Programa en Ejecución

Cuando el programa muestra la ejecución del Log-In entonces el programa estará listo para ser usado



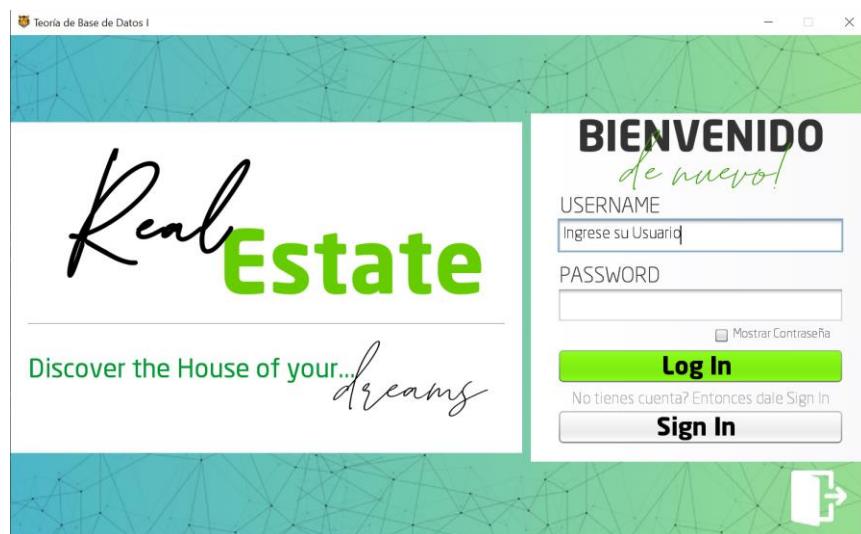
3. Funcionalidades Principales

Nuestra aplicación de la Oficina de Bienes Raíces está diseñada para brindarte todas las herramientas que necesitas para gestionar tus transacciones inmobiliarias de manera eficiente y efectiva. Con una amplia gama de funcionalidades diseñadas específicamente para satisfacer las necesidades del mercado inmobiliario actual, estamos aquí para simplificar cada paso del proceso y ayudarte a alcanzar tus objetivos.

3.1 Log-In Fácil e Intuitivo

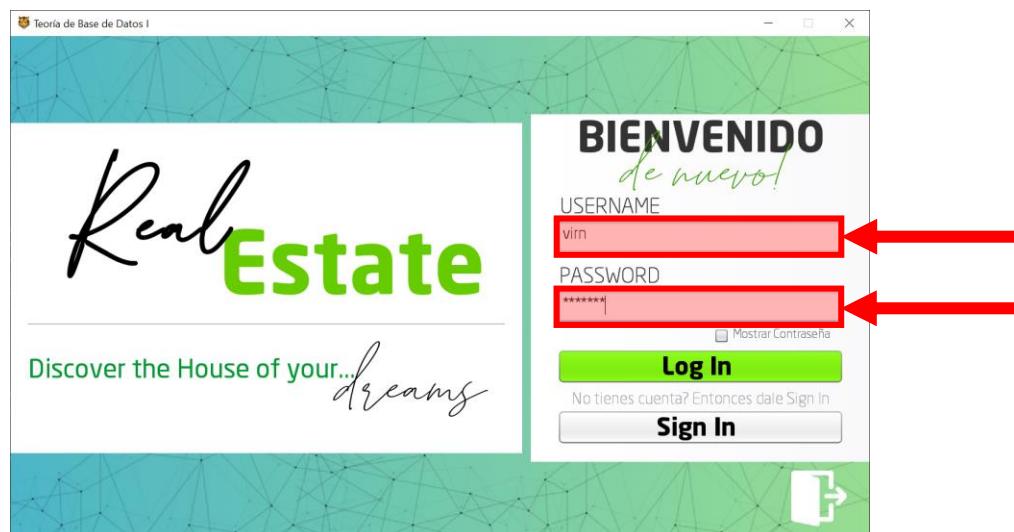
3.1.1 Ingresar a la Sesión

El sistema de Log-In genera un proceso de Sesión Sencillo y simplificado, acompañado de una interfaz intuitiva y sobre todo, lo mas importante seguridad y privacidad de extremo a extremo.



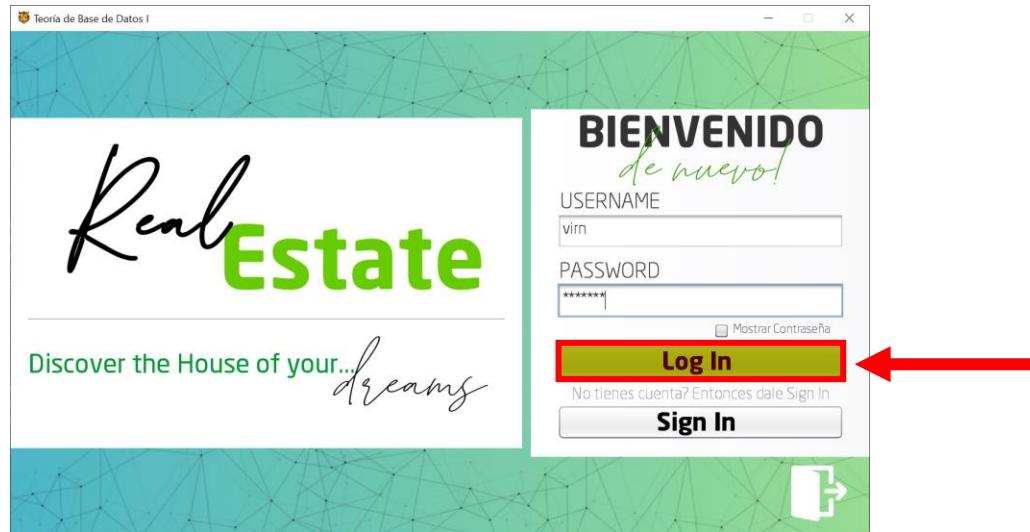
3.1.2 Ingresado de Usuario y Contraseña

Hemos implementado un proceso de inicio de sesión validado que garantiza la seguridad y confidencialidad de tu información personal. Puedes estar seguro de que tu privacidad está protegida y que tus datos están en buenas manos mientras utilizas nuestra plataforma.



3.1.3 Presionar el Botón de Log-In

Al presionar el botón de Log-In entramos a la sesión con las especificaciones dada a cada usuario, ya sea administrador, agente, vendedor o comprador. Separando los datos para cada usuario en específico.



3.2 Ventanas de Administrador

En la ventana del administrador, los usuarios con privilegios asignados en este rango tienen acceso a casi todas las funcionalidades del programa, lo que les permite gestionar eficientemente la información y las transacciones relacionadas con la oficina de bienes raíces. Desde la gestión de agentes, compradores y vendedores hasta el seguimiento de propiedades y la generación de informes, la ventana del administrador ofrece un conjunto completo de herramientas para facilitar una administración efectiva del sistema.

3.2.1 Ventana Principal (Admin_MainScreen)

Aquí se muestra la pantalla inicial de esta función englobando todas las opciones necesarias



3.2.1.1 Botón de Opciones

Al presionar el **BOTÓN DE OPCIONES** se muestran las principales funcionalidades divididas en los diferentes paneles programados para la optimización del programa



3.2.1.2 Botón de Mantenimiento

Al presionar el Botón de Mantenimiento se muestran cada una de las tablas generadas (Administradores, Agentes, Vendedores, Compradores, Propiedades en Mercado y Propiedades Vendidas) Cada tabla incluye su respectivo Agregado, Eliminado y Modificado



3.2.1.3 Botón de Bitácora

Muestra las acciones realizadas por cada Administrador en una Tabla Resumen



3.2.1.4 Botón de Reportes

Muestra cada uno de los Reportes separados por vistas específicas dependiendo de lo solicitado



3.2.1.5 Botón de Salida

El botón "Salir" ofrece a los usuarios la posibilidad de cerrar sesión y regresar a la pantalla de inicio de sesión. Al hacer clic en este botón, se finaliza la sesión actual del usuario y se le redirige de manera segura a la pantalla de inicio de sesión, donde pueden ingresar nuevamente sus credenciales para acceder a la aplicación. Esta funcionalidad es esencial para garantizar la seguridad de la cuenta y proteger la privacidad de los usuarios al finalizar su sesión cuando ya no la necesitan. Además, proporciona una manera conveniente y rápida de cambiar de usuario o cerrar la sesión de forma segura cuando se termina de utilizar la aplicación.



3.2.2 Ventana de Mantenimiento

Aquí se muestran las Tablas Resumen de cada uno de los Administradores, Agentes, Vendedores, Compradores, Propiedades Vendidas y Propiedades en Mercado.

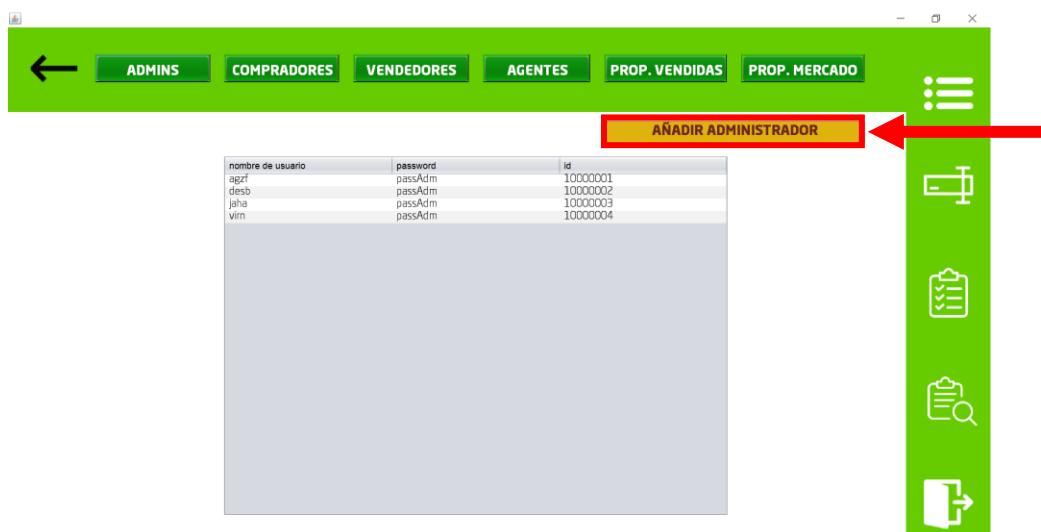
nombre de usuario	password	id
agz1	passAdm	10000001
desb	passAdm	10000002
jaha	passAdm	10000003
virm	passAdm	10000004

3.2.2.1 Botón de Administradores

En esta ventana se muestra una tabla que enumera a cada administrador junto con sus atributos correspondientes, como el nombre de usuario, la contraseña y el ID.



Del mismo modo, se puede añadir administradores presionando el botón "Añadir Administrador". Al hacer clic en este botón, aparecerá un nuevo panel donde se deberá llenar la respectiva información del administrador.



Del mismo modo se pueden eliminar o Editar los Administradores tocando el botón secundario sobre el usuario al que se le deseé realizar la acción.

nombre de usuario	password	id
agzf		10000001
desb		10000002
jaha		10000003
virm		10000004

3.2.2.2 Botón de Añadir Administrador

Aquí se muestra la ventana que se mencionó en el punto 3.2.2.1 en el cual se utiliza para crear un nuevo administrador y agregarlo en la tabla, donde se debe agregar los atributos necesitados



Al haber ingresado los valores correspondientes se presiona el botón de Crear administrador para ingresar los valores en la Tabla



3.2.2.3 Botón de Compradores

Al presionar el botón de compradores aparecerá la tabla de usuarios compradores y sus atributos.

idIdentidad	nombre	dirección	celular
1	prueba5	123 Main St	12345678
30000001	Carlos Garcia	456 Elm St	23456789
30000002	Maria Hernandez	789 Oak St	34567890
30000003	Martin Neilren	101 Pine St	45678901
30000004	Ana Lopez	202 Maple St	56789012
30000005	Pedro Perez	303 Cedar St	67890123
30000006	Laura Rodriguez	404 Birch St	78901234
30000007	Diego Gonzalez	505 Walnut St	89012345
30000008	Sofia Ramirez	606 Cherry St	90123456
30000009	Alejandro Sanchez	707 Orange St	12345678
30000010	Valelia Flores	808 Banana St	23456789
40000001	Sara Martinez	123 Elm St	12345678
40000002	Diego Gonzalez	456 Oak St	23456789
40000003	Laura Rodriguez	789 Pine St	34567890
40000004	Sofia Ramirez	101 Maple St	45678901
40000005	Alejandro Sanchez	202 Cedar St	56789012
40000006	Valelia Flores	303 Birch St	67890123
40000007	Gabriel Torres	404 Walnut St	78901234
40000008	Luis Morales	505 Cherry St	89012345
40000009	Maria Vargas	606 Pineapple St	90123456
40000010	Pablo Cruz	707 Orange St	12344321
40000011	Elena Gomez	808 Banana St	23455432
40000012	Carmen Reyes	909 Grape St	34565543
40000013	Ricardo Diaz	1010 Apple St	45677654
40000014	Daniela Castro	1111 Pear St	56788765
40000015	Fernando Ortiz	1212 Cherry St	67899876
40000016	Javier Mendoza	1313 Plum St.	78900987
40000017	Marta Arellan	1414 Lemon St.	89012098

Del mismo modo, se puede añadir Compradores presionando el botón "Añadir Cliente". Al hacer clic en este botón, aparecerá un nuevo panel donde se deberá llenar la respectiva información del comprador.

