3636 - Bases de Datos

Segundo Cuatrimestre 2024

Evaluación Grupal

Comisión 01-1900

Turno noche

Grupo 3: Casas en Venta o Alquiler

Alumnos

Battiston, Pablo	44.893.256
De Titto, Lucia	46.501.934
Pizarro, Fabricio	42.643.367
Vignardel, Francisco	45.778.667
Miguez, Alejo	41.667.306

Cuerpo docente

Crespo, Natalia

Lopez, Matías

índice

Consignas

Diseño

1. Elegir uno de los casos señalados en la Sección I. Tener en cuenta que sólo podrá elegirse un caso por grupo, sin poder repetirse.
2. Verificar el link mencionado con el set de datos propuesto. Cada caso tiene un dataset que se debe tener en cuenta para la carga de datos en el modelo diseñado. Será obligatorio en cada trabajo que ese dataset sea cargado, por lo cual se deben tomar como requerimiento inicial de los datos.
3. Verificar el link del modelo de negocio que se requiere diseñar. Cada caso se ejemplifica a través de una web comercial, para que puedan contemplar todos los actores que intervienen en cada modelo. El ejercicio desarrollado debe poder almacenar los datos para que funcione esa página web.
4. Escribir el Requerimiento inicial que se va a Modelar en modo descriptivo.
5. Diseñar un diagrama de Entidad Relación para el modelo de negocios elegido. Se debe contemplar el desarrollo gráfico de un DER. En este caso, se podrá elegir alguna aplicación que permita la diagramación del modelo.

    - ☐ Debe contener no menos de 12 entidades.
    - ☐ Al menos una entidad debe ser débil.
    - ☐ Al menos debe existir alguna relación de jerarquía.
    - ☐ Al menos un tipo de atributo de los explicados en clase.
    - ☐ Alguna de las relaciones debe contemplar un almacenamiento histórico de datos.
    - ☐ Se deben detallar todos los atributos en cada entidad.
    - ☐ Las relaciones deben estar definidas con conjugaciones verbales.
    - ☐ Los nombres de entidades deben ser definidos en singular.
    - ☐ Tanto las relaciones como entidades deben contener nombres únicos y de aspecto descriptivo de su contenido.

- ☐ La lectura del DER será de izquierda a derecha y de arriba hacia abajo.
- ☐ Se debe evitar el entrecruzamiento de líneas, si fuera posible.

6. Generar el Modelo Relacional del DER propuesto en el punto 5.
7. Verificar que el Modelo se encuentre al menos en 3FN. Si no lo está se deberá efectuar el proceso de normalización.

Desarrollo

8. **p_CrearDB**(borrar_si_existe?): Desarrollar un procedimiento almacenado que permita crear la base de datos con sus tablas, claves primarias, claves foráneas y todo lo relacionado al esquema de la base de datos. Según el parámetro enviado, se debe o no borrar la estructura si existía previamente.
9. **p_LimpiarDatos**(): Desarrollar un procedimiento almacenado que permita limpiar todos los datos almacenados en la base de datos, sin borrar estructuras.
10. **p_CagarDataset**(archivo): Desarrollar un procedimiento almacenado que permita cargar el dataset propuesto en cada caso. Se podrá contemplar la carga en una tabla temporal para luego distribuir los datos en las distintas tablas. En el caso que el dataset tuviera datos incorrectos, no consistentes o faltantes, se podría implementar la estrategia de completarlo, corregirlo o suprimirlo. La carga se debe realizar desde el path del archivo enviado por parámetro.
11. **p_CargaAleatoria**(): Desarrollar un procedimiento almacenado que permita cargar el juego de datos que se utilizarán para la generación de filas en cada una de las entidades diseñadas en el modelo. Se deben controlar los errores de carga.
12. **tg_auditar**(): Elegir alguna de las tablas que contenga datos sensibles dentro de cada modelo. Sobre dicha tabla realizar un trigger de auditoría que permita controlar cada una de las acciones realizadas, permitiendo conocer la acción, el momento de la acción, el usuario que lo realizó, el valor cambiado y todo lo que considere relevante.
13. **f_funcion1**(valor): Realizar una función que permita darle formato frecuente a un campo determinado. Se deberá elegir la función, según lo que resulte útil para el modelo diseñado.

