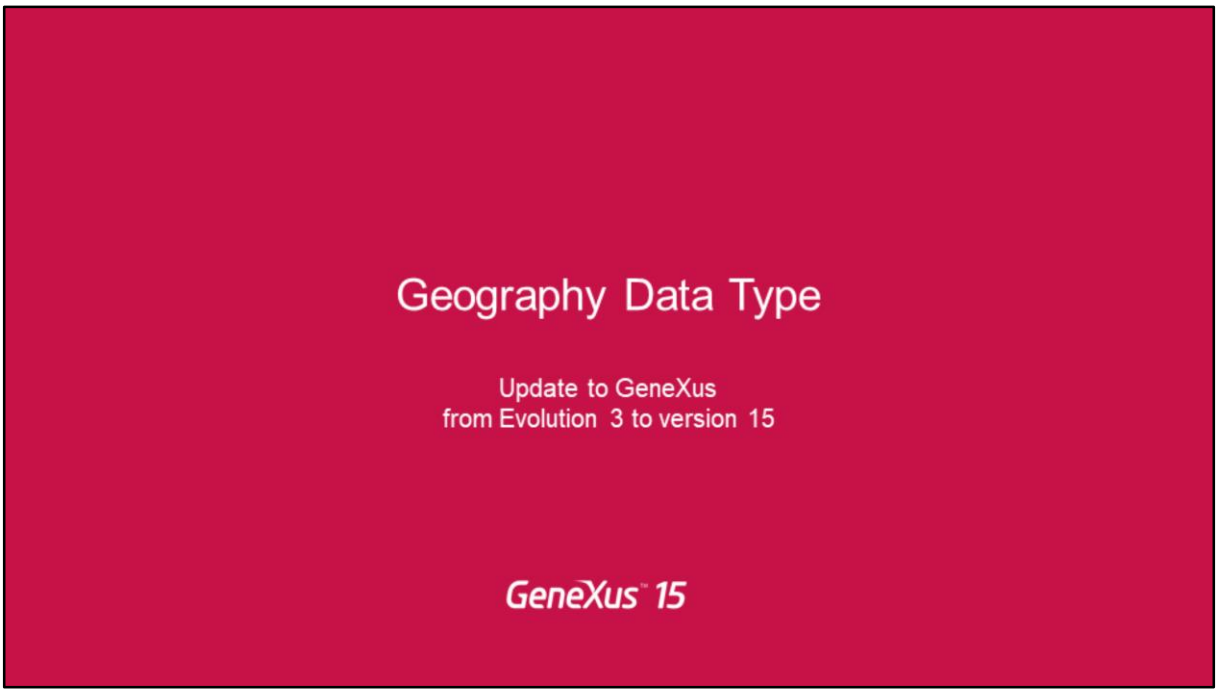
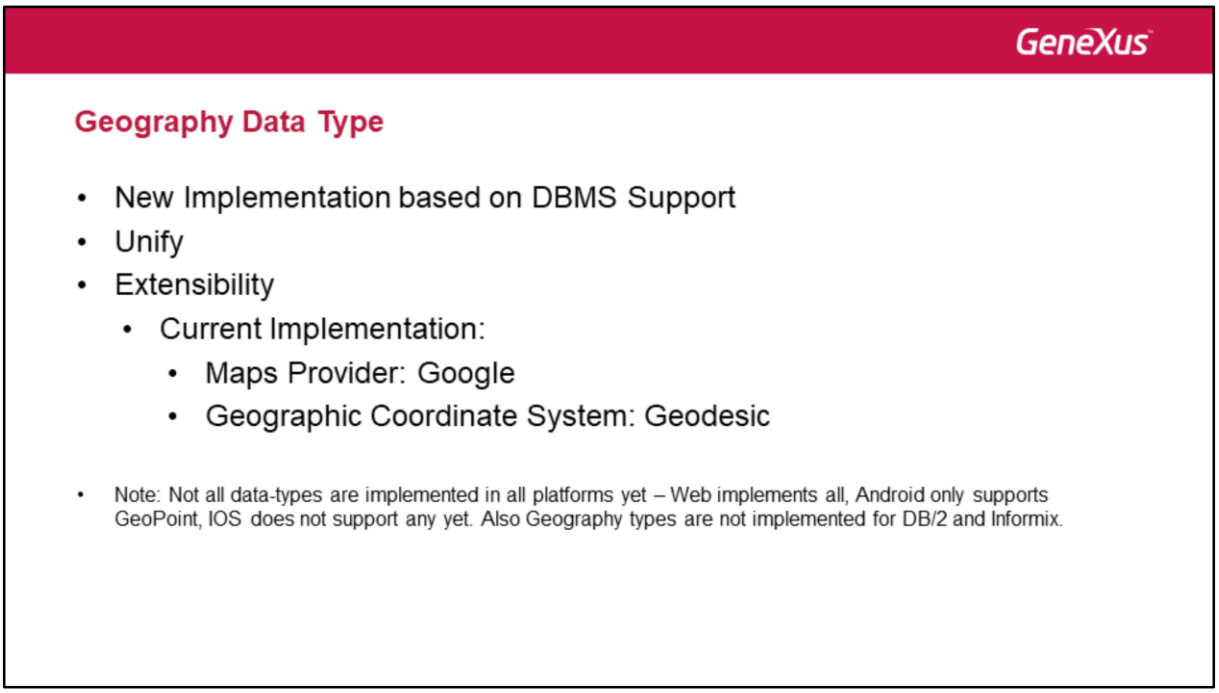


