Article
Benjamin De Boe · Aug 4, 2016 7m read

Creating a domain with the iKnow Domain Architect

In previous articles on iKnow, we described a number of demo applications (iKnow demo apps parts 1, 2, 3, 4 & 5) that are either part of the regular kit or can be easily installed from GitHub. All of those applications assumed you already had your iKnow domain ready, with your data of interest loaded and ready for exploration. In this article, we'll shed more light on how exactly you can get to that stage: how you define and then build a domain.

Background

What is a domain?

OK, we've been using the term "domain" a lot, but what exactly is a domain? A **domain** is a repository for the concepts and their context iKnow produces while indexing one or more bodies of text, made available for querying through the COS or REST APIs. For an introduction into what these iKnow concepts are and how iKnow's bottom-up approach to obtain them works, please refer to this video.

So, why not store these concepts in a simple table? Because the results iKnow produces are not just flat, one-dimensional records, we need a more complex structure, which can be exposed through a combination of tables referring to one another. While you can access these tables directly, we provide the APIs on top to help you answer typical questions more easily, rather than having to write lengthy SQL queries. More on this topic in this video.

Then, what goes into a domain and when would I need more than one? Typically, anything you may want to compare to one another would go into a single domain. For example, in a customer service environment, you may want to put all service request texts from the main system (let's take the WRC as an example) and published advisory articles into the same domain because they're about the same subject, and the service quality feedback in another as they have a different type of content you'd probably never need to compare it to the technical texts in the first domain. In most cases, your iKnow domain will correspond to the data in a single column in a single table.

Domain Definitions

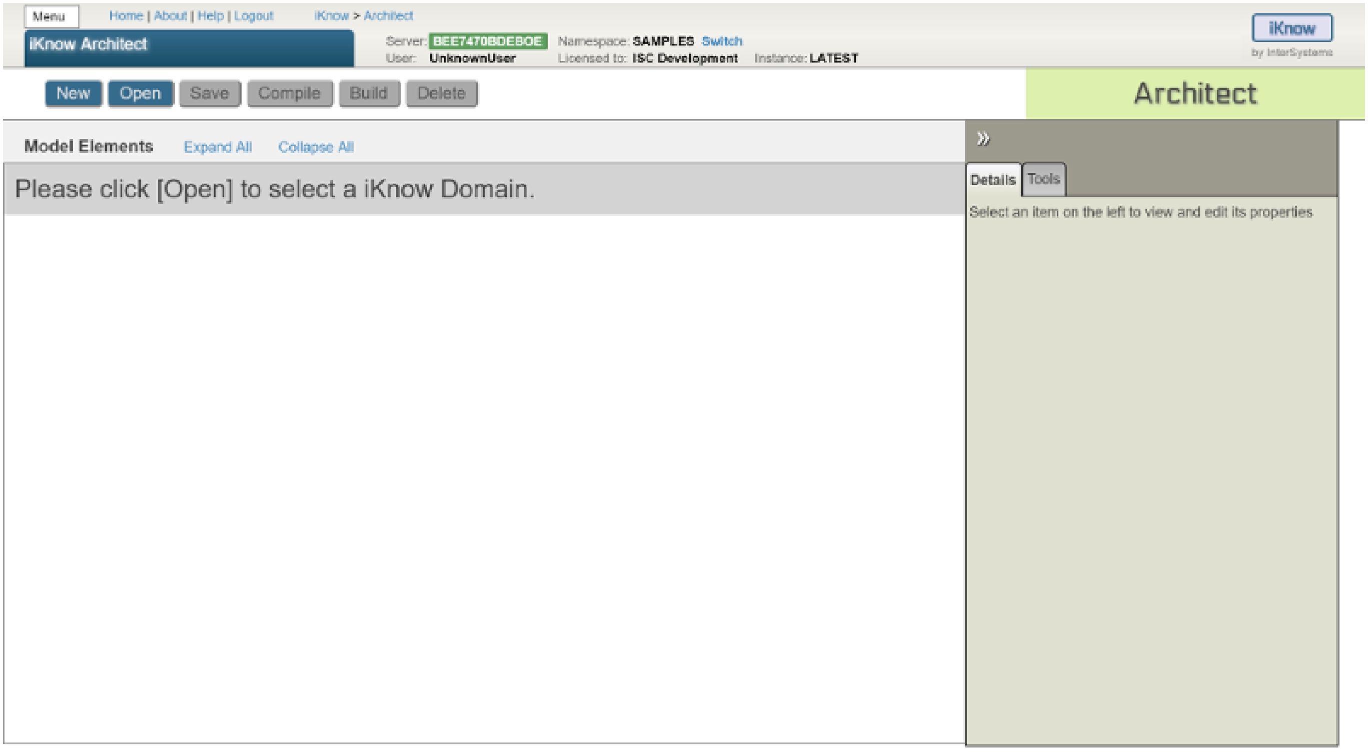
Since 2014.1, you can *define* how your domain should be configured and what data needs to go in, rather than having to *program* all of that in code. This is very similar how you *define* how a DeepSee cube should look like, rather than *program* its dimensions and where to source the data from. Domain Definitions are the recommended way of working with iKnow domains, as they make it easy to manage and load data in a repeatable fashion.