14. **f_funcion2**(contraseña): Realizar una función que permita controlar los caracteres permitidos y reglas para el seteo de contraseñas en la aplicación. Devolverá 1 si es válida ó 0 si no cumple las condiciones. Contemplar al menos 5 condiciones de diferente característica, por ejemplo, la longitud.

15. **v_vista1**() **… v_vista3**: Diseñar 3 vistas que permitan listar 3 requerimientos funcionales que puedan presentarse en cada modelo de negocio. Las mismas deben:

    - tener complejidad media/alta, con el uso de joins, agrupaciones, subconsultas, etc.
    - Al menos una de ellas debe mostrar un ranking de los 10 “mejores”/“peores”/”mayores”/”menores”/etc, según el requerimiento elegido.
    - Al menos una de ellas debe tener recursividad con jerarquías.
    - Al menos una de ellas debe contener un análisis anual de algún requerimiento oportuno al modelo.

16. **p_reporte1**() **... p_reporte5**(): Diseñar 5 procedimientos almacenados que permitan listar 5 requerimientos funcionales que puedan presentarse en cada modelo de negocio. Los mismos deben contemplar las siguientes estructuras técnicas:

    a. Al menos 2 de ellos deben recibir parámetros de entrada.
    b. Al menos 2 de ellos deben retornar datos de salida.
    c. Se deben validar que los parámetros de entrada contengan valores esperados.
    d. Se deben contemplar el uso de todos los operadores vistos en clase:
        - Joins
        - Agrupaciones y funciones de agrupación
        - Union / Union all
        - All / any / some
        - Subconsultas
        - exists / in

- with

17. **p_proceso1**(datos_del_usuario): Diseñar un procedimiento almacenado que permita realiza el alta/baja/modificación de un nuevo usuario en nuestro sistema, contemplando diferentes características, tales como que el usuario no exista/exista, que cumpla con las condiciones de contraseñas, que haya completado todos los datos mínimos necesarios, las relaciones que se afectarían, etc.