Podemos representar entidades geográficas utilizando el tipo de datos Geography.

Esto implica que la información geográfica se almacena a nivel de la base de datos utilizando tipos de datos geográficos nativos de cada DBMS.



El nuevo tipo de datos geográfico no brinda nuevas funcionalidades aún, es una re implementación de cómo GeneXus maneja la información geográfica, construyendo sobre la funcionalidades brindadas por los DBMS en vez de implementarlo a nivel de GeneXus.

Esta será la base sobre la cual GeneXus podrá proveer más funcionalidad relacionada con información geográfica de forma más rápida y mejor, y también más consistente entre web y SD. Uno de los objetivos de esta reingeniería fue proveer más consistencia entre SD y Web.

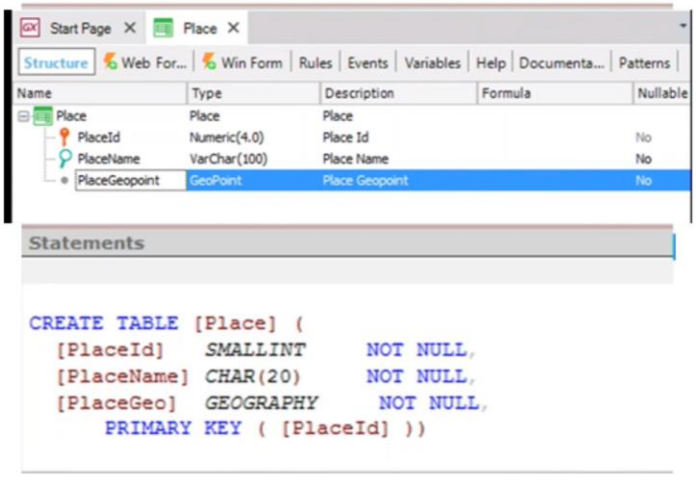
La solución va a permitir mayor extensibilidad – abriendo la puerta a nuevos proveedores de mapas (actualmente solo trabajamos con Google) y nuevos sistemas de coordenadas (actualmente solo se manejan coordenadas geodésicas).

GeneXus

Data Types

Geography Data Type:

- GeoPoint
- GeoLine
- GeoPolygon



Del tipo de datos geográfico se derivan : GeoPoint, GeoLine y GeoPolygon. Estas son especializaciones del tipo de dato Geography .