idIdentidad	nombre	dirección	celular
1	prueba5	123 Main St	12345678
30000001	Carlos Garcia	456 Elm St	23456789
30000002	Maria Hernandez	789 Oak St	34567890
30000003	Martin Neilren	101 Pine St	45678901
30000004	Ana Lopez	202 Maple St	56789012
30000005	Pedro Perez	303 Cedar St	67890123
30000006	Laura Rodriguez	404 Birch St	78901234
30000007	Diego Gonzalez	505 Walnut St	89012345
30000008	Sofia Ramirez	606 Cherry St	90123456
30000009	Alejandro Sanchez	707 Orange St	12344321
30000010	Valelia Flores	808 Banana St	23456789
40000001	Sara Martinez	123 Elm St	12345678
40000002	Diego Gonzalez	456 Oak St	23456789
40000003	Laura Rodriguez	789 Pine St	34567890
40000004	Sofia Ramirez	101 Maple St	45678901
40000005	Alejandro Sanchez	202 Cedar St	56789012
40000006	Valelia Flores	303 Birch St	67890123
40000007	Gabriel Torres	404 Walnut St	78901234
40000008	Luis Morales	505 Cherry St	89012345
40000009	Maria Vargas	606 Pineapple St	90123456
40000010	Pablo Cruz	707 Orange St	12344321
40000011	Elena Gomez	808 Banana St	23455432
40000012	Carmen Reyes	909 Grape St	34565543
40000013	Ricardo Diaz	1010 Apple St	45677654
40000014	Daniela Castro	1111 Pear St	56788765
40000015	Fernando Ortiz	1212 Cherry St	67899876
40000016	Javier Mendoza	1313 Plum St.	78900987
40000017	Marta Arellan	1414 Lemon St.	89012098

3.2.2.4 Botón de Añadir Clientes (Compradores)

Aquí se muestra la ventana que se mencionó en el punto 3.2.2.3 en el cual se utiliza para crear un nuevo Cliente y agregarlo en la tabla, donde se debe agregar los atributos necesitados

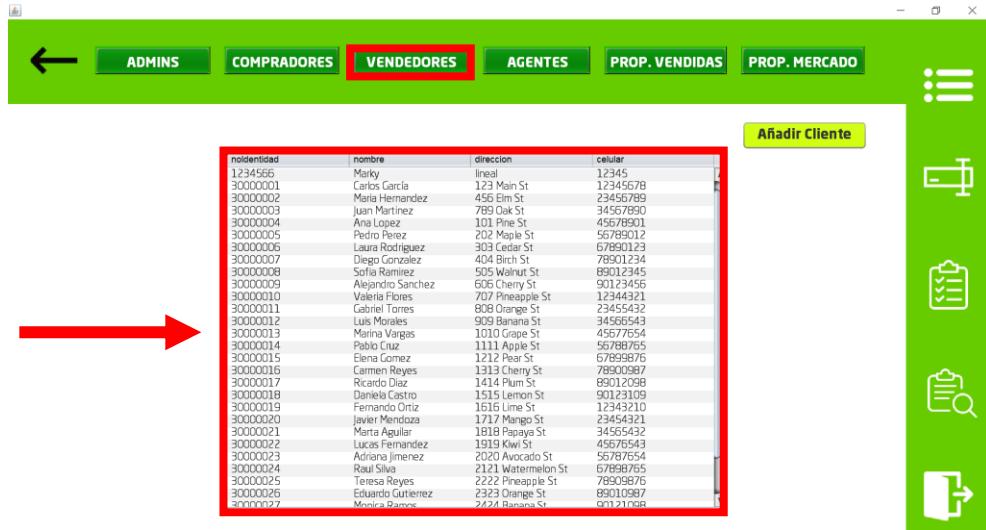


Al haber ingresado los valores correspondientes se presiona el botón de Crear cliente para ingresar los valores en la Tabla



3.2.2.5 Botón de Vendedores

Al presionar el botón de vendedores aparecerá la tabla de usuarios vendedores y sus atributos.



The screenshot shows a software interface with a green header bar containing tabs: 'ADMINS', 'COMPRADORES', 'VENDEDORES' (which is highlighted in red), 'AGENTES', 'PROP. VENDIDAS', and 'PROP. MERCADO'. To the right of the tabs is a vertical toolbar with icons for sorting, filtering, search, and export. Below the header is a table with columns: 'noidentidad', 'nombre', 'direccion', and 'celular'. The table contains numerous rows of data, each representing a vendor with their name, address, and phone number.

Del mismo modo, se puede añadir Vendedores presionando el botón "Añadir Cliente". Al hacer clic en este botón, aparecerá un nuevo panel donde se deberá llenar la respectiva información del Vendedor.



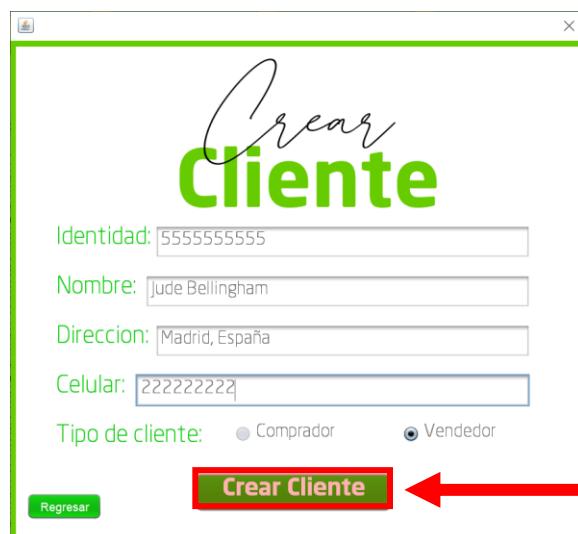
This screenshot is identical to the one above, showing the 'VENDEDORES' tab selected. A red arrow highlights the 'Añadir Cliente' button in the top right corner of the main content area, which is intended to open a new panel for adding a new vendor.

3.2.2.6 Botón de Añadir Clientes (Vendedores)

Aquí se muestra la ventana que se mencionó en el punto 3.2.2.5 en el cual se utiliza para crear un nuevo Cliente y agregarlo en la tabla, donde se debe agregar los atributos necesitados



Al haber ingresado los valores correspondientes se presiona el botón de Crear cliente para ingresar los valores en la Tabla



3.2.2.7 Botón de Agentes

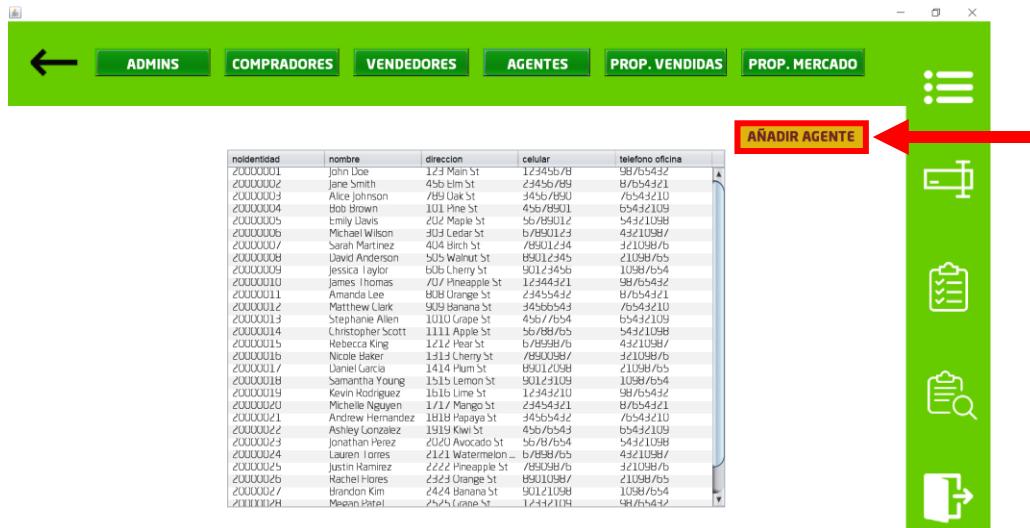
Al presionar el botón de vendedores aparecerá la tabla de usuarios vendedores y sus atributos.



The screenshot shows a green-themed application window with a navigation bar at the top. The 'AGENTES' tab is highlighted in red. Below the tabs is a table with columns: 'nóIdentidad', 'nombre', 'dirección', 'celular', and 'telefono oficina'. The table contains 24 rows of data. To the right of the table is a vertical sidebar with icons for file operations: add, edit, delete, search, and export.

nóIdentidad	nombre	dirección	celular	telefono oficina
20XXXXXX1	John Doe	123 Main St	12345678	98765432
20XXXXXX2	Jane Smith	456 Elm St	23456789	87654321
20XXXXXX3	Alice Johnson	789 Oak St	34567890	76543210
20XXXXXX4	Bob Brown	101 Pine St	45678901	65432109
20XXXXXX5	Emily Davis	202 Birch St	56789012	54321098
20XXXXXX6	Michael Wilson	303 Cedar St	67890123	43210987
20XXXXXX7	Sarah Martinez	404 Birch St	78901234	32109876
20XXXXXX8	David Anderson	505 Walnut St	89012345	21098765
20XXXXXX9	Jessica Taylor	606 Cherry St	90123456	10987654
20XXXXXX10	James Thomas	707 Pineapple St	12344321	98765432
20XXXXXX11	Amanda Lee	808 Orange St	23454321	87654321
20XXXXXX12	Matthew Clark	909 Banana St	34565432	76543210
20XXXXXX13	Stephanie Allen	1010 Grape St	45676543	65432109
20XXXXXX14	Christopher Scott	1111 Apple St	56787654	54321098
20XXXXXX15	Rebecca King	1212 Chestnut St	67898765	43210987
20XXXXXX16	Nicole Baker	1313 Chestnut St	78909876	32109875
20XXXXXX17	Daniel Lancia	1414 Plum St	89010987	21098764
20XXXXXX18	Samantha Young	1515 Lemon St	90121098	10987653
20XXXXXX19	Kevin Rodriguez	1616 Lime St	12342109	98765432
20XXXXXX20	Michelle Nguyen	1717 Mango St	23454210	87654321
20XXXXXX21	Andrew Hernandez	1818 Papaya St	34565432	76543210
20XXXXXX22	Ashley Gonzalez	1919 Kiwi St	45676432	65432109
20XXXXXX23	Omega Perez	2020 Avocado St	56787643	54321098
20XXXXXX24	Jonathan Perez	2121 Watermelon St	67898765	43210987
20XXXXXX25	Lauren Torres	2222 Pineapple St	78909876	32109875
20XXXXXX26	Justin Ramirez	2323 Orange St	89010987	21098764
20XXXXXX27	Rachel Flores	2424 Banana St	90121098	10987653
20XXXXXX28	Brandon Kim	2525 Orange St	12342109	98765432
20XXXXXX29	Megan Patel	2626 Orange St	23454210	87654321

Del mismo modo, se puede añadir Agentes presionando el botón "Añadir Agente". Al hacer clic en este botón, aparecerá un nuevo panel donde se deberá llenar la respectiva información del Agente.



This screenshot shows the same application interface as the previous one, but the 'AÑADIR AGENTE' button in the top right corner of the main content area is highlighted with a red box. A red arrow points to this button from the left side of the image.

nóIdentidad	nombre	dirección	celular	telefono oficina
20XXXXXX1	John Doe	123 Main St	12345678	98765432
20XXXXXX2	Jane Smith	456 Elm St	23456789	87654321
20XXXXXX3	Alice Johnson	789 Oak St	34567890	76543210
20XXXXXX4	Bob Brown	101 Pine St	45678901	65432109
20XXXXXX5	Emily Davis	202 Birch St	56789012	54321098
20XXXXXX6	Michael Wilson	303 Cedar St	67890123	43210987
20XXXXXX7	Sarah Martinez	404 Birch St	78901234	32109876
20XXXXXX8	David Anderson	505 Walnut St	89012345	21098765
20XXXXXX9	Jessica Taylor	606 Cherry St	90123456	10987654
20XXXXXX10	James Thomas	707 Pineapple St	12344321	98765432
20XXXXXX11	Amanda Lee	808 Orange St	23455432	87654321
20XXXXXX12	Matthew Clark	909 Banana St	34566543	76543210
20XXXXXX13	Stephanie Allen	1010 Grape St	45676543	65432109
20XXXXXX14	Christopher Scott	1111 Apple St	56787654	54321098
20XXXXXX15	Rebecca King	1212 Chestnut St	67898765	43210987
20XXXXXX16	Nicole Baker	1313 Chestnut St	78909876	32109875
20XXXXXX17	Daniel Lancia	1414 Plum St	89010987	21098764
20XXXXXX18	Samantha Young	1515 Lemon St	90121098	10987653
20XXXXXX19	Kevin Rodriguez	1616 Lime St	12342109	98765432
20XXXXXX20	Michelle Nguyen	1717 Mango St	23454210	87654321
20XXXXXX21	Andrew Hernandez	1818 Papaya St	34565432	76543210
20XXXXXX22	Ashley Gonzalez	1919 Kiwi St	45676432	65432109
20XXXXXX23	Omega Perez	2020 Avocado St	56787643	54321098
20XXXXXX24	Jonathan Perez	2121 Watermelon St	67898765	43210987
20XXXXXX25	Lauren Torres	2222 Pineapple St	78909876	32109875
20XXXXXX26	Justin Ramirez	2323 Orange St	89010987	21098764
20XXXXXX27	Rachel Flores	2424 Banana St	90121098	10987653
20XXXXXX28	Brandon Kim	2525 Orange St	12342109	98765432
20XXXXXX29	Megan Patel	2626 Orange St	23454210	87654321

3.2.2.8 Botón de Añadir Agentes

Aquí se muestra la ventana que se mencionó en el punto 3.2.2.7 en el cual se utiliza para crear un nuevo Agente y agregarlo en la tabla, donde se debe agregar los atributos necesitados