Domain Definitions appear as Caché classes inheriting from %iKnow.DomainDefinition with an XData block that specifies the properties of your domain and data locations to load data from in XML. You can learn more about creating and modifying Domain Definition classes in this video, and a couple more advanced topics related to loading in this one. In the next part of this article, we'll dig deeper into creating and managing these classes through a GUI: the **iKnow Domain Architect**, which is available as from Caché 2016.1.

The iKnow Domain Architect

Learning by example

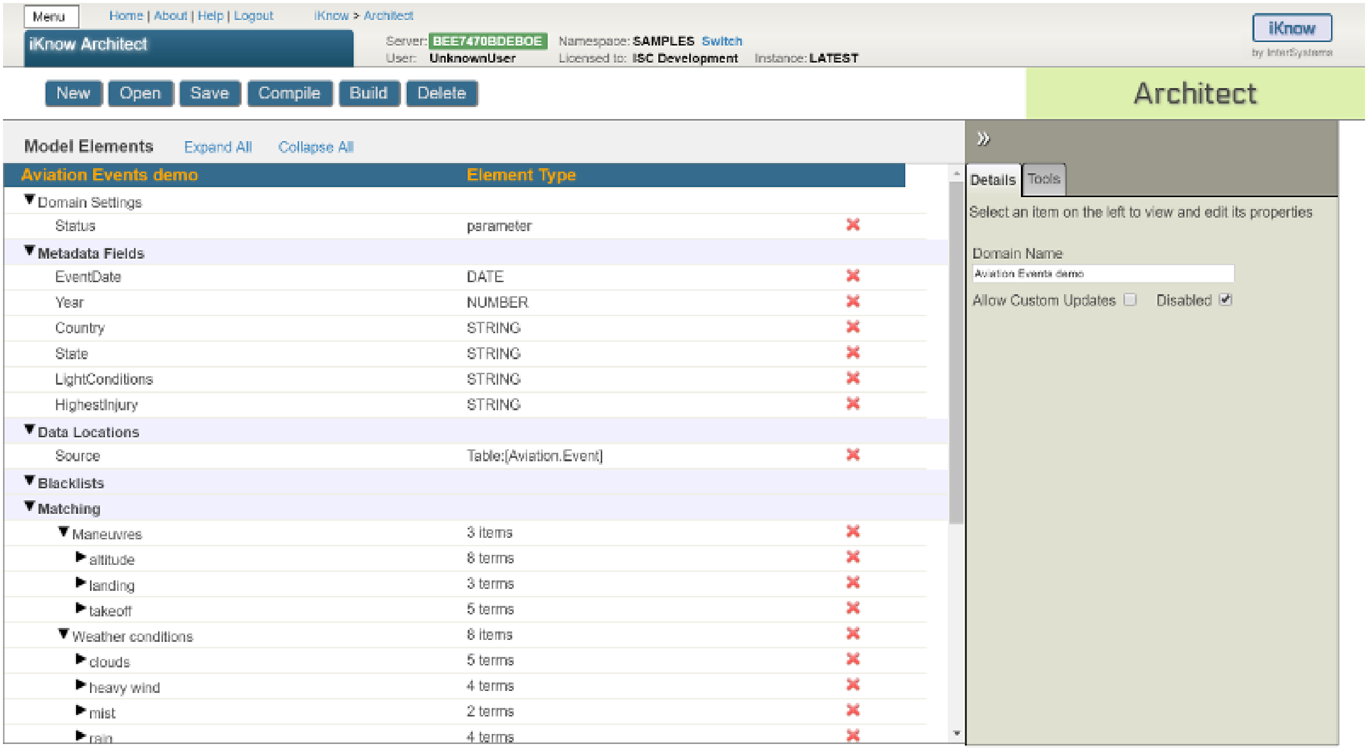
To access the iKnow Domain Architect (or simply iKnow Architect), open the System Management Portal, choose "System Explorer" > "iKnow" > "Domain Architect", after switching to a namespace that supports iKnow. If your namespace is not in the list, you can enable it for the corresponding web application through "System Administration" > "Security" > "Applications" > "Web Applications" and then checking the Enable iKnow checkbox first. Upon opening the Architect, you should see the following screen.



