**Carlos Felipe Palacio y Santiago Grisales.**

**Cada punto viene separado en diversos archivos, sin embargo, el archivo todo.sql tiene todos los puntos que requieran algo de SQL en él. Además, se adjunta el MER.**

**Modelo lógico.**

Diagrama

Descripción generada automáticamente

Analizando individualmente cada tabla sin incluir las relaciones tenemos que:

1. Persona: en ella se almacenará a todas las personas de la base de datos, sin importar si son de Transnópoli o no.

* Id: correspondería a la cédula o id de la persona.
* Nombre: nombre.
* Apellido: apellido.
* Dirección: dirección de la persona.

1. Ciudad: tabla con todas las ciudades que se van a utilizar.

* Id: Id de la ciudad.
* Nombre.

1. Trabajo: tabla donde se almacenan los trabajos de cada persona en la base de datos.

* Id: generado incrementalmente automático.
* Dirección: dirección donde se ubica el trabajo.
* NombreEmpresa.

1. Matrimonio: tabla donde se almacenan los matrimonios entre personas de la base de datos.

* Id: generado incrementalmente automático.
* Fecha: fecha donde se efectuó el compromiso.
* Tipo: ancestral o no ancestral.

1. Georreferencia: tabla donde se almacena la ubicación o ubicaciones geográficas de cada empleado.

* Id: generado incrementalmente automático.
* Latitud: latitud donde se ubica la persona.
* Longitud: longitud donde se ubica la persona.

**Relaciones entre tablas.**

**Persona**.

Tiene dos relaciones de autorreferencia. Esto lo hicimos para poder modelar el hecho de que una persona tiene dos padres. Ambas relaciones son no obligatorias, por lo cual, no es necesario que ambos padres estén en la base de datos para identificarlos. Las personas ancestrales no tienen padres (o al menos no son conocidos o no son relevantes para este modelo). La tabla personas puede contener personas tanto de Transnópoli como externas a ella.

Persona tiene dos relaciones de ciudades para poder identificar en que ciudad nacieron y en qué ciudad residen actualmente.

**Matrimonio.**

La tabla matrimonio tiene dos relaciones de uno a muchos proveniente de la tabla Personas. Esto debido a que las personas pueden tener más de un matrimonio, y son dos relaciones dado que se casan únicamente dos personas. El tipo de matrimonio se utiliza para modelar si lo que se tiene es un matrimonio ancestral, o por el contrario cualquier otro tipo de matrimonio.

**Trabajo.**

Cada persona puede tener por lo menos un trabajo. Por lo cual, esto hace que se tenga una relación de uno a muchos entre ambas tablas.

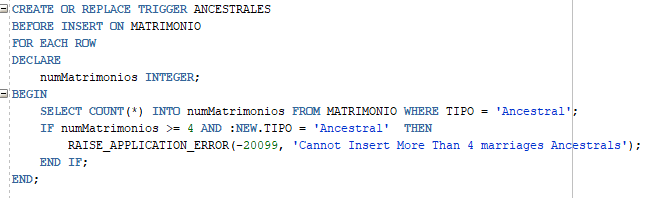
**Georreferencia.**

Cada persona puede tener varias georreferencias. Esto en el caso de que se desee guardar distintas direcciones o algo por el estilo.

**Ciudad.**

Tabla que contendrá todas las ciudades en la base de datos. De ella, por ejemplo, trabajo y persona tomarán su id.

Mediante algunos triggers se modelaron las diferentes restricciones exigidas en el enunciado. Se adjuntan imágenes de los triggers y en el .zip vienen un archivo llamado triggers.sql donde también se encuentran los scripts.