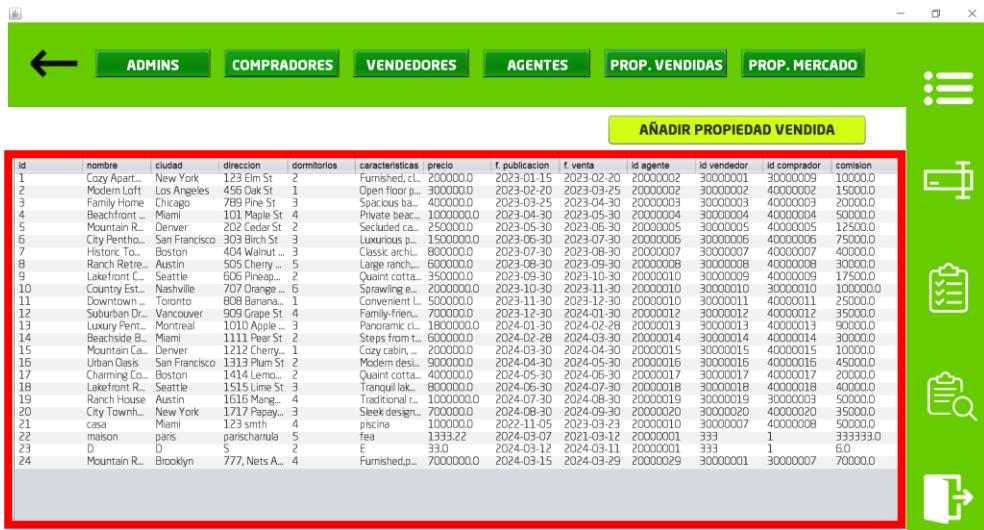
The screenshot shows a window titled "Crear Agente". Inside, there are five input fields labeled "Identidad:", "Nombre:", "Direccion:", "Celular:", and "Telefono Oficina:". Each of these labels and their corresponding input fields is highlighted with a red rectangular box. Below the input fields is a green button labeled "Crear Agente". A large green border surrounds the entire window. A red arrow points from the right side towards the "Crear Agente" button.

Al haber ingresado los valores correspondientes se presiona el botón de Crear Agente para ingresar los valores en la Tabla

The screenshot shows the same "Crear Agente" window as the previous one, but now all the input fields are empty (white). The labels "Identidad:", "Nombre:", "Direccion:", "Celular:", and "Telefono Oficina:" are visible above their respective empty fields. Below the input fields is a green button labeled "Crear Agente". This button is highlighted with a red rectangle and has a red arrow pointing to it from the right.

3.2.2.9 Botón de Propiedades Vendidas

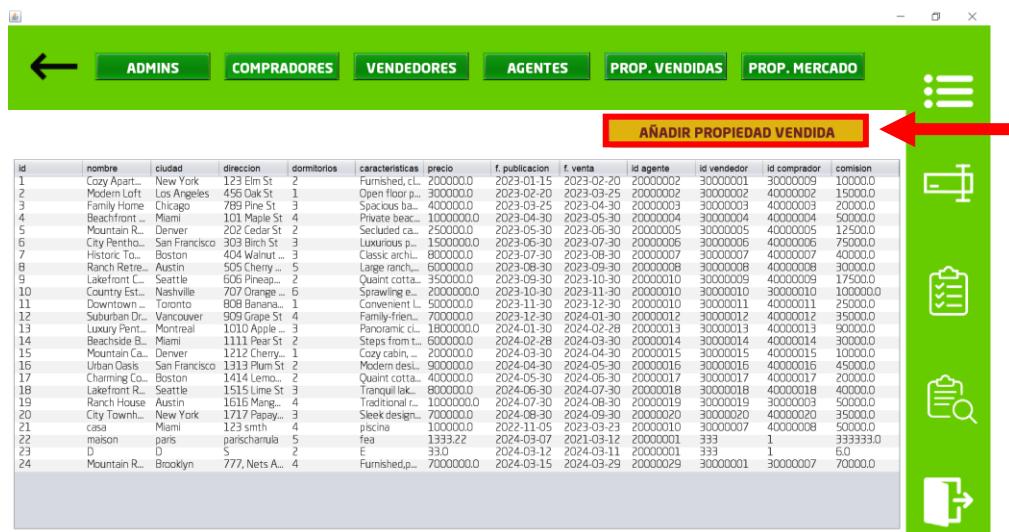
Al presionar el botón de propiedades vendidas el cual aparecerá la tabla de Propiedades las cuales han sido vendidas y sus atributos específicos.



The screenshot shows a software interface with a green header bar containing navigation tabs: ADMINS, COMPRADORES, VENDEDORES, AGENTES, PROP. VENDIDAS (highlighted in yellow), and PROP. MERCADO. Below the header is a table titled 'AÑADIR PROPIEDAD VENDIDA'. The table has columns for: id, nombre, ciudad, dirección, dormitorios, características, precio, f_publicación, f_venta, id_agente, id_vendedor, id_comprador, and comisión. The table contains 24 rows of property data. To the right of the table are several icons: a battery, a clipboard, a magnifying glass, and a file folder. A red arrow points from the left towards the 'AÑADIR PROPIEDAD VENDIDA' button.

AÑADIR PROPIEDAD VENDIDA												
id	nombre	ciudad	dirección	dormitorios	características	precio	f_publicación	f_venta	id_agente	id_vendedor	id_comprador	comisión
1	Cozy Apart...	New York	123 Elm St	2	Furnished, cl...	200000.0	2023-01-15	2023-02-20	20000002	30000001	30000009	10000.0
2	Modern Loft...	Los Angeles	456 Oak St	1	Open floor p...	300000.0	2023-02-20	2023-03-25	20000002	30000002	40000002	15000.0
3	Family Home...	Chicago	789 Pine St	3	Spacious ba...	400000.0	2023-03-25	2023-04-30	20000003	30000003	40000003	20000.0
4	Beachfront...	Miami	101 Maple St	4	Private beac...	1000000.0	2023-04-30	2023-05-30	20000004	30000004	40000004	50000.0
5	Mountain R...	Denver	202 Cedar St	2	Secluded ca...	250000.0	2023-05-30	2023-06-30	20000005	30000005	40000005	12500.0
6	City Pentho...	San Francisco	303 Birch St	3	Luxurious p...	1500000.0	2023-06-30	2023-07-30	20000006	30000006	40000006	75000.0
7	Historic To...	Boston	404 Walnut ...	3	Classic arch...	800000.0	2023-07-30	2023-08-30	20000007	30000007	40000007	40000.0
8	Ranch Retre...	Austin	505 Cherry ...	5	Large ranch...	600000.0	2023-08-30	2023-09-30	20000008	30000008	40000008	30000.0
9	Lakefront L...	Seattle	606 Pineap...	2	Quaint cotta...	350000.0	2023-09-30	2023-10-30	20000010	30000009	40000009	17500.0
10	Country Est...	Nashville	707 Orange ...	6	Sprawling e...	2000000.0	2023-10-30	2023-11-30	20000010	30000010	30000010	100000.0
11	Downtown ...	Toronto	808 Banana...	1	Convenient l...	500000.0	2023-11-30	2023-12-30	20000010	30000011	40000011	25000.0
12	Suburban Dr...	Vancouver	909 Grape St	4	Family-frien...	700000.0	2023-12-30	2024-01-30	20000012	30000012	40000012	35000.0
13	Luxury Pent...	Montreal	1010 Apple ...	3	Panoramic cl...	1800000.0	2024-01-30	2024-02-28	20000013	30000013	40000013	60000.0
14	Beachside B...	Miami	1111 Pear St	2	Steps from t...	600000.0	2024-02-28	2024-03-30	20000014	30000014	40000014	30000.0
15	Mountain Ca...	Denver	1212 Cherry ...	1	Cozy cabin...	400000.0	2024-03-30	2024-04-30	20000015	30000015	40000015	10000.0
16	Urban Oasis...	San Francisco	1313 Plum St	2	Modern desi...	900000.0	2024-04-30	2024-05-30	20000016	30000016	40000016	45000.0
17	Charming Co...	Boston	1414 Lemo...	2	Quaint cotta...	400000.0	2024-05-30	2024-06-30	20000017	30000017	40000017	20000.0
18	Lakefront R...	Seattle	1515 Lime St	3	Tranquil lak...	800000.0	2024-06-30	2024-07-30	20000018	30000018	40000018	40000.0
19	Ranch House...	Austin	1616 Mang... 4	4	Traditional r...	1000000.0	2024-07-30	2024-08-30	20000019	30000019	30000019	50000.0
20	City Townh...	New York	1717 Papay...	3	Sleek design...	700000.0	2024-08-30	2024-09-30	20000020	30000020	40000020	35000.0
21	casa	Miami	123 smith	4	pscina	100000.0	2022-11-05	2023-03-29	20000021	30000021	40000021	50000.0
22	maison	paris	parischanula	5	fea	1333.22	2024-03-07	2024-03-12	20000022	30000022	40000022	33333.0
23	D	S	E	33.0	2024-03-12	2024-03-11	20000023	30000023	333	1	6.0	
24	Mountain R...	Brooklyn	777, Nets A... 4	4	Furnished,p...	7000000.0	2024-03-15	2024-03-29	20000029	30000029	30000029	70000.0

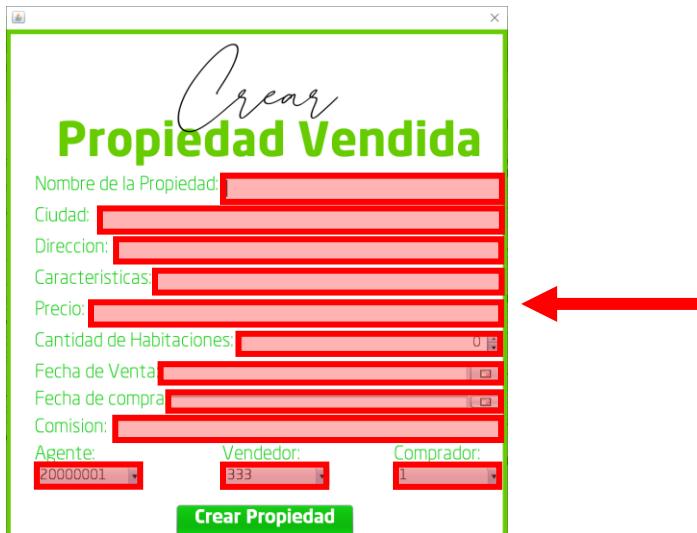
Del mismo modo, se puede añadir Propiedades presionando el botón "Añadir Propiedad Vendida". Al hacer clic en este botón, aparecerá un nuevo panel donde se deberá llenar la respectiva información de la Propiedad.



The screenshot shows a software interface with a green header bar containing navigation tabs: ADMINS, COMPRADORES, VENDEDORES, AGENTES, PROP. VENDIDAS (highlighted in yellow), and PROP. MERCADO. Below the header is a form titled 'AÑADIR PROPIEDAD VENDIDA'. The form has fields for: id, nombre, ciudad, dirección, dormitorios, características, precio, f_publicación, f_venta, id_agente, id_vendedor, id_comprador, and comisión. A red arrow points from the left towards the 'AÑADIR PROPIEDAD VENDIDA' button.

3.2.2.10 Botón de Añadir Propiedades Vendidas

Aquí se muestra la ventana que se mencionó en el punto 3.2.2.9 en el cual se utiliza para crear una nueva Propiedad y agregarlo en la tabla, donde se debe agregar los atributos necesitados



Al haber ingresado los valores correspondientes se presiona el botón de Crear Propiedad Vendida para ingresar los valores en la Tabla



3.2.2.11 Botón de Propiedades en Mercado

Al presionar el botón de propiedades en mercado el cual aparecerá la tabla de Propiedades las cuales han sido publicadas para su respectiva venta y sus atributos específicos.

ID	nombre	ciudad	direccion	dormitorios	características	precio	f. publicacion	id agente	id vendedor
1	Cozy Studio	Chicago	321 Main St	1	Compact layout, ..	150000.0	2023-01-15	20000021	30000001
2	Spacious Loft	Seattle	789 Oak St	2	High ceilings, nat..	400000.0	2023-02-20	20000022	30000002
3	Suburban Retreat	Los Angeles	456 Pine St	3	Quiet neighborho..	600000.0	2023-03-25	20000023	30000003
4	Beachfront Condo	Miami	101 Ocean Blvd	2	Ocean views, res..	800000.0	2023-04-30	20000024	30000004
5	Mountain Chalet	Denver	202 Summit Ave	4	Rustic charm, m..	500000.0	2023-05-30	20000025	30000005
6	Urban Loft	San Francisco	303 Market St	1	Industrial style, ..	900000.0	2023-06-30	20000026	30000006
7	Historic Mansion	Boston	404 Park Ave	5	Grand architectur..	2500000.0	2023-07-30	20000027	30000007
8	Country Farmhouse	Austin	505 Farm Rd	3	Farmhouse char..	3000000.0	2023-08-30	20000028	30000008
9	Lakefront Cabin	Seattle	606 Lakeview Dr	2	Cozy cabin, lakef..	3000000.0	2023-09-30	20000029	30000009
10	Luxury Villa	Miami	707 Luxury Lane	5	Private estate, lu..	3000000.0	2023-10-30	20000030	30000010
11	Downtown Apart...	Toronto	808 City Ave	1	Convenient locat..	400000.0	2023-11-30	20000031	30000011
12	Coastal Cottage	Vancouver	909 Beach Blvd	2	Charming cottage..	600000.0	2023-12-30	20000032	30000012
13	Skyline Penthouse	Montreal	1010 Skyline Dr	3	Panoramic city vi..	2000000.0	2024-01-30	20000033	30000013
14	Beachside Retreat	Miami	1111 Shoreline ..	2	Steps from the be..	750000.0	2024-02-28	20000034	30000014
15	Forest Cabin	Denver	1212 Forest Rd	1	Secluded cabin, s..	150000.0	2024-03-30	20000035	30000015
16	City View Condo	San Francisco	1313 Cityview A...	2	Panoramic city vi..	850000.0	2024-04-30	20000036	30000016
17	Countryside Cott...	Boston	1414 Countrysid...	2	Quaint cottage, r..	450000.0	2024-05-30	20000037	30000017
18	Lake House	Seattle	1515 Lakeside Dr	3	Tanquill lakefron..	950000.0	2024-06-30	20000028	30000038
19	Ranch	Austin Butler	prueba	4	Ranch-style livin..	4.0	2024-03-06	20000019	30000039
20	prueba	prueba	prueba	3	prueba	3.0	2024-03-18	20000001	333
21	d	d	d	1	d	1.0	2024-03-07	20000001	333

Del mismo modo, se puede añadir Propiedades presionando el botón "Añadir Propiedad Mercado". Al hacer clic en este botón, aparecerá un nuevo panel donde se deberá llenar la respectiva información de la Propiedad.