Please click [Open] to select a iKnow Domain.

Select an item on the left to view and edit its properties

Let's first access an existing domain definition to get a sense of how they look. The SAMPLES namespace contains one for the Aviation Event demo (Aviation.ReportDomain), and another one for the NewsReader tutorial (News.NewsDomain). Let's open the first one, clicking the "open" button and navigating to Aviation > ReportDomain.



Aviation Events demo	Element Type	
▼Domain Settings		
Status	parameter	✖
▼Metadata Fields		
EventDate	DATE	✖
Year	NUMBER	✖
Country	STRING	✖
State	STRING	✖
LightConditions	STRING	✖
HighestInjury	STRING	✖
▼Data Locations		
Source	Table:[Aviation.Event]	✖
▼Blacklists		
▼Matching		
▼Maneuvres	3 items	✖
▶altitude	8 terms	✖
▶landing	3 terms	✖
▶takeoff	5 terms	✖
▼Weather conditions	8 items	✖
▶clouds	5 terms	✖
▶heavy wind	4 terms	✖
▶mist	2 terms	✖
▶rain	4 terms	✖

Select an item on the left to view and edit its properties

Domain Name: Aviation Events demo

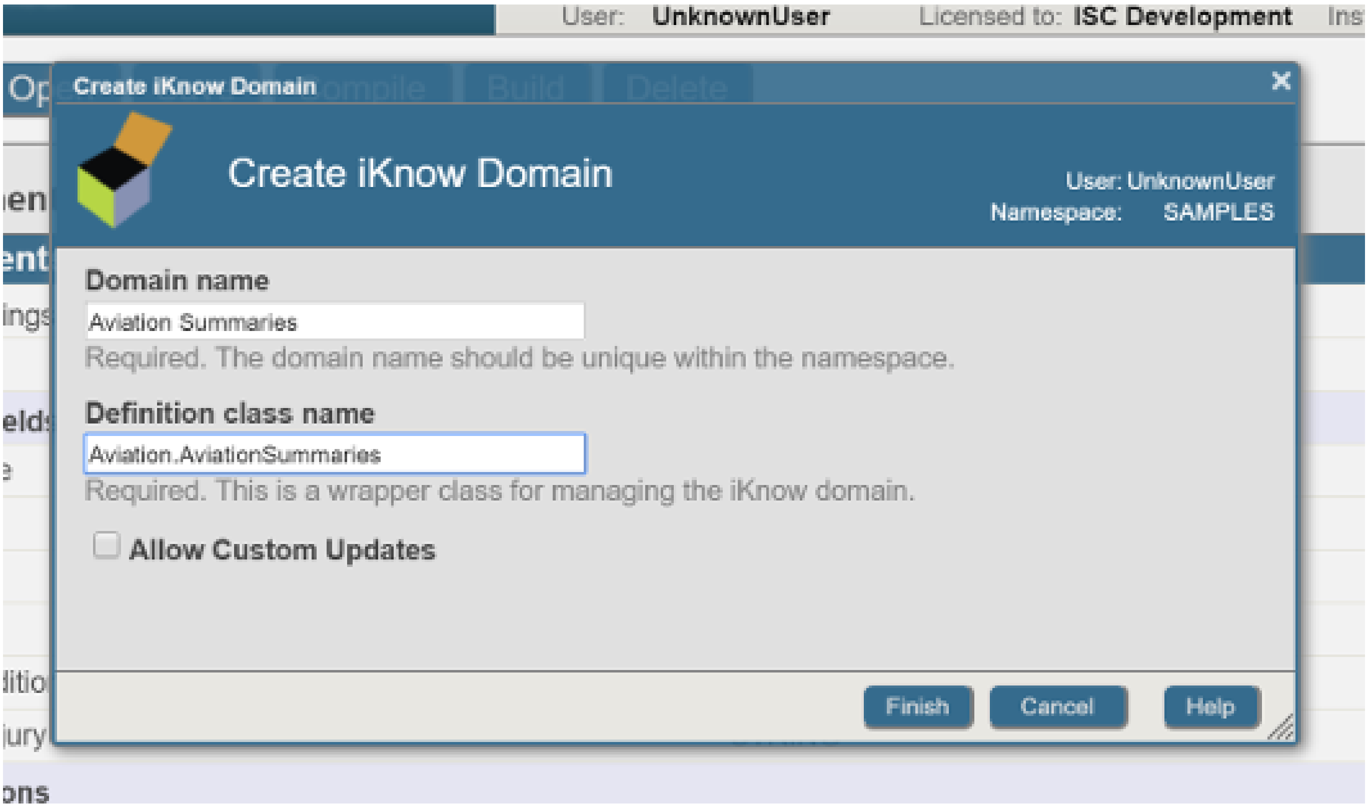
Allow Custom Updates ☐ Disabled ☑

When you click any of the model elements on the left, you can see and change its properties in the detail panel on the right. Especially the Data Locations section is probably the most interesting one. As this is a showcase for many Domain Definition features, the screen may look a little dense at first, but we'll build a simple one from scratch in the next section.

Note that both the Aviation Events and NewsReader domains are disabled by default, as iKnow requires an appropriate license, so we cannot package them enabled. Click the domain name in the blue bar and then uncheck the "disabled" checkbox in the detail panel on the right before compiling to really get started. For the Aviation demo, you can also refer to the SetupStandalone() method in the Aviation.Utils class.

Creating your first domain definition

Now, let's create our first domain definition from scratch. Click the "new" button and choose a name for your domain, as well as a valid class name to store the definition in.