Resolución

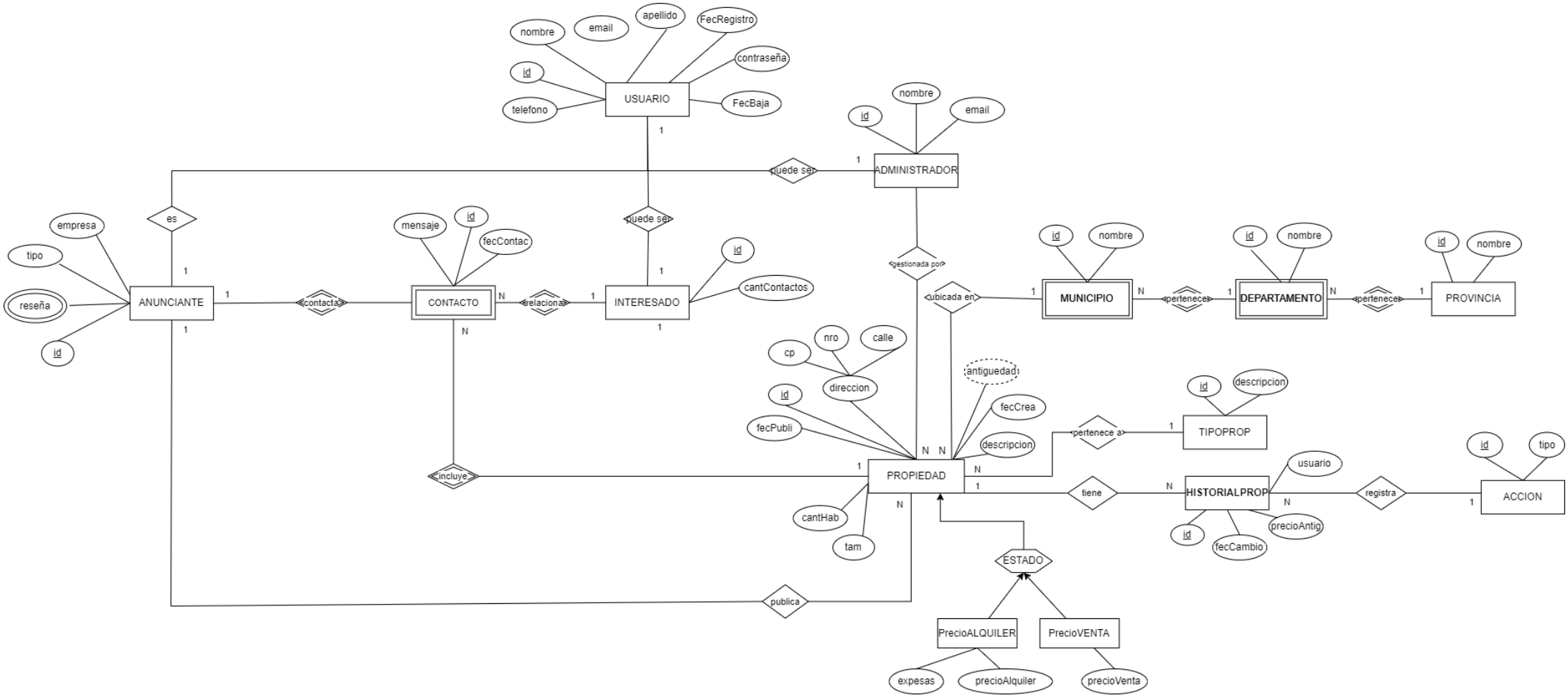
Diseño

4. El modelo de datos a desarrollar será una base de datos relacional que permitirá gestionar la publicación y consulta de **propiedades en venta o alquiler** en todo el territorio argentino. La plataforma permitirá a los **interesados** (potenciales compradores o arrendatarios) ponerse en contacto con los **anunciantes** (propietarios o inmobiliarias) para negociar la compra o alquiler de propiedades fuera del sistema. Debe considerarse que un usuario interesado puede ser, simultáneamente, un anunciante si fuese a publicar una propiedad.

   Este sistema deberá contemplar la publicación de **propiedades** (casas, departamentos, terrenos, etc.), detallando características como el tipo de propiedad, ubicación, precio, estado (venta/alquiler), y otros atributos relevantes. Las propiedades estarán asociadas a **ubicaciones** geográficas, delimitadas por **provincias**, **departamentos** y **municipios**, facilitando la consulta de propiedades por localidad.

   El modelo también deberá contemplar un registro de **contactos** entre los usuarios interesados y los anunciantes, de modo que se registre el historial de interacciones en la plataforma, sin llegar a concretar la transacción dentro de la misma. Los **administradores** del sistema tendrán acceso a gestionar los datos de las propiedades, usuarios, y anunciantes, así como verificar las publicaciones.

5. A continuación, se presenta el diagrama de Entidad-Relación propuesto para el caso analizado:


nombre
email
apellido
FecRegistro
contraseña
id
telefono
USUARIO
FecBaja
1
puede ser
1
id
nombre
email
ADMINISTRADOR
es
puede ser
gestionada por
empresa
tipo
reseña
id
ANUNCIANTE
1
1
contacta
mensaje
id
fecContac
CONTACTO
N
relaciona
1
1
id
cantContactos
INTERESADO
1
N
ubicada en
1
id
nombre
MUNICIPIO
N
pertenece
1
id
nombre
DEPARTAMENTO
N
pertenece
1
id
nombre
PROVINCIA
cp
nro
calle
direccion
id
fecPubli
antiguedad
fecCrea
descripcion
N N
1
PROPIEDAD
N
pertenece a
1
id
descripcion
TIPOPROP
incluye
N
1
tiene
N
HISTORIALPROP
usuario
N
precioAntig
id
fecCambio
registra
1
id
tipo
ACCION
cantHab
tam
N
ESTADO
publica
PrecioALQUILER
PrecioVENTA
expesas
precioAlquiler
precioVenta

6. El Modelo Relaciones que surge a partir del diagrama de Entidad-Relación previo está dado por las siguientes relaciones (PK | **FK**):

USUARIO(IDUsuario, Nombre, Apellido, Email, Teléfono, Contraseña, FechaRegistro, Tipo, FechaBaja)

INTERESADO(IDInteresado, **IDUsuario**, CantContactos)