ID	nombre	ciudad	direccion	dormitorios	características	precio	f. publicacion	id agente	id vendedor
1	Cozy Studio	Chicago	321 Main St	1	Compact layout, ..	150000.0	2023-01-15	20000021	30000001
2	Spacious Loft	Seattle	789 Oak St	2	High ceilings, nat..	400000.0	2023-02-20	20000022	30000002
3	Suburban Retreat	Los Angeles	456 Pine St	3	Quiet neighborho..	600000.0	2023-03-25	20000023	30000003
4	Beachfront Condo	Miami	101 Ocean Blvd	2	Ocean views, res..	800000.0	2023-04-30	20000024	30000004
5	Mountain Chalet	Denver	202 Summit Ave	4	Rustic charm, m..	500000.0	2023-05-30	20000025	30000005
6	Urban Loft	San Francisco	303 Market St	1	Industrial style, ..	900000.0	2023-06-30	20000026	30000006
7	Historic Mansion	Boston	404 Park Ave	5	Grand architectur..	2500000.0	2023-07-30	20000027	30000007
8	Country Farmhouse	Austin	505 Farm Rd	3	Farmhouse char..	3000000.0	2023-08-30	20000028	30000008
9	Lakefront Cabin	Seattle	606 Lakeview Dr	2	Cozy cabin, lakef..	3000000.0	2023-09-30	20000029	30000009
10	Luxury Villa	Miami	707 Luxury Lane	5	Private estate, lu..	3000000.0	2023-10-30	20000030	30000010
11	Downtown Apart...	Toronto	808 City Ave	1	Convenient locat..	400000.0	2023-11-30	20000031	30000011
12	Coastal Cottage	Vancouver	909 Beach Blvd	2	Charming cottage..	600000.0	2023-12-30	20000032	30000012
13	Skyline Penthouse	Montreal	1010 Skyline Dr	3	Panoramic city vi..	2000000.0	2024-01-30	20000033	30000013
14	Beachside Retreat	Miami	1111 Shoreline ..	2	Steps from the be..	750000.0	2024-02-28	20000034	30000014
15	Forest Cabin	Denver	1212 Forest Rd	1	Secluded cabin, s..	150000.0	2024-03-30	20000035	30000015
16	City View Condo	San Francisco	1313 Cityview A...	2	Panoramic city vi..	850000.0	2024-04-30	20000036	30000016
17	Countryside Cott...	Boston	1414 Countrysid...	2	Quaint cottage, r..	450000.0	2024-05-30	20000037	30000017
18	Lake House	Seattle	1515 Lakeside Dr	3	Tanquill lakefron..	950000.0	2024-06-30	20000028	30000038
19	Ranch	Austin Butler	prueba	4	Ranch-style livin..	4.0	2024-03-06	20000019	30000039
20	prueba	prueba	prueba	3	prueba	3.0	2024-03-18	20000001	333
21	d	d	d	1	d	1.0	2024-03-07	20000001	333

3.2.2.12 Botón de Añadir Propiedades en Mercado

Aquí se muestra la ventana que se mencionó en el punto 3.2.2. en el cual se utiliza para crear una nueva Propiedad para su publicación y agregarlo en la tabla, donde se debe agregar los atributos necesitados



Al haber ingresado los valores correspondientes se presiona el botón de Crear Propiedad en Mercado para ingresar los valores en la Tabla



3.2.3 Ventana de Bitácora

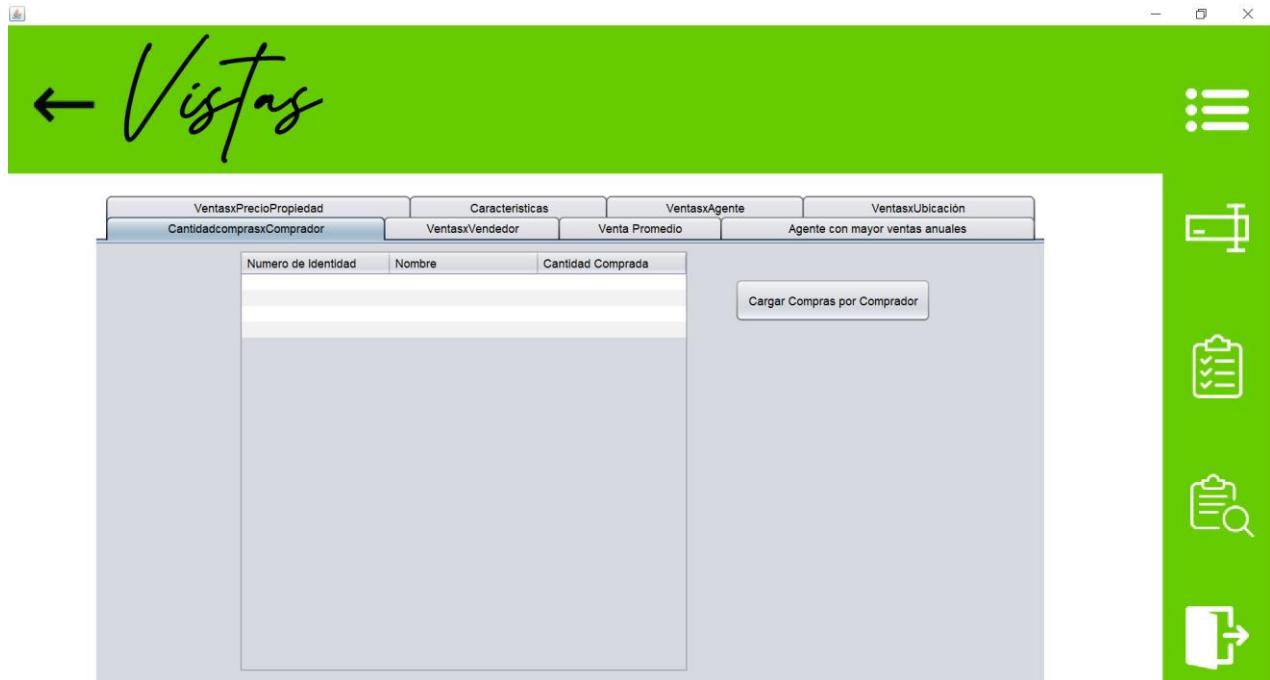
En esta sección, se presentan las Tablas Resumen que detallan las acciones llevadas a cabo por los administradores, agentes, vendedores, compradores, así como también las propiedades vendidas y aquellas en el mercado. Estas tablas ofrecen una visión completa y actualizada de todas las interacciones y transacciones realizadas en tiempo real, brindando una comprensión detallada del estado y las actividades en curso en la plataforma. Desde la gestión de agentes y propiedades hasta el seguimiento de ventas y compras, estas tablas proporcionan una perspectiva completa de las operaciones que se llevan a cabo, facilitando una toma de decisiones informada y estratégica para todos los usuarios involucrados.

idOperacion	username	operacion	tabla	id	fecha	hora
1	desb	CREAR	propiedades_vendidas	22	2024-03-13	09:24:01
2	desb	MODIFICAR	propiedades_en_merc...	22	2024-03-13	10:19:32
3	desb	MODIFICAR	compradores	1	2024-03-13	19:21:44
4	desb	MODIFICAR	compradores	1	2024-03-13	19:23:36
5	desb	MODIFICAR	compradores	1	2024-03-13	19:25:02
6	desb	MODIFICAR	compradores	1	2024-03-13	19:26:40
7	desb	MODIFICAR	compradores	30000003	2024-03-13	19:27:42
8	desb	CREAR	propiedades_en_merc...	20	2024-03-13	21:48:23
9	desb	CREAR	propiedades_en_merc...	21	2024-03-13	21:49:49
10	desb	MODIFICAR	propiedades_en_merc...	19	2024-03-13	22:10:35
11	agzf	CREAR	propiedades_vendidas	23	2024-03-15	18:10:36
12	vim	CREAR	propiedades_vendidas	24	2024-03-15	23:30:02

Además de las Tablas Resumen que muestran las acciones realizadas por los diferentes usuarios y el estado de las propiedades, nuestra plataforma también ofrece herramientas adicionales para optimizar la gestión y análisis de datos. Estas incluyen funciones avanzadas de filtrado y búsqueda que permiten a los usuarios explorar los datos de manera más específica, así como la generación de informes personalizados que proporcionan una visión más detallada y personalizada de la actividad en la plataforma. Con estas herramientas adicionales, los usuarios pueden obtener información aún más precisa y relevante para respaldar sus decisiones y estrategias en el mercado inmobiliario.

3.2.4 Ventana de Reportes

La ventana de "Resumen de Ventas" ofrece una vista completa y detallada de todas las vistas disponibles relacionadas con las ventas en la plataforma. En esta ventana, los usuarios pueden acceder fácilmente a cada una de las vistas mencionadas anteriormente, incluyendo "Venta por Precio de Propiedad", "Características de las Propiedades Vendidas", "Año de Valor Total", "Ventas Promedio", "Ventas por Agente", "Ventas por Vendedor" y "Cantidad de Propiedades Compradas por Comprador".



Cada botón en la ventana representa una vista específica y al hacer clic en él, se abre la vista correspondiente, permitiendo a los usuarios explorar y analizar diferentes aspectos de las ventas de manera rápida y conveniente. Esta ventana de resumen proporciona una herramienta centralizada para acceder a todas las perspectivas relevantes sobre las ventas en la plataforma, lo que facilita la toma de decisiones informadas y estratégicas por parte de los usuarios.

3.2.4.1 Botón de Ventas por Ubicación

La vista "Venta por Ubicación" ofrece una representación visual de las ventas según la ubicación geográfica de las propiedades. Presenta datos de manera clara y concisa, permitiendo a los usuarios identificar patrones y tendencias en diferentes áreas. Esta vista facilita la comprensión y el análisis de las ventas por ubicación, lo que puede ser útil para la toma de decisiones estratégicas en el mercado inmobiliario.

Ciudad	Cantidad Vendida
Boston	2
caliro	1
Denver	2
Los Angeles	1
Miami	3
Montreal	1
Nashville	1
New York	2
p	1
paris	1
San Francisco	2
Seattle	2
Toronto	1
Vancouver	1

3.2.4.2 Botón de Ventas por Precio de Propiedad

La vista "Venta por Precio de Propiedad" proporciona una visualización de las ventas según el rango de precios de las propiedades. Permite a los usuarios identificar cómo se distribuyen las ventas en diferentes rangos de precios, lo que puede ser útil para comprender mejor la demanda del mercado y ajustar estrategias de precios en consecuencia.

Precio	Cantidad
333.00	1
1333.22	1
3333.00	1
100000.00	1
200000.00	1
250000.00	1
300000.00	1
350000.00	1
400000.00	1
500000.00	1
600000.00	1
700000.00	2
800000.00	2
900000.00	1
1000000.00	2
1500000.00	1
1800000.00	1
2000000.00	1

3.2.4.3 Botón de Características

Permite ver las características específicas de las propiedades vendidas en función de su precio, como el número de dormitorios, la presencia de piscina u otras características relevantes.

3.2.4.4 Botón de Año de Valor Total

Muestra cómo ha variado el valor total de las ventas a lo largo de los años, lo que proporciona información sobre las tendencias de precios a lo largo del tiempo.

holderId	nombre	valor total	año
20000010	James Thomas	2950000.0	2023

3.2.4.5 Botón de Ventas Promedio

Calcula y muestra el precio promedio de las propiedades vendidas en función de su precio, lo que puede ser útil para evaluar el valor medio del mercado en diferentes rangos de precios.

3.2.4.6 Botón de Ventas por Agente

Permite visualizar las ventas realizadas por cada agente en función del precio de las propiedades, lo que facilita la evaluación del desempeño individual de los agentes en relación con las ventas.

3.2.4.7 Botón de Ventas por Vendedor

Muestra las ventas realizadas por cada vendedor en función del precio de las propiedades, lo que proporciona información sobre la contribución de cada vendedor al volumen total de ventas.