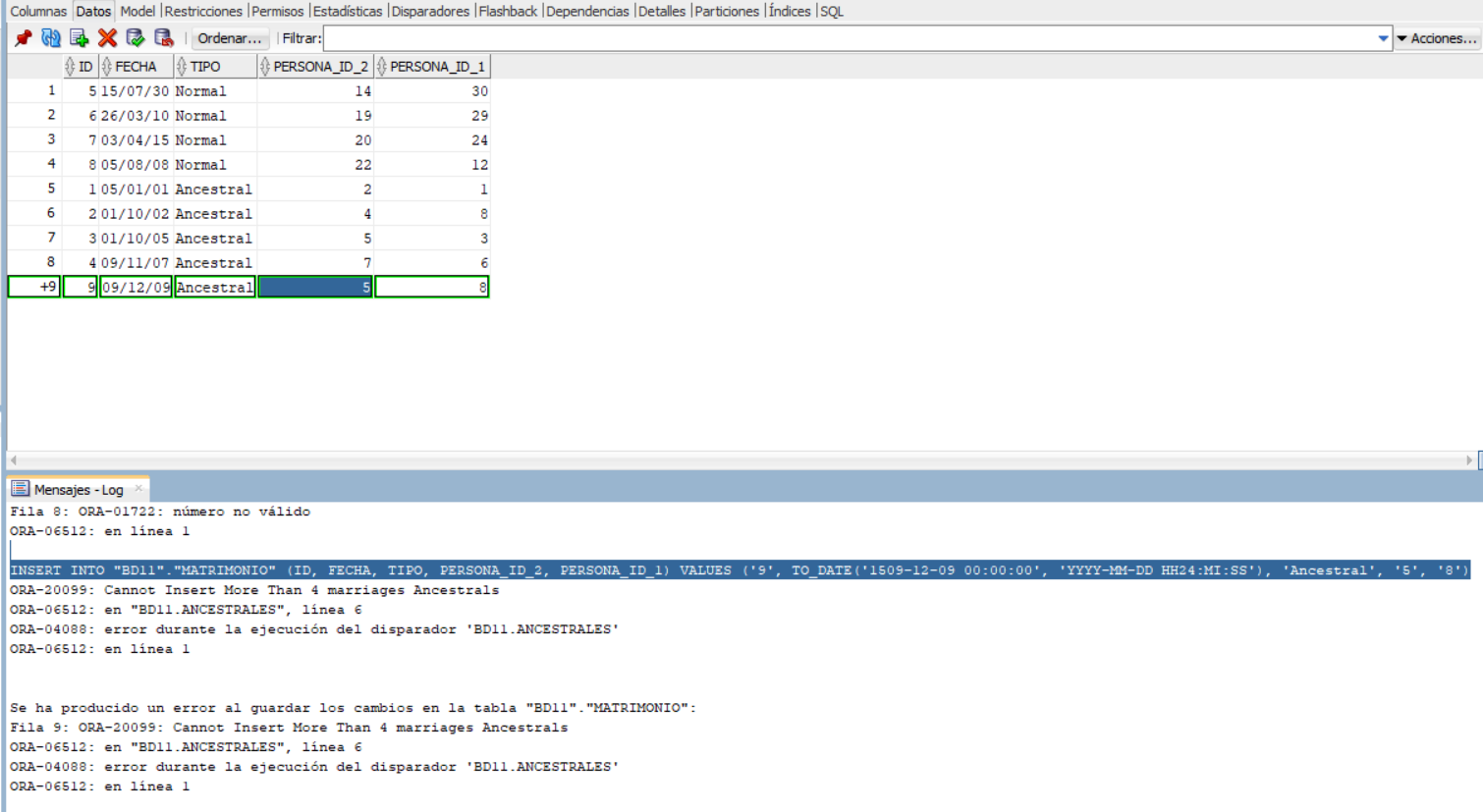
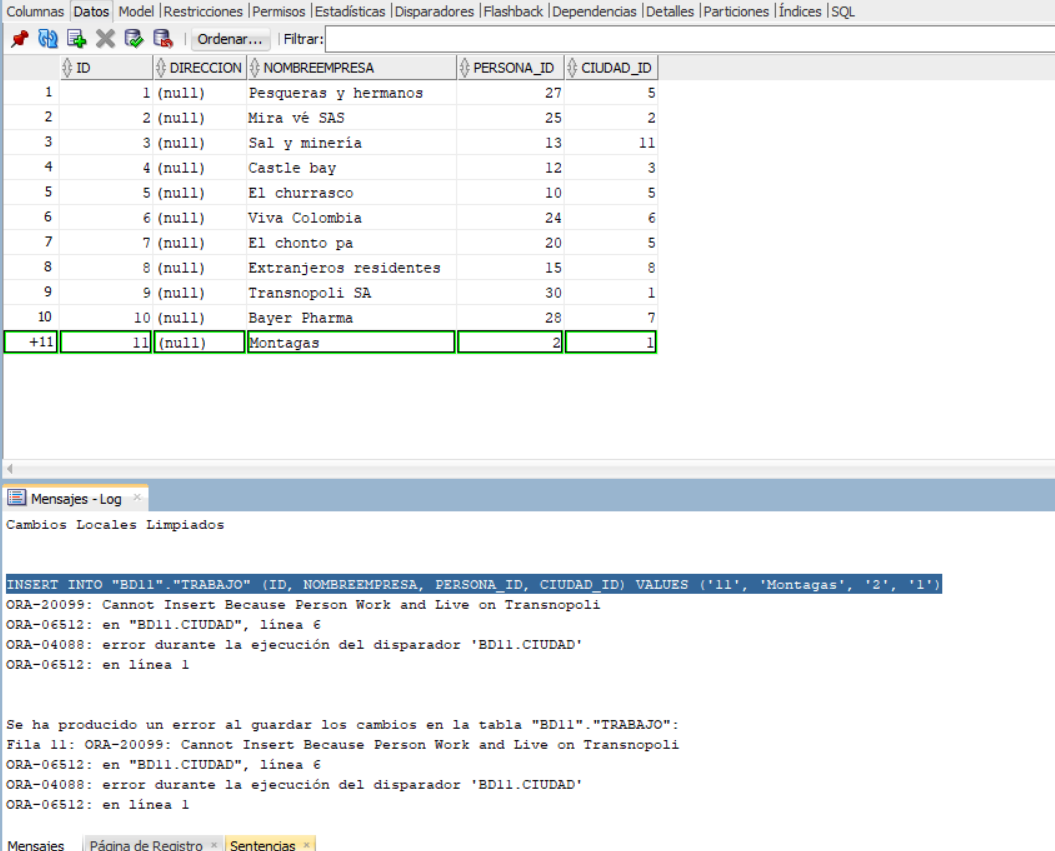