ANUNCIANTE(IDAnunciante, **IDUsuario**, TipoAnunciante, Empresa, Reseña)

PROPIEDAD(IDPropiedad, **IDAnunciante**, **IDAdministrador**, Cp, Nro, Calle, Precio, **IDTipoPropiedad**, **IDEstadoPropiedad**, **IDMunicipio**, CantHab, Tam, Descripción, FecCrea, FecPubli)

TIPOPROPIEDAD(IDTipoPropiedad, Descripción)

PRECIOALQUILER(**IDPropiedad**, Expensas, PrecioAlquiler)

PRECIOVENTA(**IDPropiedad**, PrecioVenta)

PROVINCIA(IDProvincia, Nombre)

DEPARTAMENTO(IDDepartamento, Nombre, **IDProvincia**)

MUNICIPIO(IDMunicipio, Nombre, **IDProvincia, IDDepartamento**)

HISTORIALPROPIEDAD(IDHistorial, **IDPropiedad**, FecCambio, PrecioAntiguo, Usuario, **Accion**)

ACCION(IDAccion, tipo)

ADMINISTRADOR(IDAdministrador, Nombre, Email)

CONTACTO(IDContacto, **IDInteresado**, **IDAnunciante**, FecContac, Mensaje)

7. A partir del Modelo Relacional generado, se proponen los siguientes esquemas de Dependencias Funcionales correspondientes a cada relación:
   Usuario
   A = { {IDUsuario → Nombre, Apellido, Contraseña, Email, Teléfono, FechaRegistro, Tipo,FechaRegistro},

{Nombre, Apellido, Contraseña, Email, Teléfono, FechaRegistro, Tipo,FechaRegistro → IDUsuario},

{Email → IDUsuario, Nombre, Apellido, Contraseña, Teléfono, FechaRegistro, Tipo,FechaRegistro} }

CC = { {IDUsuario}, {Nombre, Apellido, Contraseña, Email, Teléfono, FechaRegistro, Tipo,FechaRegistro}, {Email} } por lo que se encuentra en FNBC.

Anunciante

B = { {IDAnunciante → IDUsuario, TipoAnunciante, Empresa, Reseña},
{IDUsuario → IDAnunciante} }

CC = { {IDAnunciante}, {IDUsuario} } por lo que se encuentra en FNBC.

Interesado

C = { {IDInteresado → IDUsuario, CantContactos},
{IDUsuario → IDInteresado} }

CC = { {IDInteresado}, {IDUsuario} } por lo que se encuentra en FNBC.

Propiedad

D = { {IDPropiedad → IDAnunciante, IDAdministrador, Cp, Nro, Calle, Precio, TipoPropiedad, EstadoPropiedad, IDMunicipio, CantHab, Tam, Descripción, FecCrea, FecPubli},
{IDAnunciante, IDAdministrador, IDTipoPropiedad, IDEstadoPropiedad, IDMunicipio → IDPropiedad} }

CC = { {IDPropiedad}, {IDAnunciante, IDAdministrador, IDTipoPropiedad, IDEstadoPropiedad, IDMunicipio} } por lo que se encuentra en FNBC.

TipoPropiedad

E = { IDTipoPropiedad → DescripcionPropiedad }

CC = { {IDTipoPropiedad} } por lo que se encuentra en FNBC.

PrecioVenta

F = { IDPropiedad → PrecioVenta }

CC = { {IDPropiedad } } por lo que se encuentra en FNBC.

PrecioAlquiler

F = { IDPropiedad → PrecioAlquiler, Expensas }

CC = { {IDPropiedad } } por lo que se encuentra en FNBC.

Provincia

G = {IDProvincia → Nombre}

CC = { {IDProvincia} } por lo que se encuentra en FNBC.

Departamento

H = { {IDDepartamento → Nombre},
{IDProvincia → IDDepartamento, Nombre} }

CC = { {IDDepartamento}, {IDProvincia} } por lo que se encuentra en FNBC.

Municipio

I = { {IDMunicipio → Nombre},
{IDDepartamento, IDProvincia → IDMunicipio, Nombre} }

CC = { {IDMunicipio}, {IDDepartamento, IDProvincia} } por lo que se encuentra en FNBC.