The screenshot shows a software window titled 'Vistas' with a green header bar. On the right side, there is a vertical toolbar with several icons: a list, a magnifying glass, a clipboard, and a file icon. The main content area has a table with the following columns: 'VentasxPrecioPropiedad', 'Características', 'VentasxAgente', and 'VentasxUbicación'. Below these columns is a sub-table with the following columns: 'CantidadcomprasxComprador', 'VentasxVendedor', 'Venta Promedio', and 'Agente con mayor ventas anuales'. A button labeled 'Cargar Ventas por Vendedor' is located to the right of the sub-table. The data in the sub-table is as follows:

Número de Identidad	Nombre	Cantidad Vendida
333	d	3
30000002	Maria Hernandez	2
30000004	Ana Lopez	1
30000005	Pedro Perez	1
30000006	Laura Rodriguez	1
30000007	Diego Gonzalez	2
30000009	Alejandro Sanchez	1
30000010	Valeria Flores	1
30000011	Gabriel Torres	1
30000012	Luis Morales	1
30000013	Marina Vargas	1
30000014	Pablo Cruz	1
30000015	Elena Gomez	1
30000016	Carmen Reyes	1
30000017	Ricardo Diaz	1
30000018	Daniela Castro	1
30000020	Javier Mendoza	1

3.2.4.8 Botón de Cantidad de Propiedades Compradas por Comprador

Permite ver cuántas propiedades ha comprado cada comprador en función del precio, lo que puede ayudar a identificar clientes potenciales para futuras transacciones.

The screenshot shows a software window titled 'Vistas' with a green header bar. On the right side, there is a vertical toolbar with several icons: a list, a magnifying glass, a clipboard, and a file icon. The main content area has a table with the following columns: 'VentasxPrecioPropiedad', 'Características', 'VentasxAgente', and 'VentasxUbicación'. Below these columns is a sub-table with the following columns: 'CantidadcomprasxComprador', 'VentasxVendedor', 'Venta Promedio', and 'Agente con mayor ventas anuales'. A button labeled 'Cargar Compras por Comprador' is located to the right of the sub-table. The data in the sub-table is as follows:

Número de Identidad	Nombre	Cantidad Comprada
1	prueba5	3
30000002	Maria Hernandez	1
30000010	Valeria Flores	1
40000001	Diego Gonzalez	1
40000004	Sofia Diaz	1
40000005	Alejandro Sanchez	1
40000006	Valeria Flores	1
40000007	Gabriel Torres	1
40000008	Luis Morales	1
40000009	Marina Vargas	1
40000011	Elena Gomez	1
40000013	Carmen Reyes	1
40000014	Ricardo Diaz	1
40000015	Daniela Castro	1
40000016	Fernando Ortiz	1
40000017	Javier Mendoza	1
40000018	Marta Aguilera	1
40000020	Lucas Fernandez	1
	Raul Silva	1

3.3 Ventanas de Vendedor

La ventana de "Vendedores" en nuestra plataforma ofrece un panorama completo de las actividades de los vendedores, dividiendo la información en dos secciones principales: "Asignadas" y "Vendidas".

En la sección "Asignadas", se muestran todas las propiedades que han sido asignadas a cada vendedor para su venta. Aquí, los usuarios pueden acceder fácilmente a detalles específicos de cada propiedad, como la dirección, el precio de venta y el estado actual de la transacción. Esta sección permite a los vendedores dar seguimiento a las propiedades que les han sido asignadas de manera efectiva, lo que facilita la gestión de su cartera y el seguimiento del progreso de las ventas.

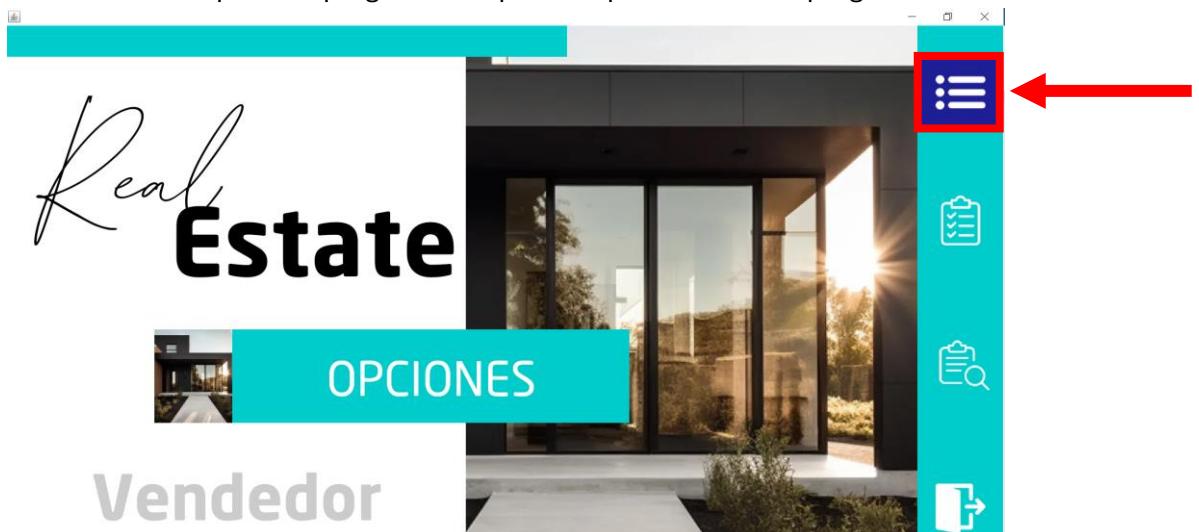
3.3.1 Ventana Principal (Vendedor_MainScreen)

Aquí se muestra la pantalla inicial de esta función englobando todas las opciones necesarias



3.3.1.1 Botón de Opciones

Al presionar el **BOTÓN DE OPCIONES** se muestran las principales funcionalidades divididas en los diferentes paneles programados para la optimización del programa



3.3.1.2 Botón de Propiedades Asignadas al Vendedor

La función "Propiedades Asignadas por el Vendedor" ofrece una visión detallada de todas las propiedades actualmente asignadas a un vendedor específico para la venta. Aquí, se muestra información específica sobre cada propiedad, como dirección y estado de la transacción, permitiendo a los vendedores gestionar su cartera de manera eficiente.



3.3.1.3 Botón de Propiedades Vendidas por el Vendedor

La función "Propiedades Vendidas por el Vendedor" proporciona un registro completo de todas las propiedades que un vendedor ha vendido en la plataforma. Los usuarios pueden revisar detalles como el precio de venta y la fecha de cierre, lo que permite evaluar el rendimiento del vendedor de manera rápida y efectiva. En conjunto, estas funciones ofrecen una herramienta integral para gestionar y evaluar las actividades de venta de los vendedores en la plataforma.



3.3.1.4 Botón de Salida

El botón "Salir" ofrece a los usuarios la posibilidad de cerrar sesión y regresar a la pantalla de inicio de sesión. Al hacer clic en este botón, se finaliza la sesión actual del usuario y se le redirige de manera segura a la pantalla de inicio de sesión, donde pueden ingresar nuevamente sus credenciales para acceder a la aplicación. Esta funcionalidad es esencial para garantizar la seguridad de la cuenta y proteger la privacidad de los usuarios al finalizar su sesión cuando ya no la necesitan. Además, proporciona una manera conveniente y rápida de cambiar de usuario o cerrar la sesión de forma segura cuando se termina de utilizar la aplicación.



3.3.2 Ventana de Propiedades Asignadas al Vendedor

La función "Propiedades Asignadas por el Vendedor" tiene una ventana dedicada que muestra una vista específica definida para esta función. En esta ventana, los usuarios pueden acceder a información detallada sobre cada propiedad asignada a un vendedor específico, incluyendo detalles como dirección, características y estado de la transacción.

3.3.3 Ventana de Propiedades Vendidas por el Vendedor

La función "Propiedades Vendidas por el Vendedor" también cuenta con su propia ventana, que muestra una vista específica diseñada para esta función. Aquí, los usuarios pueden acceder a un registro completo de todas las propiedades que un vendedor ha vendido en la plataforma. Se proporciona información detallada, como el precio de venta, la fecha de cierre y la información del comprador, lo que permite evaluar el rendimiento del vendedor de manera precisa.

3.4 Ventanas de Comprador

La ventana del "Comprador" ofrece una vista completa de las opciones disponibles para los compradores en la plataforma, incluyendo la opción de explorar "Propiedades en el Mercado". En esta ventana, los compradores pueden acceder a una lista detallada de todas las propiedades actualmente disponibles para la venta en la plataforma.

3.4.1 Ventana Principal (Comprador_MainScreen)



3.4.1.1 Botón de Opciones

Al presionar el **BOTÓN DE OPCIONES** se muestran las principales funcionalidades divididas en los diferentes paneles programados para la optimización del programa



3.4.1.2 Botón de Propiedad en Mercado

La sección "Propiedades en el Mercado" muestra una lista de todas las propiedades que están disponibles para la compra. Los compradores pueden explorar esta lista y filtrar las propiedades según sus preferencias específicas, como ubicación, precio y características. Cada propiedad en la lista está acompañada de información detallada, como la dirección, características principales y precio de venta.



3.4.1.3 Botón de Salida

El botón "Salir" ofrece a los usuarios la posibilidad de cerrar sesión y regresar a la pantalla de inicio de sesión. Al hacer clic en este botón, se finaliza la sesión actual del usuario y se le redirige de manera segura a la pantalla de inicio de sesión, donde pueden ingresar nuevamente sus credenciales para acceder a la aplicación. Esta funcionalidad es esencial para garantizar la seguridad de la cuenta y proteger la privacidad de los usuarios al finalizar su sesión cuando ya no la necesitan. Además, proporciona una manera conveniente y rápida de cambiar de usuario o cerrar la sesión de forma segura cuando se termina de utilizar la aplicación.



4. Triggers

4.1 Registrar Agente

Este trigger se activa antes de eliminar un agente de la tabla AGENTES. Primero, recupera el nombre de usuario activo de la tabla USUARIOS. Luego, obtiene el número de identidad del agente que se va a eliminar y elimina todas las propiedades asociadas a ese agente tanto en la tabla PROPIEDADES_EN_MERCADO como en PROPIEDADES_VENDIDAS. Después, cuenta el número de operaciones registradas en la tabla BITACORA para determinar el último ID de operación. Finalmente, registra la operación de eliminación del agente en la tabla BITACORA, incluyendo el ID de operación, nombre de usuario, tipo de operación, tabla afectada, número de identidad del agente eliminado y la fecha y hora actual.

Name:	register_b_agente	Definer:	admin@%
On table:	agentes		
Event:	BEFORE	DELETE	
Trigger statement: (e.g. "SET NEW.columnA = TRIM(OLD.columnA)"')			
<pre> 1 BEGIN 2 DECLARE noID INT; 3 DECLARE lastID INT; 4 DECLARE usuario VARCHAR(20); 5 6 SELECT username INTO usuario 7 FROM usuarios 8 WHERE activo = 1; 9 10 SELECT noIdentidad INTO noID 11 FROM agentes 12 WHERE noIdentidad = OLD.noIdentidad; 13 14 DELETE FROM propiedades_en_mercado 15 WHERE noIdentidad_Agente = noID; 16 17 DELETE FROM propiedades_vendidas 18 WHERE noIdentidad_Agente = noID; 19 20 SELECT COUNT(idOperacion) INTO lastID 21 FROM bitacora; 22 23 INSERT INTO bitacora VALUES (lastID+1, usuario, 'BORRAR', 'agentes', noID, CURRENT_DATE, NOW()); 24 END </pre>			

4.2 Registrar Comprador

Este trigger se activa antes de eliminar un registro de la tabla de compradores. Durante su ejecución, recupera el nombre de usuario activo, el número de identidad del comprador que se eliminará y borra todas las propiedades vendidas asociadas a ese comprador. Luego, registra esta operación de eliminación en la bitácora junto con detalles como el ID de operación, nombre de usuario, tipo de operación, tabla afectada, número de identidad del comprador eliminado y la fecha y hora actual.

Name:	register_b_comprador	Definer:	admin@%
On table:	compradores		
Event:	BEFORE	DELETE	
Trigger statement: (e.g. "SET NEW.columnA = TRIM(OLD.columnA)"')			
<pre> 1 BEGIN 2 DECLARE noID INT; 3 DECLARE lastID INT; 4 DECLARE usuario VARCHAR(20); 5 6 SELECT username INTO usuario 7 FROM usuarios 8 WHERE activo = 1; 9 10 SELECT noIdentidad INTO noID 11 FROM compradores 12 WHERE noIdentidad = OLD.noIdentidad; 13 14 15 DELETE FROM propiedades_vendidas 16 WHERE noIdentidad_Comprador = noID; 17 18 SELECT COUNT(idOperacion) INTO lastID 19 FROM bitacora; 20 21 INSERT INTO bitacora VALUES (lastID+1, usuario, 'BORRAR', 'compradores', noID, CURRENT_DATE, NOW()); 22 END </pre>			

4.3 Registrar Propiedad en Mercado

Este trigger se activa antes de eliminar un registro de la tabla de propiedades en mercado. Durante su ejecución, obtiene el nombre de usuario activo y el ID de la propiedad que se eliminará. Luego, registra esta operación de eliminación en la bitácora, incluyendo detalles como el ID de operación, nombre de usuario, tipo de operación, tabla afectada, ID de la propiedad eliminada y la fecha y hora actual.

Name:	register_b_propiedadm	Definer:	admin@%
On table:	propiedades_en_mercado		
Event:	BEFORE	DELETE	

Trigger statement: (e.g. "SET NEW.columnA = TRIM(OLD.columnA)")

```

1 BEGIN
2   DECLARE noID INT;
3   DECLARE lastID INT;
4   DECLARE usuario VARCHAR(20);
5
6   SELECT username INTO usuario
7   FROM usuarios
8   WHERE activo = 1;
9
10  SELECT idPropiedad INTO noID
11   FROM propiedades_en_mercado
12   WHERE idPropiedad = OLD.idPropiedad;
13
14  SELECT COUNT(idOperacion) INTO lastID
15   FROM bitacora;
16
17  INSERT INTO bitacora VALUES (lastID+1, usuario, 'BORRAR', 'propiedades_en_mercado', noID, CURRENT_DATE, NOW());
18 END

```

4.4 Registrar Propiedad Vendida

Este trigger se activa antes de eliminar un registro de la tabla de propiedades vendidas. Durante su ejecución, obtiene el nombre de usuario activo y el ID de la propiedad que se eliminará. Luego, registra esta operación de eliminación en la bitácora, incluyendo detalles como el ID de operación, nombre de usuario, tipo de operación, tabla afectada, ID de la propiedad vendida eliminada y la fecha y hora actual.