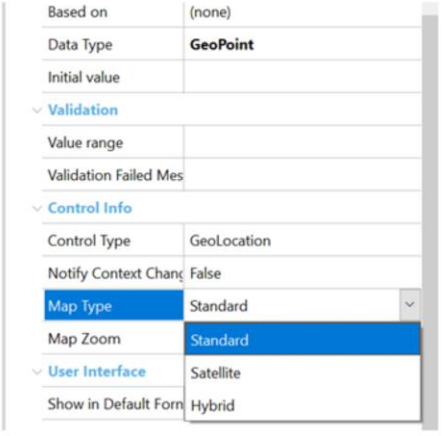
A nivel de la base de datos, todas las especializaciones (GeoPoint, Geoline y GeoPolygon) se almacenan de la misma forma – salvo en MYSQL que tiene soporte specific para los diferentes tipos. Pero a nivel de GeneXus se manejan de forma separada para capturar la semántica de los atributos/variables.

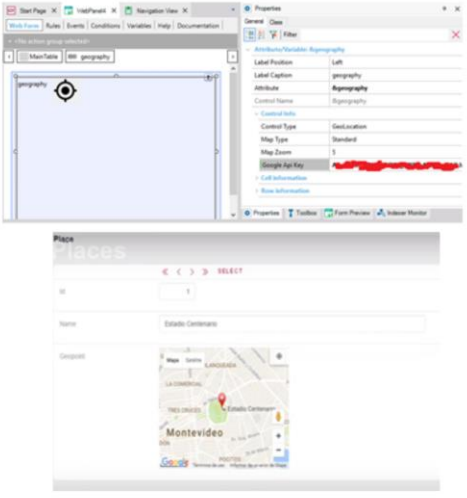
GeoPoint se utiliza para capturar un punto (latitud/longitud) como puede ser la ubicación de un edificio o una atracción turística
GeoPolygon captura un área, como puede ser una manzana, barrio, etc.
GeoLine captura una línea, que se utiliza para representar calles, recorridos, etc.

Al definir un atributo basado en un tipo de datos geográfico – como GeoPoint – vamos a ver en el análisis de impacto que se usa el tipo de dato Geography (a diferencia de cuando usamos el dominio GeoLocation que se almacena como un string en la base de datos, y si miramos la base de datos nada nos dice que allí se almacena latitud/longitud – el significado está defindo a nivel de GeneXus pero no del DBMS. En el caso de los tipos de datos geográficos esta información llega al DBMS)