Acción

J = { {IDAccion → Tipo} }

CC = { { IDAccion } } por lo que se encuentra en FNBC.

Historial

K = { {IDHistorial → IDPropiedad, FecCambio, PrecioAntiguo},
{IDPropiedad, FecCambio → IDHistorial, PrecioAntiguo} }

CC = { {IDHistorial}, {IDPropiedad, FecCambio} } por lo que se encuentra en FNBC.

Administrador

L = { {IDAdministrador → Nombre, Email},
{Email → IDAdministrador, Nombre} }

CC = { {IDAdministrador}, {Email} } por lo que se encuentra en FNBC.

Contacto

M = { {IDContacto → IDInteresado, IDAnunciante, FecContac, Mensaje},
{IDInteresado, IDAnunciante, FecContac → IDContacto, Mensaje} }

CC = { {IDContacto}, {IDInteresado, IDAnunciante, FecContac} } por lo que se encuentra en FNBC.

Habiendo demostrado que todos los conjuntos de dependencias funcionales asociados a relaciones del Modelo Relacional se encuentran en Forma Normal de Boyce-Codd, queda en evidencia que el conjunto de

DFs S = A U B U C U D U E U F U G U H U I U J U K U L U M

resulta encontrarse en Forma Normal de Boyce-Codd.

Desarrollo

8. El procedimiento almacenado **p_CrearDB** está diseñado para crear la base de datos **Argenprop** según el Diagrama Entidad-Relación (DER) y el Modelo Relacional (MR). Su función es generar las tablas, claves primarias, claves foráneas y relaciones entre las entidades.

   Parámetro @borrar_si_existe:

   Si @borrar_si_existe = 1, se elimina la base de datos actual con:

   > *ALTER DATABASE Argenprop SET SINGLE_USER WITH ROLLBACK IMMEDIATE; DROP DATABASE Argenprop;*

   Si @borrar_si_existe = 0, solo se crea si no existe.

   El procedimiento **p_CrearDB** está diseñado de manera modular, con cada tabla definida en un procedimiento almacenado separado. El procedimiento principal **p_CrearDB** llama a estos procedimientos de forma ordenada para asegurar que las tablas y relaciones se creen correctamente según el DER y MR.

9. El procedimiento almacenado **p_LimpiarDatos** elimina todos los datos de la base de datos Argenprop sin alterar su estructura. Se eliminan los datos en un orden que respeta las dependencias entre tablas, comenzando por aquellas que dependen de otras a través de claves foráneas.

   El procedimiento se ejecuta con EXEC p_LimpiarDatos.

10. El procedimiento almacenado **p_CargarDataset** permite cargar datos desde archivos CSV en las tablas de la base de datos Argenprop. Se crea una tabla temporal para almacenar temporalmente los datos del archivo y luego se distribuyen a las tablas correspondientes (PROVINCIA, DEPARTAMENTO, MUNICIPIO).

    > Cargar datos: Según el tipo de dataset (Provincias, Departamentos, Municipios), el procedimiento invoca el correspondiente sub-procedimiento (p_CargarProvincias, p_CargarDepartamentos, p_CargarMunicipios).
    >
    > Carga Archivos CSV: Los datos se cargan en tablas temporales mediante BULK INSERT desde un archivo CSV especificado por el parámetro @filePath. Se utiliza una expresión dinámica con sp_executesql para el BULK INSERT debido a que no es posible pasar directamente el parámetro

@filePath al BULK INSERT en un procedimiento almacenado. De esta manera, se construye el comando SQL como una cadena y se ejecuta dinámicamente.

Asignación de IDs: Para departamentos y municipios, se asignan los IDProvincia o IDDepartamento de manera aleatoria.

Limpieza: Después de insertar los datos, las tablas temporales se eliminan para liberar recursos.

11. El procedimiento almacenado **p_CargaAleatoria** genera datos de prueba para las diferentes tablas del sistema Argenprop. A partir de tres parámetros (@numUsuarios, @numPropiedades, y @numContactos), el procedimiento inserta registros en las tablas relacionadas:

    Usuarios: Crea usuarios con datos simulados.

    Anunciantes e Interesados: Divide los usuarios en anunciantes e interesados y los asigna aleatoriamente.