Name:	register_b_propiedadv	Definer:	admin@%
On table:	propiedades_vendidas		
Event:	BEFORE	DELETE	

Trigger statement: (e.g. "SET NEW.columnA = TRIM(OLD.columnA)")

```

1 BEGIN
2   DECLARE noID INT;
3   DECLARE lastID INT;
4   DECLARE usuario VARCHAR(20);
5
6   SELECT username INTO usuario
7   FROM usuarios
8   WHERE activo = 1;
9
10  SELECT idPropiedad INTO noID
11   FROM propiedades_vendidas
12   WHERE idPropiedad = OLD.idPropiedad;
13
14  SELECT COUNT(idOperacion) INTO lastID
15   FROM bitacora;
16
17  INSERT INTO bitacora VALUES (lastID+1, usuario, 'BORRAR', 'propiedades_vendidas', noID, CURRENT_DATE, NOW());
18 END

```

4.5 Registrar Usuario

Este trigger se activa antes de eliminar un registro de la tabla de usuarios. Durante su ejecución, primero obtiene el número de identidad del usuario que se eliminará. Luego, recupera el nombre de usuario activo. Después, procede a eliminar al usuario de las tablas de compradores, vendedores y agentes utilizando el número de identidad obtenido. Finalmente, cuenta el número de operaciones registradas en la tabla de bitácora para determinar el último ID de operación y registra la operación de eliminación del usuario en la bitácora, incluyendo detalles como el ID de operación, nombre de usuario, tipo de operación, tabla afectada, número de identidad del usuario eliminado y la fecha y hora actual.

Name:	register_b_usuario	Definer:	admin@%
On table:	usuarios		
Event:	BEFORE	DELETE	

Trigger statement: (e.g. "SET NEW.columnA = TRIM(OLD.columnA)")

```

1 BEGIN
2   DECLARE noID INT;
3   DECLARE lastID INT;
4   DECLARE usuario VARCHAR(20);
5
6   SELECT noIdentidad INTO noID
7   FROM usuarios
8   WHERE noIdentidad = OLD.noIdentidad;
9
10
11  SELECT username INTO usuario
12  FROM usuarios
13  WHERE activo = 1;
14
15  DELETE FROM compradores
16  WHERE noIdentidad = noID;
17
18  DELETE FROM vendedores
19  WHERE noIdentidad = noID;
20
21  DELETE FROM agentes
22  WHERE noIdentidad = noID;
23
24  SELECT COUNT(idOperacion) INTO lastID
25  FROM bitacora;
26
27  INSERT INTO bitacora VALUES (lastID+1, usuario, 'BORRAR', 'usuarios', noID, CURRENT_DATE, NOW());
28 END

```

4.6 Registrar Vendedor

Este trigger se activa antes de eliminar un registro de la tabla de vendedores. Durante su ejecución, primero obtiene el nombre de usuario activo. Luego, recupera el número de identidad del vendedor que se eliminará. Despues, borra todas las propiedades en mercado y vendidas asociadas a ese vendedor. Posteriormente, cuenta el número de operaciones registradas en la tabla de bitácora para determinar el último ID de operación y registra la operación de eliminación del vendedor en la bitácora, incluyendo detalles como el ID de operación, nombre de usuario, tipo de operación, tabla afectada, número de identidad del vendedor eliminado y la fecha y hora actual.

Name:	register_b_vendedor	Definer:	admin@%
On table:	vendedores		
Event:	BEFORE	DELETE	

Trigger statement: (e.g. "SET NEW.columnA = TRIM(OLD.columnA)")

```

1 BEGIN
2   DECLARE noID INT;
3   DECLARE lastID INT;
4   DECLARE usuario VARCHAR(20);
5
6   SELECT username INTO usuario
7   FROM usuarios
8   WHERE activo = 1;
9
10
11  SELECT noIdentidad INTO noID
12  FROM vendedores
13  WHERE noIdentidad = OLD.noIdentidad;
14
15
16  DELETE FROM propiedades_en_mercado
17  WHERE noIdentidad_Vendedor = noID;
18
19  DELETE FROM propiedades_vendidas
20  WHERE noIdentidad_Vendedor = noID;
21
22
23  SELECT COUNT(idOperacion) INTO lastID
24  FROM bitacora;
25
26  INSERT INTO bitacora VALUES (lastID+1, usuario, 'BORRAR', 'vendedores', noID, CURRENT_DATE, NOW());
27 END

```

4.7 Registrar Agente

Este trigger se activa después de insertar un nuevo registro en la tabla de agentes. Durante su ejecución, primero obtiene el nombre de usuario activo. Luego, cuenta el número de operaciones registradas en la tabla de bitácora para determinar el último ID de operación. Después, recupera el número de identidad del agente recién creado. Finalmente, registra la operación de creación del agente en la bitácora, incluyendo detalles como el ID de operación, nombre de usuario, tipo de operación, tabla afectada, número de identidad del agente creado y la fecha y hora actual.

Name:	register_c_agente	Definer:	admin@%
On table:	agentes		
Event:	AFTER	INSERT	

Trigger statement: (e.g. "SET NEW.columnA = TRIM(OLD.columnA)"

```

1 BEGIN
2   DECLARE lastID INT;
3   DECLARE usuario VARCHAR(20);
4   DECLARE noID INT;
5
6   SELECT username INTO usuario
7   FROM usuarios
8   WHERE activo = 1;
9
10  SELECT COUNT(idOperacion) INTO lastID
11    FROM bitacora;
12
13  SELECT noIdentidad INTO noID
14    FROM agentes
15   WHERE noIdentidad = NEW.noIdentidad;
16
17  INSERT INTO bitacora VALUES(lastID+1, usuario, 'CREAR', 'agentes', noID, CURRENT_DATE, NOW());
18 END

```

4.8 Registrar Comprador

Este trigger se activa después de insertar un nuevo registro en la tabla de compradores. Durante su ejecución, primero obtiene el nombre de usuario activo. Luego, cuenta el número de operaciones registradas en la tabla de bitácora para determinar el último ID de operación. Después, recupera el número de identidad del comprador recién creado. Finalmente, registra la operación de creación del comprador en la bitácora, incluyendo detalles como el ID de operación, nombre de usuario, tipo de operación, tabla afectada, número de identidad del comprador creado y la fecha y hora actual.

Name:	register_c_comprador	Definer:	admin@%
On table:	compradores		
Event:	AFTER	INSERT	

Trigger statement: (e.g. "SET NEW.columnA = TRIM(OLD.columnA)"

```

1 BEGIN
2   DECLARE noID INT;
3   DECLARE usuario VARCHAR(20);
4   DECLARE lastID INT;
5
6   SELECT username INTO usuario
7   FROM usuarios
8   WHERE activo = 1;
9
10  SELECT COUNT(idOperacion) INTO lastID
11    FROM bitacora;
12
13  SELECT noIdentidad INTO noID
14    FROM compradores
15   WHERE noIdentidad = NEW.noIdentidad;
16
17  INSERT INTO bitacora VALUES (lastID+1, usuario, 'CREAR', 'compradores', noID, CURRENT_DATE, NOW());
18 END

```

4.9 Registrar Propiedades en Mercado

Este trigger se activa después de insertar un nuevo registro en la tabla de propiedades en mercado. Durante su ejecución, primero obtiene el nombre de usuario activo. Luego, cuenta el número de operaciones registradas en la tabla de bitácora para determinar el último ID de operación. Despues, recupera el ID de la propiedad recién creada. Finalmente, registra la operación de creación de la propiedad en mercado en la bitácora, incluyendo detalles como el ID de operación, nombre de usuario, tipo de operación, tabla afectada, ID de la propiedad creada y la fecha y hora actual.

Name:	register_c_propiedadm	Definer:	admin@%
On table:	propiedades_en_mercado		
Event:	AFTER	INSERT	

Trigger statement: (e.g. "SET NEW.columnA = TRIM(OLD.columnA)"

```

1 BEGIN
2   DECLARE noID INT;
3   DECLARE usuario VARCHAR(20);
4   DECLARE lastID INT;
5
6   SELECT username INTO usuario
7   FROM usuarios
8   WHERE activo = 1;
9
10  SELECT COUNT(idOperacion) INTO lastID
11  FROM bitacora;
12
13  SELECT idPropiedad INTO noID
14  FROM propiedades_en_mercado
15  WHERE idPropiedad = NEW.idPropiedad;
16
17  INSERT INTO bitacora VALUES (lastID+1, usuario, 'CREAR', 'propiedades_en_mercado', noID, CURRENT_DATE, NOW());
18 END

```

4.10 Registrar Propiedades Vendidas

Este trigger se activa después de insertar un nuevo registro en la tabla de propiedades vendidas. Durante su ejecución, primero obtiene el nombre de usuario activo. Luego, cuenta el número de operaciones registradas en la tabla de bitácora para determinar el último ID de operación. Despues, recupera el ID de la propiedad vendida recién creada. Finalmente, registra la operación de creación de la propiedad vendida en la bitácora, incluyendo detalles como el ID de operación, nombre de usuario, tipo de operación, tabla afectada, ID de la propiedad vendida creada y la fecha y hora actual.

Name:	register_c_propiedadv	Definer:	admin@%
On table:	propiedades_vendidas		
Event:	AFTER	INSERT	

Trigger statement: (e.g. "SET NEW.columnA = TRIM(OLD.columnA)"

```

1 BEGIN
2   DECLARE noID INT;
3   DECLARE usuario VARCHAR(20);
4   DECLARE lastID INT;
5
6   SELECT username INTO usuario
7   FROM usuarios
8   WHERE activo = 1;
9
10  SELECT COUNT(idOperacion) INTO lastID
11  FROM bitacora;
12
13  SELECT idPropiedad INTO noID
14  FROM propiedades_vendidas
15  WHERE idPropiedad = NEW.idPropiedad;
16
17  INSERT INTO bitacora VALUES (lastID+1, usuario, 'CREAR', 'propiedades_vendidas', noID, CURRENT_DATE, NOW());
18 END

```

4.11 Registrar usuarios

Este trigger se activa después de insertar un nuevo registro en la tabla de usuarios. Durante su ejecución, primero cuenta el número de operaciones registradas en la tabla de bitácora para determinar el último ID de operación. Luego, obtiene el nombre de usuario activo. Después, recupera el número de identidad del usuario recién creado. Finalmente, registra la operación de creación del usuario en la bitácora, incluyendo detalles como el ID de operación, nombre de usuario, tipo de operación, tabla afectada, número de identidad del usuario creado y la fecha y hora actual.

Name:	register_c_users	Definer:	admin@%
On table:	usuarios		
Event:	AFTER	INSERT	

Trigger statement: (e.g. "SET NEW.columnA = TRIM(OLD.columnA)"

```

1 BEGIN
2   DECLARE lastID INT;
3   DECLARE usuario VARCHAR(20);
4   declare noID INT;
5
6   select COUNT(idOperacion)
7     into lastID
8   FROM bitacora;
9
10  SELECT username
11    into usuario
12  FROM usuarios
13 WHERE activo = 1;
14
15  SELECT noIdentidad into noID
16  FROM usuarios
17 WHERE noIdentidad = NEW.noIdentidad;
18
19  INSERT INTO bitacora VALUES (lastID+1, usuario, 'CREAR', 'usuarios', noID, CURRENT_DATE, NOW());
20 END

```

4.12 Registrar Vendedores

Este trigger se activa después de insertar un nuevo registro en la tabla de vendedores. Durante su ejecución, primero obtiene el nombre de usuario activo. Luego, cuenta el número de operaciones registradas en la tabla de bitácora para determinar el último ID de operación. Después, recupera el número de identidad del vendedor recién creado. Finalmente, registra la operación de creación del vendedor en la bitácora, incluyendo detalles como el ID de operación, nombre de usuario, tipo de operación, tabla afectada, número de identidad del vendedor creado y la fecha y hora actual.

Name:	register_c_vendedor	Definer:	admin@%
On table:	vendedores		
Event:	AFTER	INSERT	

Trigger statement: (e.g. "SET NEW.columnA = TRIM(OLD.columnA)"

```

1 BEGIN
2   DECLARE noID INT;
3   DECLARE usuario VARCHAR(20);
4   DECLARE lastID INT;
5
6   SELECT username INTO usuario
7   FROM usuarios
8 WHERE activo = 1;
9
10  SELECT COUNT(idOperacion) INTO lastID
11  FROM bitacora;
12
13  SELECT noIdentidad INTO noID
14  FROM vendedores
15 WHERE noIdentidad = NEW.noIdentidad;
16
17  INSERT INTO bitacora VALUES (lastID+1, usuario, 'CREAR', 'vendedores', noID, CURRENT_DATE, NOW());
18 END

```

4.13 Registrar Agente

Este trigger se activa después de modificar un registro en la tabla de agentes. Durante su ejecución, primero obtiene el nombre de usuario activo. Luego, cuenta el número de operaciones registradas en la tabla de bitácora para determinar el último ID de operación. Después, recupera el número de identidad del agente que ha sido modificado. Finalmente, registra la operación de modificación del agente en la bitácora, incluyendo detalles como el ID de operación, nombre de usuario, tipo de operación, tabla afectada, número de identidad del agente modificado y la fecha y hora actual.