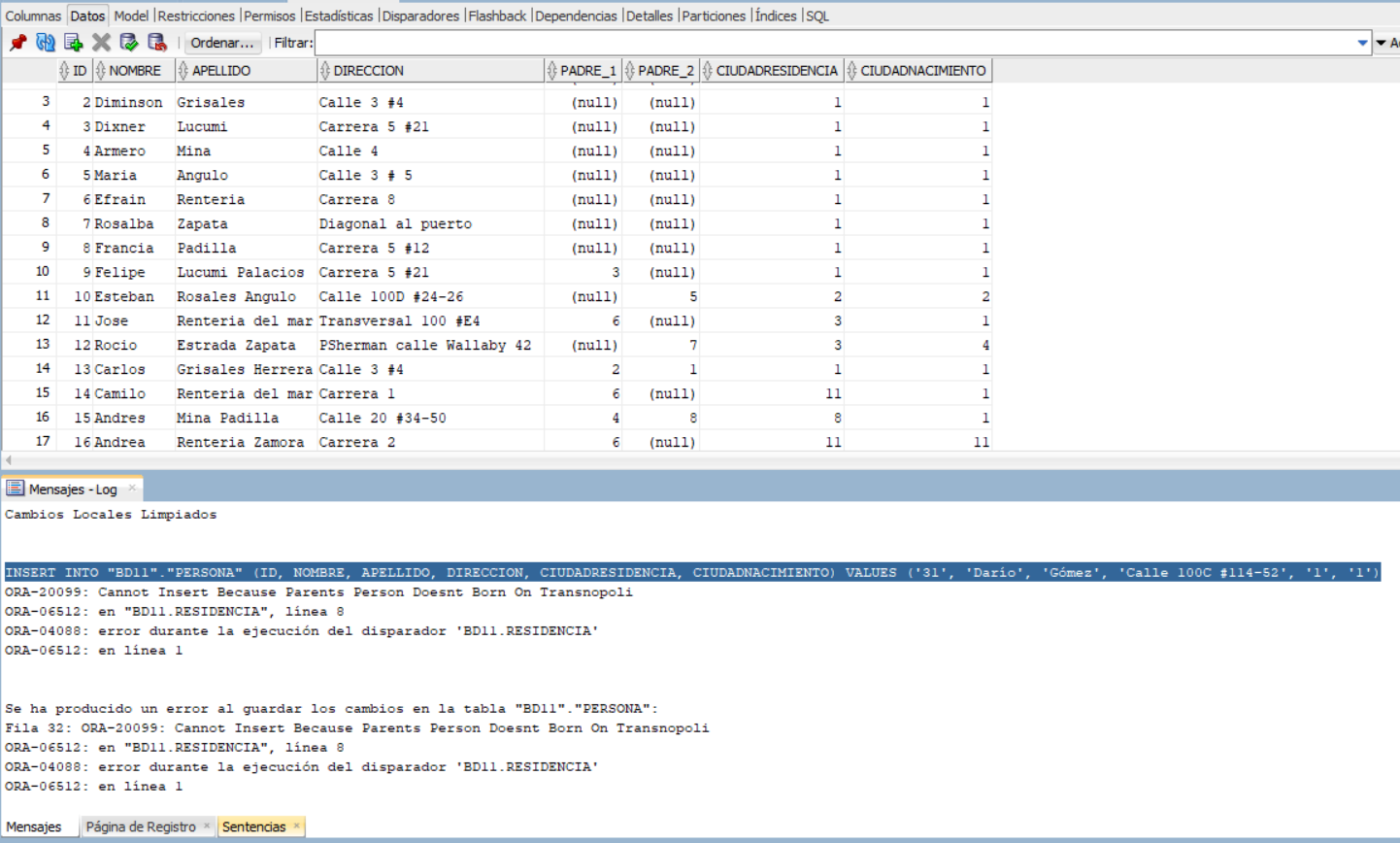
Texto