    Administradores: Selecciona usuarios restantes para ser administradores.

    Propiedades: Crea propiedades vinculadas a anunciantes, municipios, departamentos y provincias, generando detalles como estado (alquiler o venta), tipo de propiedad y fechas.

    Precios: Genera precios aleatorios para las propiedades en alquiler y venta.

    Contactos: Crea contactos aleatorios entre interesados y anunciantes, vinculándolos a propiedades.

    El procedimiento utiliza ciclos WHILE y selección aleatoria con NEWID() y RAND() para generar los datos

12. El trigger **tg_auditar** registra cambios en la tabla PRECIOVENTA. Después de cualquier inserción, actualización o eliminación, guarda en la tabla HISTORIALPROPIEDAD el ID de la propiedad, el precio antiguo, el usuario que realizó la acción, y el tipo de acción (INSERT, UPDATE o DELETE).

13. La función **f_funcion1** convierte Conversión de divisas (USD) a pesos argentinos (ARS) utilizando una tasa de cambio proporcionada como parámetro. Retorna el valor convertido con formato numérico y la unidad monetaria 'ARS'.

14. La función **f_funcion2** valida una contraseña según cinco reglas específicas y devuelve 1 si cumple con todas, o 0 si no cumple. Entre las reglas establecidas se encuentran:
    a. La contraseña debe tener más de 8 caracteres.
    b. La contraseña debe tener al menos una minúscula (a-z).
    c. La contraseña debe tener al menos una mayúscula (A-Z).
    d. La contraseña debe tener al menos un carácter numérico (0-9).
    e. La contraseña debe tener al menos un carácter especial (! " # $ % & ' ( ) * + , - . / : ; < = > ? @ [\ ] ^ _` { | } ~).

15. Para la resolución de este ejercicios, se optó por realizar tres vistas que responden a
    a. **v_vista1()**: Selecciona las 10 casas más caras. Usa INNER JOIN entre tablas PROPIEDAD, TIPOPROPIEDAD y PRECIOVENTA, filtrando por casas y ordenado por precio de venta de mayor a menor

    b. **v_vista2()**: Muestra el análisis anual de ventas. Utiliza LEFT JOIN y GROUP BY para agrupar propiedades por calle y tipo, contando los contactos de este año.

    c. **v_vista3()**: Selecciona los usuarios, jerarquizados bajo su condición de interesados o anunciantes. Combina propiedades en venta y alquiler mediante un WITH y el uso de UNION ALL, mostrando detalles como calle, número y precio de cada propiedad

16. Para la resolución de este ejercicios, se optó por realizar cinco procedimientos que responden a:
    a. **p_reporte1**(@fechaIni,@fechaFin,@anunciante): Todos los contactos de un anunciante durante un periodo de tiempo.
    b. **p_reporte2**(@precioMin,@precioMax): Propiedades en venta dentro de un rango de precios.
    c. **p_reporte3**(@Tipo,@Estado,@Provincia): Propiedades por tipo, estado o provincia.
    d. **p_reporte4**(@Municipio,@PrecioEnUSD,@ExistePropiedad[OUTPUT]): Propiedades en un municipio y precio igual o menor al ingresado.
    e. **p_reporte5**(@Estado,@Cantidad[OUTPUT]): Cantidad de propiedades en un estado dado.
17. El procedimiento almacenado **p_proceso1** recibe como argumentos los datos del usuario a manipular siendo:
    a. Acción (ALTA, BAJA o MODIFICACIÓN)

b. Email
c. Nombre
d. Apellido
e. Teléfono
f. Contraseña
g. TipoUsuario
h. Empresa
i. Reseña
j. TipoAnunciante
k. CantContactos
l. ModificarContactos (entero, sumará o restará según el valor ingresado).

El usuario es identificado mediante su Email, contemplando que el mismo es de carácter único. Dependiendo el tipo de proceso, ya sea por ALTA, BAJA o MODIFICACIÓN, se despliegan diversas operaciones sobre la tabla USUARIO de modo tal que generen la acción. Los tres procesos comparten una validación de datos, ya sea de la validez de los datos ingresados como del estado del usuario en la tabla (si ya fue dado de baja o alta previamente).