Name:	register_m_agente	Definer:	admin@%
On table:	agentes		
Event:	AFTER	UPDATE	

```
trigger statement: (e.g. "SET NEW.columnA = TRIM(OLD.columnA)")

1 BEGIN
2   DECLARE noID INT;
3   DECLARE lastID INT;
4   DECLARE usuario VARCHAR(20);
5
6   SELECT username INTO usuario
7   FROM usuarios
8   WHERE activo = 1;
9
10  SELECT COUNT(idOperacion) INTO lastID
11  FROM bitacora;
12
13  SELECT noIdentidad INTO noID
14  FROM agentes
15  WHERE noIdentidad = NEW.noIdentidad;
16
17
18  INSERT INTO bitacora VALUES (lastID+1, usuario, 'MODIFICAR', 'agentes', noID, CURRENT_DATE, NOW());
19 END
```

4.14 Registrar Compradores

Este trigger se activa después de modificar un registro en la tabla de compradores. Durante su ejecución, primero obtiene el nombre de usuario activo. Luego, cuenta el número de operaciones registradas en la tabla de bitácora para determinar el último ID de operación. Después, recupera el número de identidad del comprador que ha sido modificado. Finalmente, registra la operación de modificación del comprador en la bitácora, incluyendo detalles como el ID de operación, nombre de usuario, tipo de operación, tabla afectada, número de identidad del comprador modificado y la fecha y hora actual.

Name:	register_m_comprador	Definer:	admin@%
On table:	compradores		
Event:	AFTER	UPDATE	

```
trigger statement: (e.g. "SET NEW.columnA = TRIM(OLD.columnA)")

1 BEGIN
2   DECLARE noID INT;
3   DECLARE lastID INT;
4   DECLARE usuario VARCHAR(20);
5
6   SELECT username INTO usuario
7   FROM usuarios
8   WHERE activo = 1;
9
10  SELECT COUNT(idOperacion) INTO lastID
11  FROM bitacora;
12
13  SELECT noIdentidad INTO noID
14  FROM compradores
15  WHERE noIdentidad = NEW.noIdentidad;
16
17
18  INSERT INTO bitacora VALUES (lastID+1, usuario, 'MODIFICAR', 'compradores', noID, CURRENT_DATE, NOW());
19 END
```

4.15 Registrar Propiedad en Mercado

Este trigger se activa después de modificar un registro en la tabla de propiedades en mercado. Durante su ejecución, primero obtiene el ID de la propiedad que ha sido modificada. Luego, cuenta el número de operaciones registradas en la tabla de bitácora para determinar el último ID de operación. Después, obtiene el nombre de usuario activo. Finalmente, registra la operación de modificación de la propiedad en mercado en la bitácora, incluyendo detalles como el ID de operación, nombre de usuario, tipo de operación, tabla afectada, ID de la propiedad modificada y la fecha y hora actual.

Name:	register_m_propiedaddm	Definer:	admin@%
On table:	propiedades_en_mercado		
Event:	AFTER	UPDATE	

Trigger statement: (e.g. "SET NEW.columnA = TRIM(OLD.columnA)")

```

1 BEGIN
2   DECLARE lastID INT;
3   DECLARE noID INT;
4   DECLARE usuario VARCHAR(20);
5
6   SELECT idPropiedad INTO noID
7   FROM propiedades_en_mercado
8   WHERE idPropiedad = NEW.idPropiedad;
9
10  SELECT COUNT(idOperacion) INTO lastID
11  FROM bitacora;
12
13  SELECT username INTO usuario
14  FROM usuarios
15  WHERE activo = 1;
16
17  INSERT INTO bitacora VALUES (lastID+1, usuario, 'MODIFICAR', 'propiedades_en_mercado', noID, CURRENT_DATE, NOW());
18 END

```

4.16 Registrar Propiedad Vendida

Este trigger se activa después de modificar un registro en la tabla de propiedades vendidas. Durante su ejecución, primero obtiene el ID de la propiedad que ha sido modificada desde la tabla de propiedades vendidas. Luego, cuenta el número de operaciones registradas en la tabla de bitácora para determinar el último ID de operación. Después, obtiene el nombre de usuario activo. Finalmente, registra la operación de modificación de la propiedad en la bitácora, pero se debe corregir el nombre de la tabla afectada, que debería ser 'propiedades_vendidas' en lugar de 'propiedades_en_mercado'.

Name:	register_m_propiedaddv	Definer:	admin@%
On table:	propiedades_vendidas		
Event:	AFTER	UPDATE	

Trigger statement: (e.g. "SET NEW.columnA = TRIM(OLD.columnA)")

```

1 BEGIN
2   DECLARE lastID INT;
3   DECLARE noID INT;
4   DECLARE usuario VARCHAR(20);
5
6   SELECT idPropiedad INTO noID
7   FROM propiedades_vendidas
8   WHERE idPropiedad = NEW.idPropiedad;
9
10  SELECT COUNT(idOperacion) INTO lastID
11  FROM bitacora;
12
13  SELECT username INTO usuario
14  FROM usuarios
15  WHERE activo = 1;
16
17  INSERT INTO bitacora VALUES (lastID+1, usuario, 'MODIFICAR', 'propiedades_en_mercado', noID, CURRENT_DATE, NOW());
18 END

```

4.17 Registrar Usuario

Este trigger se activa después de modificar un registro en la tabla de usuarios, pero solo si el campo de contraseña (` `passwd`) ha sido modificado. Durante su ejecución, primero verifica si la contraseña nueva (` `NEW.passwd`) es diferente a la contraseña anterior (` `OLD.passwd`). Si es así, obtiene el nombre de usuario activo y cuenta el número de operaciones registradas en la tabla de bitácora para determinar el último ID de operación. Luego, recupera el número de identidad del usuario modificado. Finalmente, registra la operación de modificación del usuario en la bitácora, incluyendo detalles como el ID de operación, nombre de usuario, tipo de operación, tabla afectada, número de identidad del usuario modificado y la fecha y hora actual.

Name:	register_m_usuario	Definer:	admin@%
On table:	usuarios		
Event:	BEFORE	UPDATE	

```

Trigger statement: (e.g. "SET NEW.columnA = TRIM(OLD.columnA)")

1 BEGIN
2
3     DECLARE noID INT;
4     DECLARE lastID INT;
5     DECLARE usuario VARCHAR(20);
6     IF NEW.passwd <> OLD.passwd then
7         SELECT username INTO usuario
8         FROM usuarios
9         WHERE activo = 1;
10
11        SELECT COUNT(idOperacion) INTO lastID
12        FROM bitacora;
13
14        SELECT noIdentidad INTO noID
15        FROM usuarios
16        WHERE noIdentidad = NEW.noIdentidad;
17
18        INSERT INTO bitacora VALUES (lastID+1, usuario, 'MODIFICAR', 'usuarios', noID, CURRENT_DATE, NOW());
19    END if;
20 END;

```

4.18 Registrar Vendedor

Este trigger se activa después de modificar un registro en la tabla de vendedores. Durante su ejecución, primero obtiene el nombre de usuario activo. Luego, cuenta el número de operaciones registradas en la tabla de bitácora para determinar el último ID de operación. Despues, recupera el número de identidad del vendedor modificado. Finalmente, registra la operación de modificación del vendedor en la bitácora, incluyendo detalles como el ID de operación, nombre de usuario, tipo de operación, tabla afectada, número de identidad del vendedor modificado y la fecha y hora actual.

Name:	register_m_vendedor	Definer:	admin@%
On table:	vendedores		
Event:	AFTER	UPDATE	

```

Trigger statement: (e.g. "SET NEW.columnA = TRIM(OLD.columnA)")

1 BEGIN
2
3     DECLARE noID INT;
4     DECLARE lastID INT;
5     DECLARE usuario VARCHAR(20);
6
7     SELECT username INTO usuario
8     FROM usuarios
9     WHERE activo = 1;
10
11    SELECT COUNT(idOperacion) INTO lastID
12    FROM bitacora;
13
14    SELECT noIdentidad INTO noID
15    FROM vendedores
16    WHERE noIdentidad = NEW.noIdentidad;
17
18    INSERT INTO bitacora VALUES (lastID+1, usuario, 'MODIFICAR', 'vendedores', noID, CURRENT_DATE, NOW());
19 END;

```

5. Views

5.1 Ventas de Agente por Promedio Anual

Este procedimiento almacenado está diseñado para analizar las ventas de propiedades por parte de agentes inmobiliarios en un año específico. Utiliza una vista llamada `vw_agentexfechaVenta` para obtener datos relevantes, como la identificación del agente, el nombre, el año de la transacción, el precio de venta de la propiedad y las fechas de publicación y venta. Esta vista proporciona una herramienta útil para evaluar el rendimiento de los agentes y entender mejor las dinámicas del mercado inmobiliario en un contexto temporal específico.

Name:	vwp_agentexpromedioAnual	Definer:	admin@%
Comment:			
Type:	Procedure (doesn't return a result)	Data access:	Contains SQL
Returns:		SQL Security:	Definer
<input type="checkbox"/> Deterministic			
routine body:			
1	BEGIN		
2	SELECT noIdentidad,nombre,año,AVG(precio) AS precio_promedio, avg(to_days(fechaVenta) - to_days(fechaPublicacion))		
3	AS tiempo_promedio_en_dias		
4	FROM vw_agentexfechaVenta		
5	WHERE año = user_year		
6	GROUP BY noIdentidad		
7	ORDER BY noIdentidad;		
8	END		

5.2 Valor de Venta por Agente

Este es un procedimiento almacenado que identifica al agente inmobiliario con el mayor valor total de ventas en un año específico, utilizando la vista `vw_agentexyearVenta`. Selecciona la identificación del agente, su nombre, la suma total de los precios de las propiedades vendidas y el tiempo de venta. Los resultados se filtran y agrupan por el año proporcionado, se ordenan por valor total en orden descendente y se limitan a un solo resultado, proporcionando así una rápida visión del agente más exitoso en términos de ventas durante ese período.

Name:	vwp_agentexvalorVenta	Definer:	admin@%
Comment:			
Type:	Procedure (doesn't return a result)	Data access:	Contains SQL
Returns:		SQL Security:	Definer
<input type="checkbox"/> Deterministic			
routine body:			
1	BEGIN		
2	SELECT noIdentidad, nombre, sum(precio) AS valor_total, tiempo		
3	FROM vw_agentexyearVenta WHERE tiempo = user_year		
4	GROUP BY noIdentidad		
5	ORDER BY valor_total DESC LIMIT 1;		
6	END		

5.3 Agentes por Fecha de Venta

Este es un comando de consulta que selecciona datos de la tabla `agentes` y la tabla `propiedades_vendidas`, uniéndolas mediante una condición de coincidencia en la columna `noldentidad` de los agentes y la columna `noldentidad_Agente` de las propiedades vendidas. La consulta selecciona la identificación del agente, su nombre, el año de la fecha de venta, el precio de venta de la propiedad, así como las fechas de venta y publicación de la propiedad vendida.

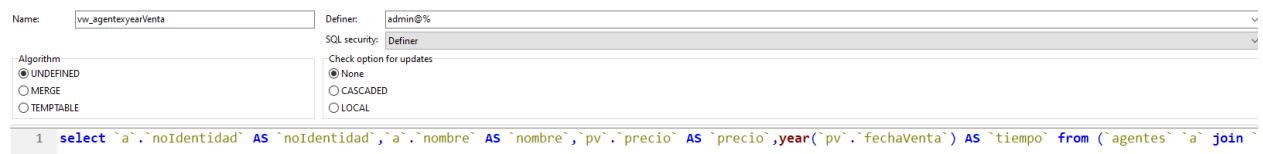
```
select `a`.`noldentidad` AS `noldentidad`, `a`.`nombre` AS
`nombre`, year(`pv`.`fechaVenta`) AS `año`, `pv`.`precio` AS
`precio`, `pv`.`fechaVenta` AS `fechaVenta`, `pv`.`fechaPublicacion` AS
`fechaPublicacion` from (`agentes` `a` join `propiedades_vendidas` `pv`
on(`a`.`noldentidad` = `pv`.`noldentidad_Agente`))
//Code completo
```



5.4 Agente por Año de Venta

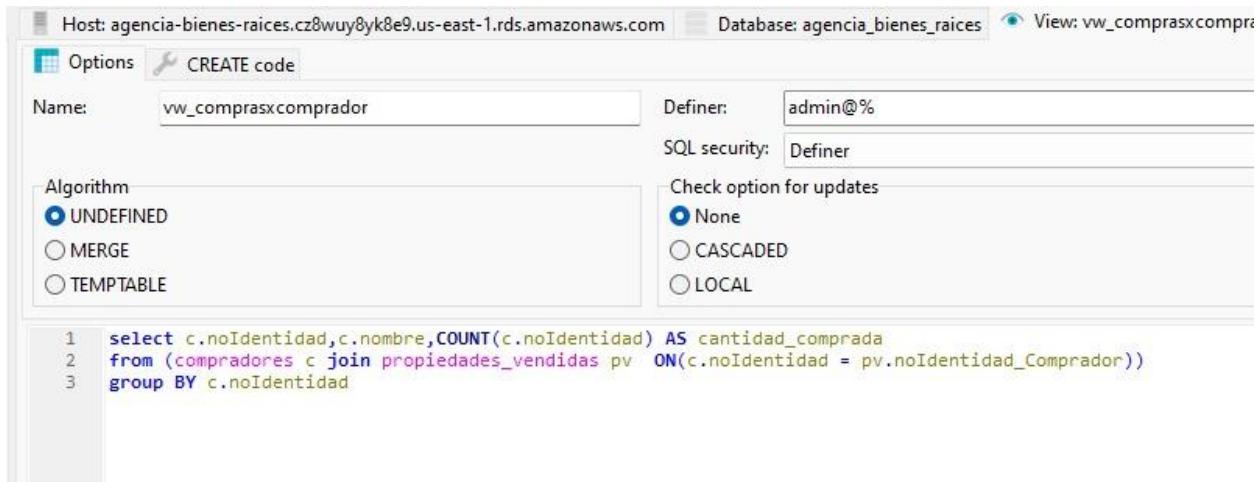
Este es un comando de consulta que selecciona datos de las tablas `agentes` y `propiedades_vendidas`, unidas mediante una condición de coincidencia en la columna `noldentidad` de los agentes y la columna `noldentidad_Agente` de las propiedades vendidas. La consulta selecciona la identificación del agente, su nombre, el precio de venta de la propiedad y el año de la fecha de venta de la propiedad. Luego, ordena los resultados en orden descendente según el precio de venta de las propiedades.

```
select `a`.`noldentidad` AS `noldentidad`, `a`.`nombre` AS
`nombre`, `pv`.`precio` AS `precio`, year(`pv`.`fechaVenta`) AS `tiempo` from
(`agentes` `a` join `propiedades_vendidas` `pv` on(`a`.`noldentidad` =
`pv`.`noldentidad_Agente`)) order by `pv`.`precio` desc
//Code completo
```



5.5 Compras por Comprador

Este es un comando de consulta que selecciona datos de las tablas `compradores` y `propiedades_vendidas`, unidas mediante una condición de coincidencia en la columna `noldentidad` de los compradores y la columna `noldentidad_Comprador` de las propiedades vendidas. La consulta cuenta la cantidad de propiedades compradas por cada comprador y agrupa los resultados por la identificación del comprador y su nombre.