GeneXus

Attribute Definition & Visualization

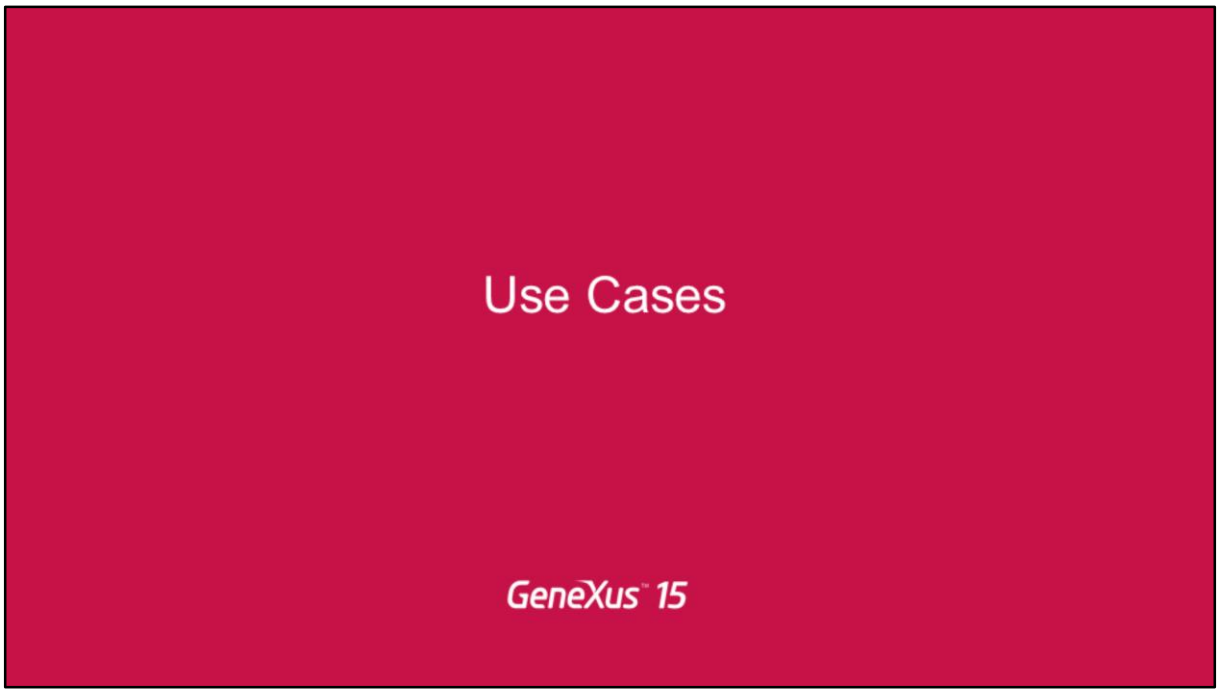




Cuando definimos un atributo basado en un tipo de dato geográfico, podemos elegir cómo se visualiza

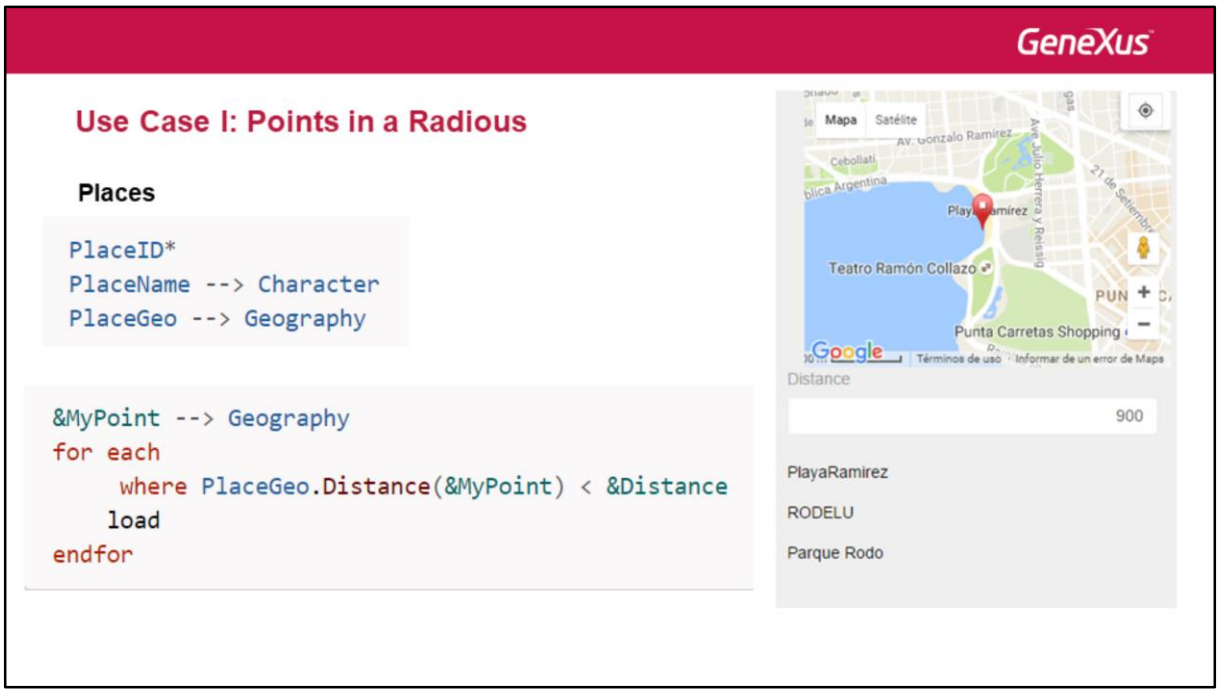
Al incluir en un web panel o panel SD un atributo de este tipo, necesitamos ingresar la API Key de Google – por ahora es el único proveedor –

Instrucciones de cómo obtener la API Key y más información: https://wiki.genexus.com/commwiki/servlet/wiki?32408,Geography%20data%20type)



¿Qué casos de uso podemos resolver con el nuevo tipo de datos geográfico?

