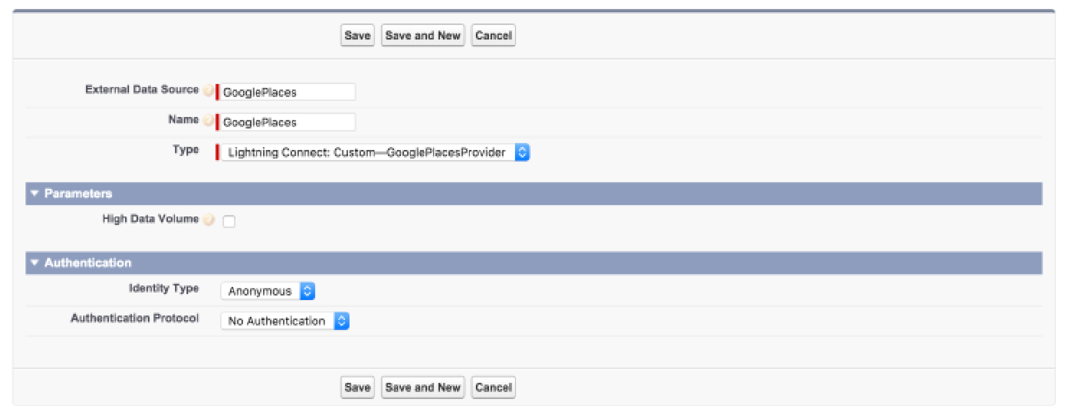
High Data Volume External DataSources

Lightning Connect is a Salesforce feature that allows integrations with external systems. Before Winter 16 release, it only read-only capabilities were provided. Since Winter 16, writable datasources were delivered, making Lightning Connect a very interesting option to think about.

In order to work with Lightning Connect you have to declare an **External Datasource**, and a way to access this external datasource. This is, if your datasource understands the OData protocol, Salesforce provides **OData adapters**. In addition, Salesforce gives us the capability of writing a **custom adapter in Apex** for reading any kind of datasources. You can take a look at this interesting post about how to create a custom adapter.

Playing with External Datasources, I found an interesting option that you can use, which is declaring it as a "**High Data Volume External Datasource**". This option is available with OData adapters as well as with custom adapters:



External Data Source: GooglePlaces
Name: GooglePlaces
Type: Lightning Connect: Custom—GooglePlacesProvider

Parameters
High Data Volume: ☐

Authentication
Identity Type: Anonymous
Authentication Protocol: No Authentication

What does this option mean? Well, as everything in Salesforce, accessing external objects through external datasources has **limits**. The general external data integration limits can be seen here.

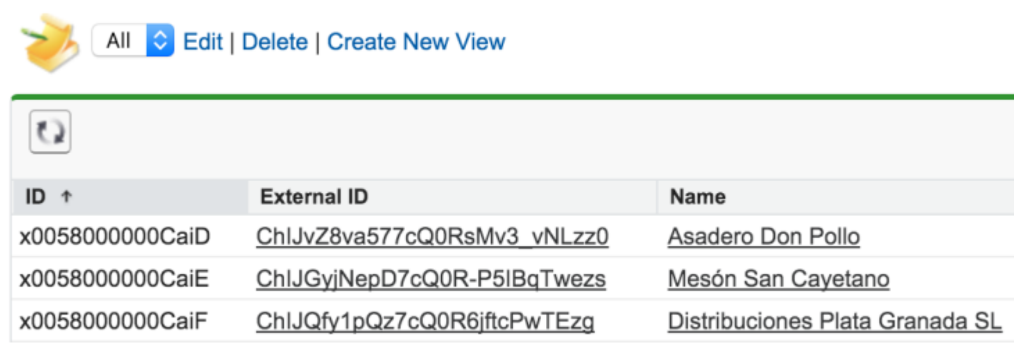
However, if your external datasource is hitting rate limits, you have the option of transforming it into a **High-Volume** one.

The documentation says that doing so bypasses most of the rate limits. However, external objects that are associated with high-data-volume external data sources aren't available in **Salesforce1** and don't support **record feeds**.

In addition, doing some tests with High-Volume datasources I have found some interesting facts:

Using a **Non High-Volume Datasource**, I noticed that Salesforce **assigns an Id over the scenes** to each one of the external objects you access in the system (I guess it stores an equivalence between the real ExternalIds of the objects and the "virtual" Ids it had assigned to them somehow).

This is, each time a new external object is retrieved, Salesforce will assign an Id for each external object, as we can see in the following listview:



ID ↑	External ID	Name
x0058000000CaiD	ChIJvZ8va577cQ0RsMv3_vNLzz0	Asadero Don Pollo
x0058000000CaiE	ChIJGyjNepD7cQ0R-P5IBqTwezs	Mesón San Cayetano
x0058000000CaiF	ChIJQfy1pQz7cQ0R6jftcPwTEzg	Distribuciones Plata Granada SL

Knowing a record's Id, then, **we can then perform SOQL queries based on that Id**, even if we don't know its ExternalId:

```
[SELECT Id FROM Restaurant__x WHERE Id = 'x0058000000CaiD']
```

Salesforce will automatically **translate this query to a query which filters by the related ExternalId**, which is the query that will arrive in the end to your custom adapter implementation (if you are using a custom one, which is my case):

```
[SELECT Id FROM Restaurant__x WHERE ExternalId = 'ChIJvZ8va577cQ0RsMv3_vNLzz0']
```

However, if the datasource is a **High-Volume one**, **no Ids are set** and the Id field is not available for being added to the list view! So the only possible queries we could do will be **ExternalId based queries**.

This fact has implications when developing a pair of **Visualforce pages** for having a **customized list – detail navigation**.

In the first case (**Non High-Volume datasource**), we are able to add **Id-based detail navigation links** to the list page, and query the detail page information through the object Id, as Salesforce is able to translate this "virtual " Id to an ExternalId field. For example having this link:

```
<a href="{!URLFOR($Page.restaurantdetail,'',[id=restaurant.Id])}">
    <apex:outputField value="{!restaurant.Name__c}"/>
</a>
```

In the **controller of your detail page** you can query your object's information **using the Id**:

```
public RestaurantController(ApexPages.StandardController standardController)
{
        Restaurant__x restaurant = (Restaurant__x) standardController.getRecord();

        restaurant  = [SELECT Location__c FROM Restaurant__x WHERE Id := restaurant.Id];
}
```

However, in the second case (**High-Volume datasource**), if we have the same implementation, all links will point to Id “ `000000000000000AAA` “, which is an standard **empty key Id** from Salesforce.

Then, in that case, one option is to create your **detail links based on ExternalIds**, and **parse the ExternalId parameter in your detail page controller** for querying your detail page information through the received **ExternalId**.

```
<a href="{!URLFOR($Page.restaurantdetail,'',[ExternalId=restaurant.ExternalId])}">
        <apex:outputField value="{!restaurant.Name__c}"/>
</a>
```

```
public RestaurantController(ApexPages.StandardController standardController)
{
        String externalId = ApexPages.currentPage().getParameters().get('ExternalId');

        Restaurant__x restaurant  = [SELECT Location__c FROM Restaurant__x WHERE ExternalId := externalId];
}
```

Indeed, if we retrieve the **metadata** that defines an External Datasource, we find that setting a datasource as a High-Volume one at metadata level is done adding the next **property**:

```
<customConfiguration>{'noIdMapping':'true';}</customConfiguration>
```

To finish I just want to add that according to the documentation, High-data-volume external data sources are still limited to **10,000 OData queries per hour** for Enterprise, Performance, and Unlimited Editions for OData adapters, but higher limits are available on request.