Descripción generada automáticamente con confianza media

Escala de tiempo

Descripción generada automáticamente

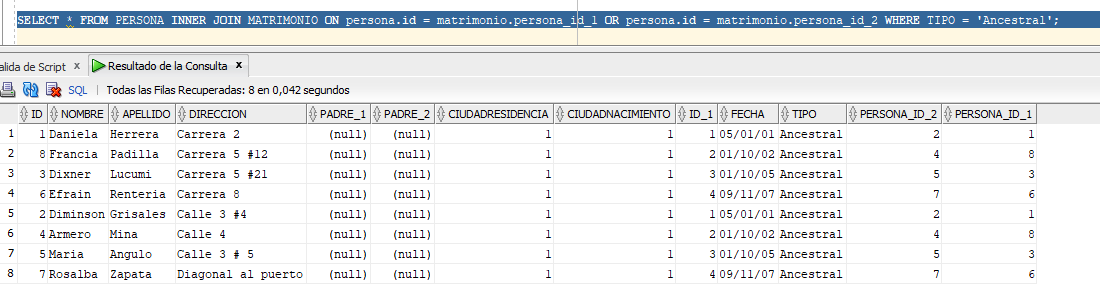
**Restricciones en funcionamiento:**

1. **No se puede insertar más de 4 matrimonios ancestrales.**
2. **No se puede insertar una persona que vive y trabaja en Transnopoli.**
3. **No se puede insertar una persona que vive en Transnopoli si por lo menos uno de sus dos padres no nació.**

**Consulta matrimonios iniciales.**

**El script de las consultas va adjunto en él .zip y se llama consultas.sql.**

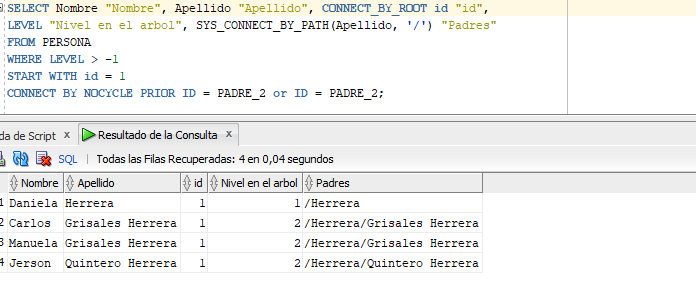
Nota: no entendimos bien lo que quería decir esta consulta, así que hicimos una consulta que mostrara los matrimonios y las personas ancestrales.



**Consulta árbol genealógico.**

**El script de las consultas va adjunto en él .zip y se llama consultas.sql.**

En este caso nos enfrentamos a una consulta recursiva, ya que nuestra tabla de Personas se autorreferencia para así poder guardar ambos padres de una persona. El resultado final fue este:

Donde lo que se llama Padres viene siendo algo así como todas las personas que se necesitan para poder llegar a la persona que se tiene en ese momento. En este caso, la consulta comienza desde el id 1 que es un matrimonio ancestral, y llega hasta un segundo nivel en el árbol. Cada uno de estos parientes es hijo de Herrera con Grisales.