Host: agencia-bienes-raices.cz8wuy8yk8e9.us-east-1.rds.amazonaws.com Database: agencia_bienes_raices View: vw_comprasxcomprador

Options CREATE code

Name: vw_comprasxcomprador Definer: admin@%

SQL security: Definer

Algorithm:

- UNDEFINED
- MERGE
- TEMPTABLE

Check option for updates:

- None
- CASCDED
- LOCAL

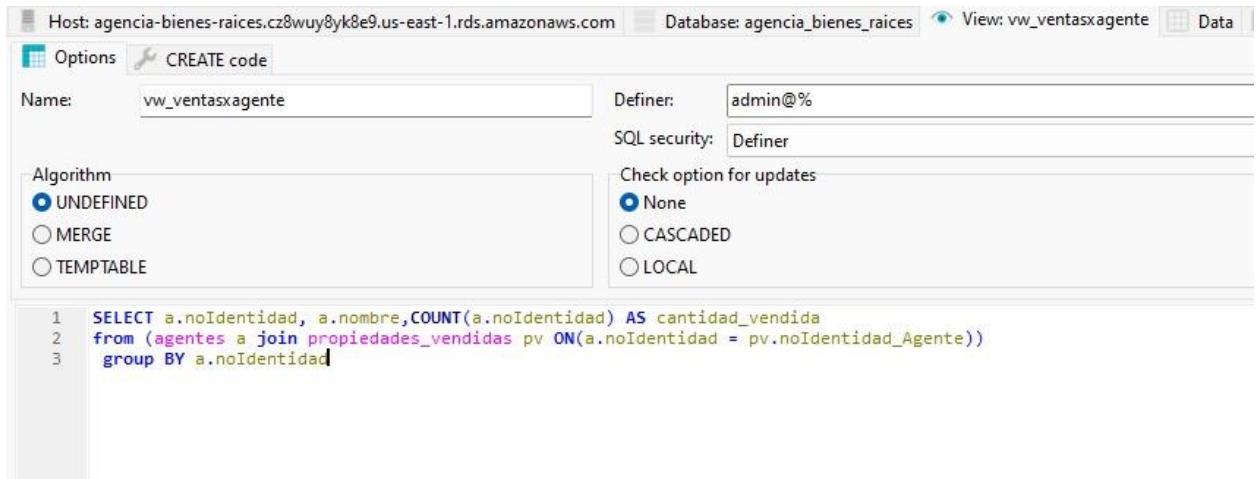
```

1 select c.noldentidad,c.nombre,COUNT(c.noldentidad) AS cantidad_comprada
2 from (compradores c join propiedades_vendidas pv ON(c.noldentidad = pv.noldentidad_Comprador))
3 group BY c.noldentidad

```

5.6 Ventas por Agente

Este es un comando de consulta que selecciona datos de las tablas `agentes` y `propiedades_vendidas`, unidas mediante una condición de coincidencia en la columna `noldentidad` de los agentes y la columna `noldentidad_Agente` de las propiedades vendidas. La consulta cuenta la cantidad de propiedades vendidas por cada agente y agrupa los resultados por la identificación del agente y su nombre.



Host: agencia-bienes-raices.cz8wuy8yk8e9.us-east-1.rds.amazonaws.com Database: agencia_bienes_raices View: vw_ventasxagente

Options CREATE code

Name: vw_ventasxagente Definer: admin@%

SQL security: Definer

Algorithm:

- UNDEFINED
- MERGE
- TEMPTABLE

Check option for updates:

- None
- CASCDED
- LOCAL

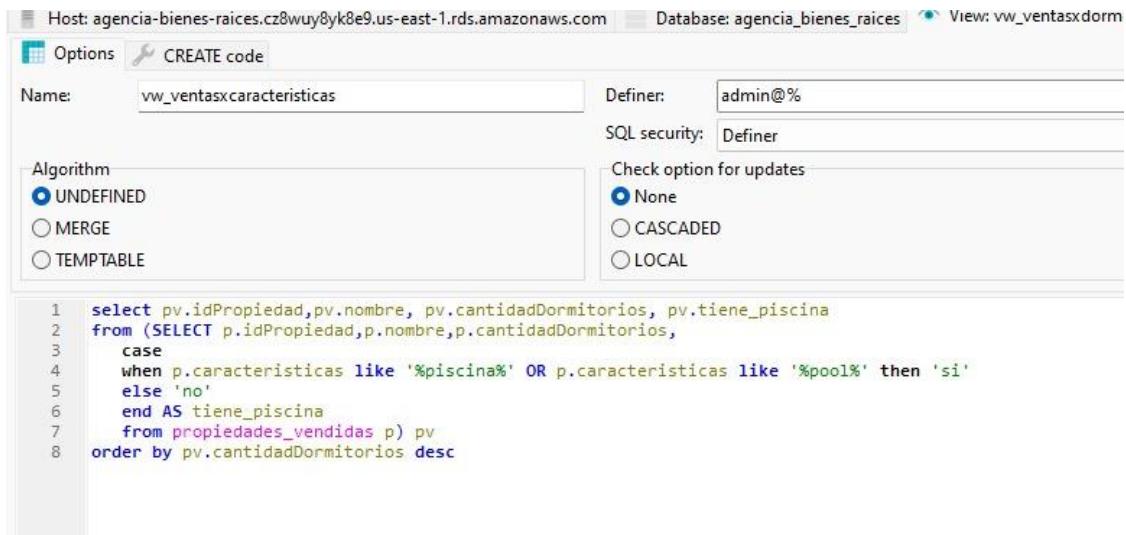
```

1 SELECT a.noldentidad, a.nombre,COUNT(a.noldentidad) AS cantidad_vendida
2 from (agentes a join propiedades_vendidas pv ON(a.noldentidad = pv.noldentidad_Agente))
3 group BY a.noldentidad

```

5.7 Ventas por Característica

Este es un comando de consulta que selecciona datos de la tabla `propiedades_vendidas` y determina si cada propiedad tiene una piscina. Utiliza una subconsulta para crear una columna adicional llamada `tiene_piscina`, que se establece en "sí" si las características de la propiedad incluyen "piscina" o "pool", y "no" en caso contrario. Luego, selecciona el identificador de propiedad, nombre, cantidad de dormitorios y si tiene piscina o no, ordenando los resultados por la cantidad de dormitorios en orden descendente.



```

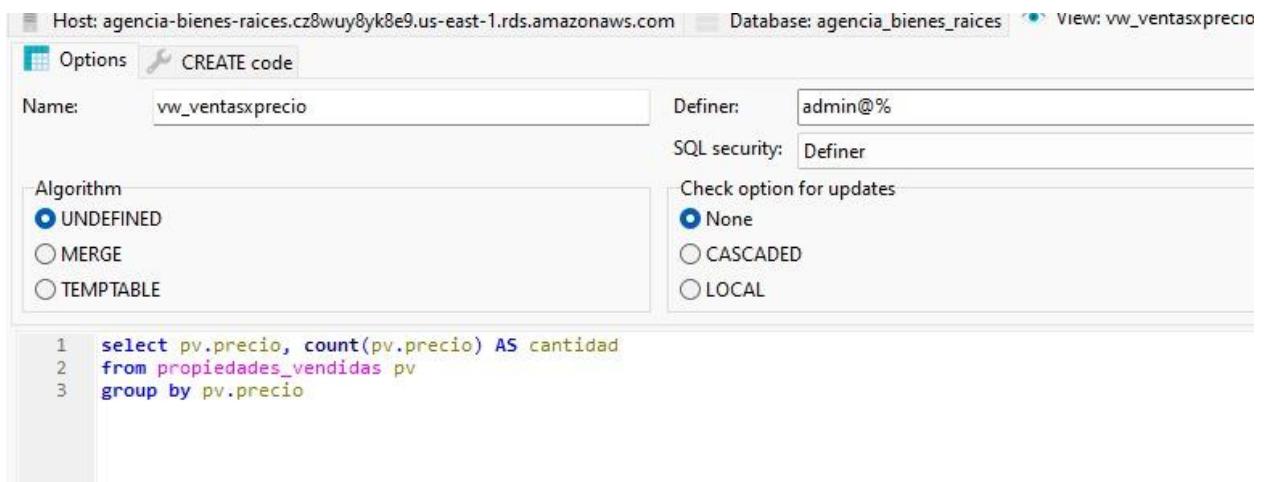
Host: agencia-bienes-raices.cz8wuy8yk8e9.us-east-1.rds.amazonaws.com Database: agencia_bienes_raices View: vw_ventasxcaracteristicas
Options CREATE code
Name: vw_ventasxcaracteristicas Definer: admin@%
SQL security: Definer
Algorithm:
 UNDEFINED
 MERGE
 TEMPTABLE
Check option for updates:
 None
 CASCDED
 LOCAL

1 select pv.idPropiedad, pv.nombre, pv.cantidadDormitorios, pv.tiene_piscina
2 from (SELECT p.idPropiedad, p.nombre, p.cantidadDormitorios,
3 case
4 when p.caracteristicas like '%piscina%' OR p.caracteristicas like '%pool%' then 'si'
5 else 'no'
6 end AS tiene_piscina
7 from propiedades_vendidas p) pv
8 order by pv.cantidadDormitorios desc

```

5.8 Ventas por Precio

Este es un comando de consulta que selecciona el precio de venta de las propiedades vendidas y cuenta cuántas propiedades se vendieron a cada precio. Agrupa los resultados por precio y muestra la cantidad de propiedades vendidas a ese precio.



```

Host: agencia-bienes-raices.cz8wuy8yk8e9.us-east-1.rds.amazonaws.com Database: agencia_bienes_raices View: vw_ventasxprecio
Options CREATE code
Name: vw_ventasxprecio Definer: admin@%
SQL security: Definer
Algorithm:
 UNDEFINED
 MERGE
 TEMPTABLE
Check option for updates:
 None
 CASCDED
 LOCAL

1 select pv.precio, count(pv.precio) AS cantidad
2 from propiedades_vendidas pv
3 group by pv.precio

```

5.9 Ventas por Ubicación

Este comando de consulta selecciona la ciudad de las propiedades vendidas y cuenta cuántas propiedades se vendieron en cada ciudad. Luego, agrupa los resultados por ciudad y muestra la cantidad de propiedades vendidas en cada una.

Host: agencia-bienes-raices.cz8wuy8yk8e9.us-east-1.rds.amazonaws.com Database: agencia_bienes_raices View: vw_ventasxubicacion

Name: vw_ventasxubicacion Definer: admin@%

Algorithm: UNDEFINED SQL security: Definer

Check option for updates: None

MERGE CASCDED LOCAL

```

1 select pv.ciudad, count(pv.ciudad) AS cantidad_vendida
2 from propiedades_vendidas pv
3 group by pv.ciudad

```

5.10 Ventas por Vendedor

Este comando de consulta selecciona la identificación y el nombre de los vendedores, y cuenta cuántas propiedades cada vendedor ha vendido. Utiliza una combinación interna entre las tablas `vendedores` y `propiedades_vendidas` basada en la coincidencia de la identificación del vendedor en ambas tablas. Luego, agrupa los resultados por la identificación del vendedor y muestra la cantidad de propiedades vendidas por cada uno.

Host: agencia-bienes-raices.cz8wuy8yk8e9.us-east-1.rds.amazonaws.com Database: agencia_bienes_raices View: vw_ventasxvendedor

Name: vw_ventasxvendedor Definer: admin@%

Algorithm: UNDEFINED SQL security: Definer

Check option for updates: None

MERGE CASCDED LOCAL

```

1 SELECT v.noIdentidad, v.nombre, COUNT(v.noIdentidad) AS cantidad_vendida
2 from (vendedores v join propiedades_vendidas pv ON(v.noIdentidad = pv.noIdentidad_Vendedor))
3 group BY v.noIdentidad

```