Ninguno de los casos de uso son nuevos – todos tenían ya una solución en versiones previas de GeneXus utilizando el dominio GeoLocation – lo que es nuevo es la implementación de la solución y el hecho de que las consultas se resuelven a nivel del DBMS – lo cual es igual o más performante. La nueva implementación permitirá resolver más casos de uso en menores tiempos y con mayor potencia a medida que se usa toda la potencia de los DBMS.



El primer escenario es encontrar los puntos cercanos dado un punto definido y un radio.

Para esto, el tipo de dato GeoPoint implementa un método “Distance”. Esto devuelve todos los puntos que tenemos definidos en la tabla Places que están a cierta distancia definida por parámetro (en el ejemplo 900 m) desde un cierto punto (&MyPoint). Si quiero el más cercano me quedaría con el que tiene la menor distancia

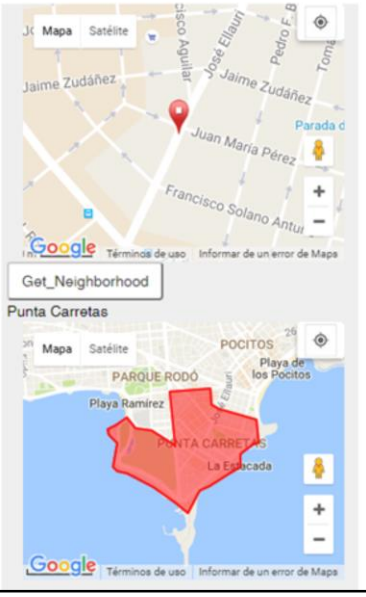
Antes podíamos implementar esto utilizando &GeoLocation.GetDistance (&MyLocation, &SomeLocation) donde tanto &MyLocation como &SomeLocation eran variables de tipo GeoLocation. La nueva implementación es mas limpia, pero la mayor ventaja es que la consulta es resuelta a nivel del DBMS y no por un algoritmo de GeneXus.

GeneXus

Use Case II: Point in Polygon

```
NeighborhoodId*
NeighborhoodName --> Character
NeighborhoodPlace --> Geography
```

```
For each
      Where NeighborhoodPlace.Intersect(&Mypoint)
      &NeighborhoodPlace = NeighborhoodPlace
endfor
```



En este caso queremos saber si un punto dado se encuentra en un barrio – para resolver esta pregunta necesitamos intersectar dos estructuras: el punto y el polígono que representa el barrio.

De forma similar al caso anterior, era posible realizar esto en versiones previas de GeneXus usando el dominio GeoLocation – nuevamente lo que es nuevo es la implementación ya que la consulta se resuelve en el DBMS.

GeneXus

Data Input

```
New
   PlaceId = 1
   PlaceName = "Golf Club"
   PlaceGeo.FromString("POINT(-56.163740158081055 -34.92478600243492)")
endNew

New
   PlaceId = 2
   PlaceName = "Ramirez Beach"
   PlaceGeo.FromGeoJson('{ "type": "Point", "coordinates":  -56.1701774597168 -34.91676309400329 }')
endnew
```

Si queremos asignar valores a atributos de tipo geográfico en pantalla lo haríamos de forma interactiva gracias al control “Mapa”.
Programáticamente, también podemos interactuar con datos geográficos utilizando FromString and FromGeoJson.
FromGeoJson recibe un archivo/texto en formato GeoJSON, mientras que string recibe un texto plano.

Geolocation Improvements

GeneXus 15

GeoLocation improvements

- Android:
  - Background Tracking
  - Proximity Alert
- IOS:
  - Proximity Alert improved precision

GeneXus™

Videos	training.genexus.com
Documentation	wiki.genexus.com
Certificactions	training.genexus.com/certifications