**PL nombrado.**

**Esta función se encuentra en el archivo functions.sql.**

En primera instancia creímos que era necesario solamente aplicar la ecuación para encontrar la distancia de un punto a otro en un plano 2D, pero por eso tuvimos un problema, dado que estábamos trabajando en un plano de 3 dimensiones. Por tanto, el resultado final fue:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

La función degrees es una función auxiliar cuya función es convertir un numero en su correspondiente valor en radianes.

Texto

Descripción generada automáticamente con confianza media

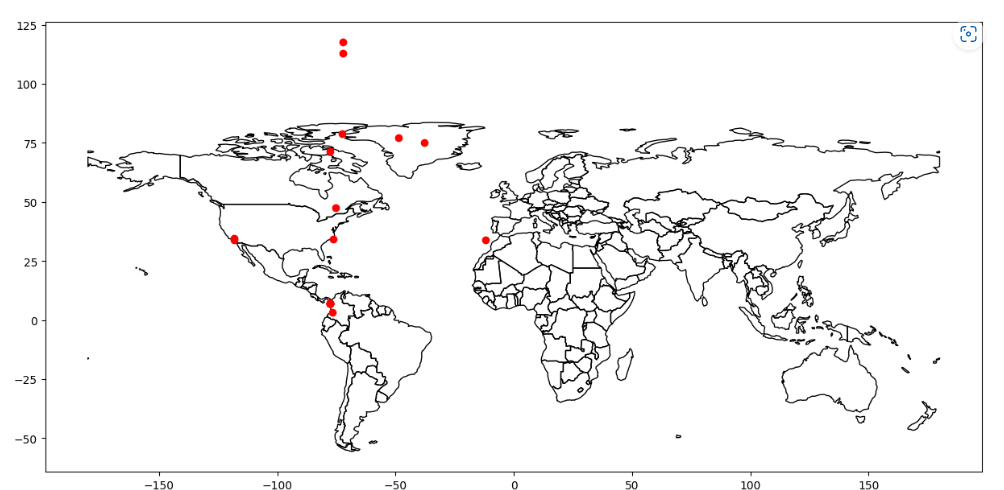
Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

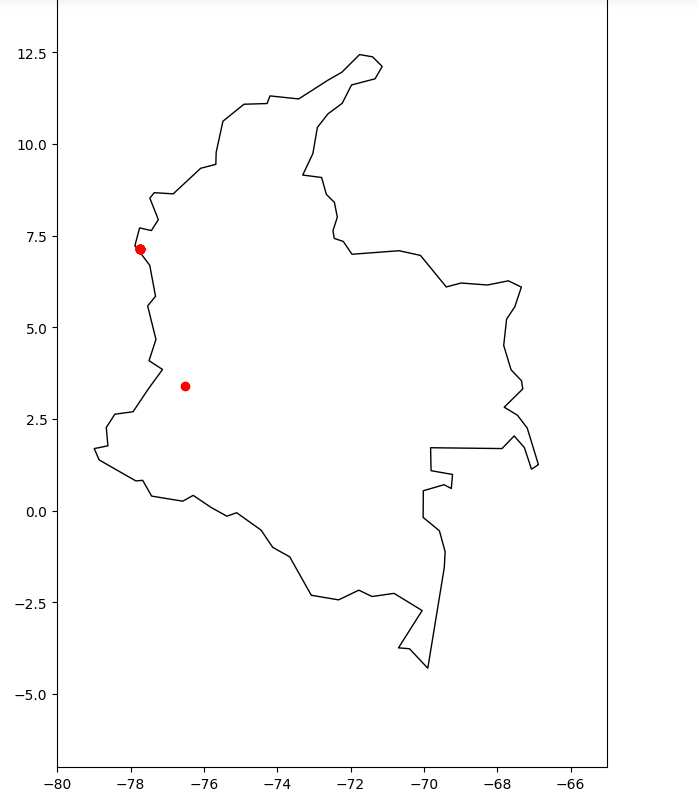
Descripción generada automáticamente

Que entrega la distancia total en km entre dos latitudes y longitudes para dos personas en la tabla Personas.

**Shapes y despliegue de los datos geoespaciales.**

Utilizando Python junto a Geopandas y un shape obtenido de internet, tenemos que el despliegue de los datos sería algo así:





Algo importante a destacar en este punto es que no se toman los valores de la base de datos automáticamente, si no, que se toman de forma manual. Sin embargo, junto con el script de Python que despliega toda la información, se añadirá también el script para conectarse a la base de datos.