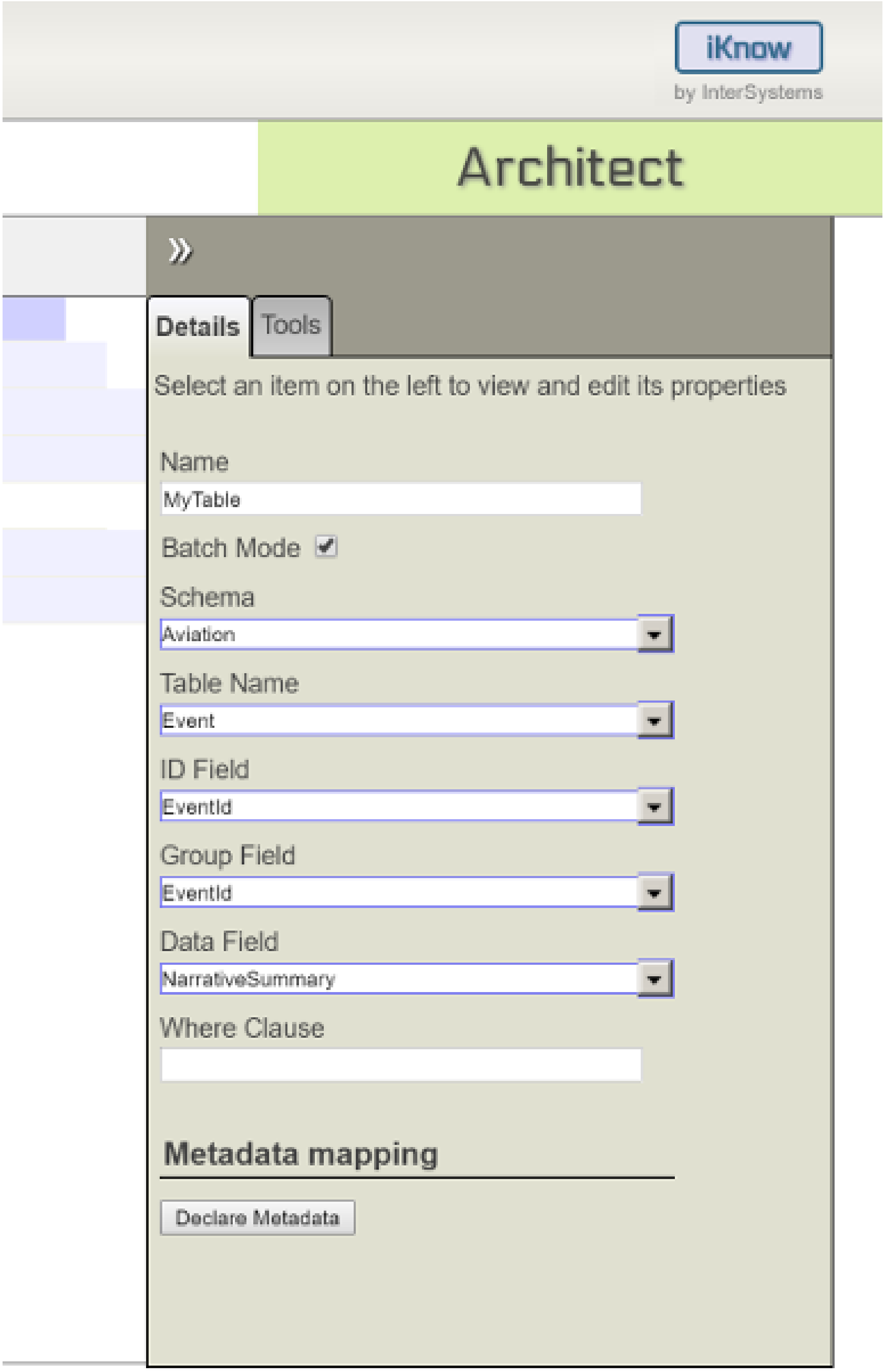


Domain name
Aviation Summaries
Required. The domain name should be unique within the namespace.

Definition class name
Aviation.AviationSummaries
Required. This is a wrapper class for managing the iKnow domain.

□ Allow Custom Updates

Now click "Finish" and you'll see an empty domain definition. Let's add a data location to source data from by first selecting the "Data Locations" bar on the left and then clicking the "Add data from table" button on the right. This will add a row to the Data Locations section that represents a single location iKnow will source data from when you "build" the domain. Let's populate our domain with the summary reports in the Aviation Events table. Don't worry too much about the ID field and group field columns. They are meant to combine into a unique identifier for every record (aka "source") in an iKnow domain that makes some sense to the outside world. Any combination of columns that will be unique for your domain will do. In the end, your details pane should look like this:



Select an item on the left to view and edit its properties

Name: MyTable

Batch Mode ☑

Schema: Aviation

Table Name: Event

ID Field: EventId

Group Field: EventId

Data Field: NarrativeSummary

Where Clause:

Metadata mapping

Compiling and Building

Now that we're done defining our first domain, it's about time to make it real. There are three different stages to this:

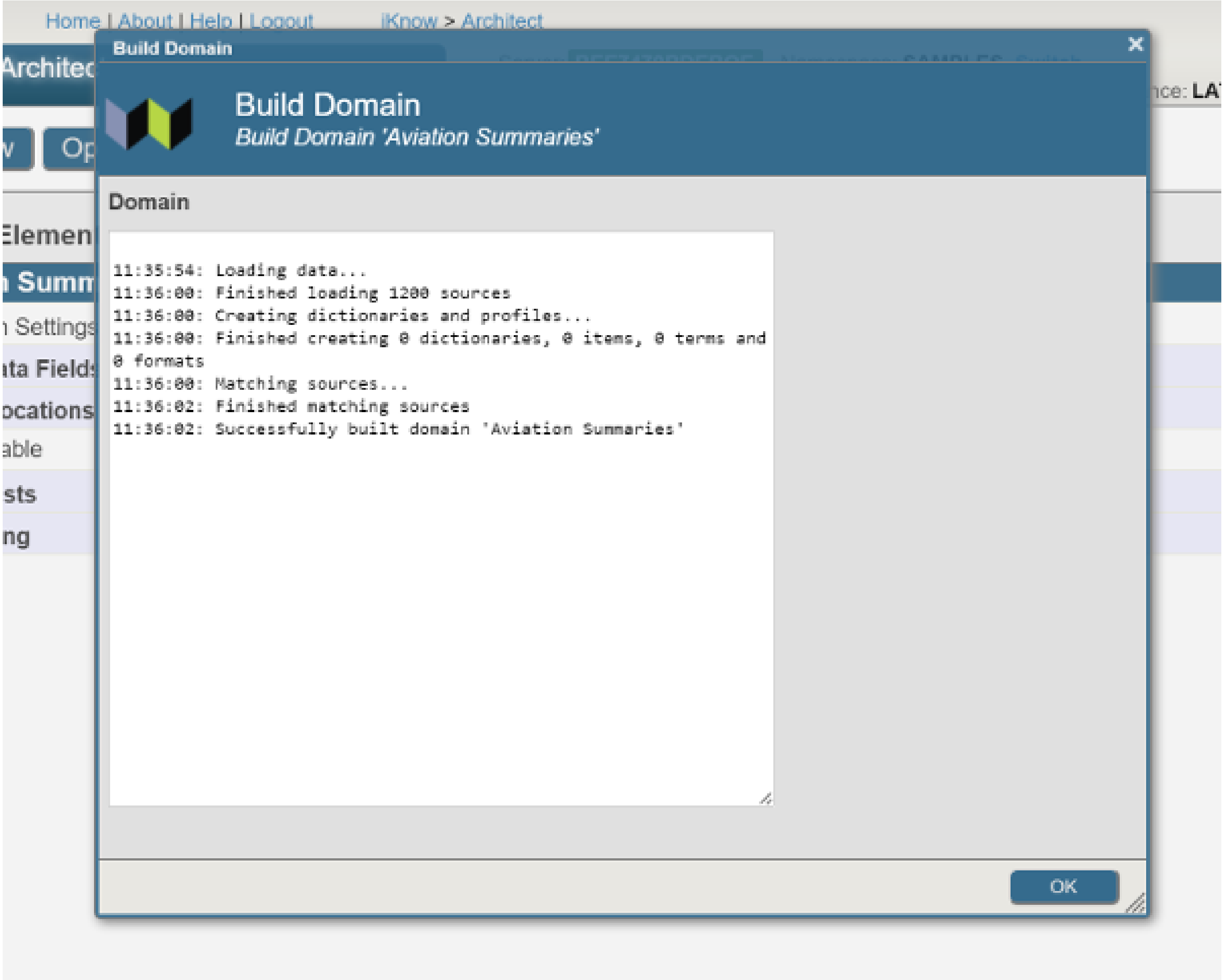
- When pressing the **Save** button, your definition will be saved to a class. No validation is performed.
- When pressing the **Compile** button, your definition class will be compiled. This will validate all the settings

and locations you defined for your domain and automatically generate any required code to load the domain. In addition, any static elements, such as the domain, configuration objects and metadata fields, will be created in the background. Compiling a domain definition implicitly saves it beforehand.

- When pressing the **Build** button, the code that was generated when the definition was compiled is invoked to populate the domain, loading the data from the defined data locations and indexing them with iKnow. You can only build a domain definition after it has been compiled.

Underneath the GUI, the first two actions are no different from how you'd interact with regular classes in Studio or the command line. Building a domain, on the other hand, comes down to executing the %Build() method from the %iKnow.DomainDefinition class.

Now go ahead and compile your domain. You can then open it in Studio or Atelier if you'd like to see how the definition XData block looks like. Back in Architect, click the "Build" button. You'll see a popup with basic output about what's happening that should look more or less like this at the end:

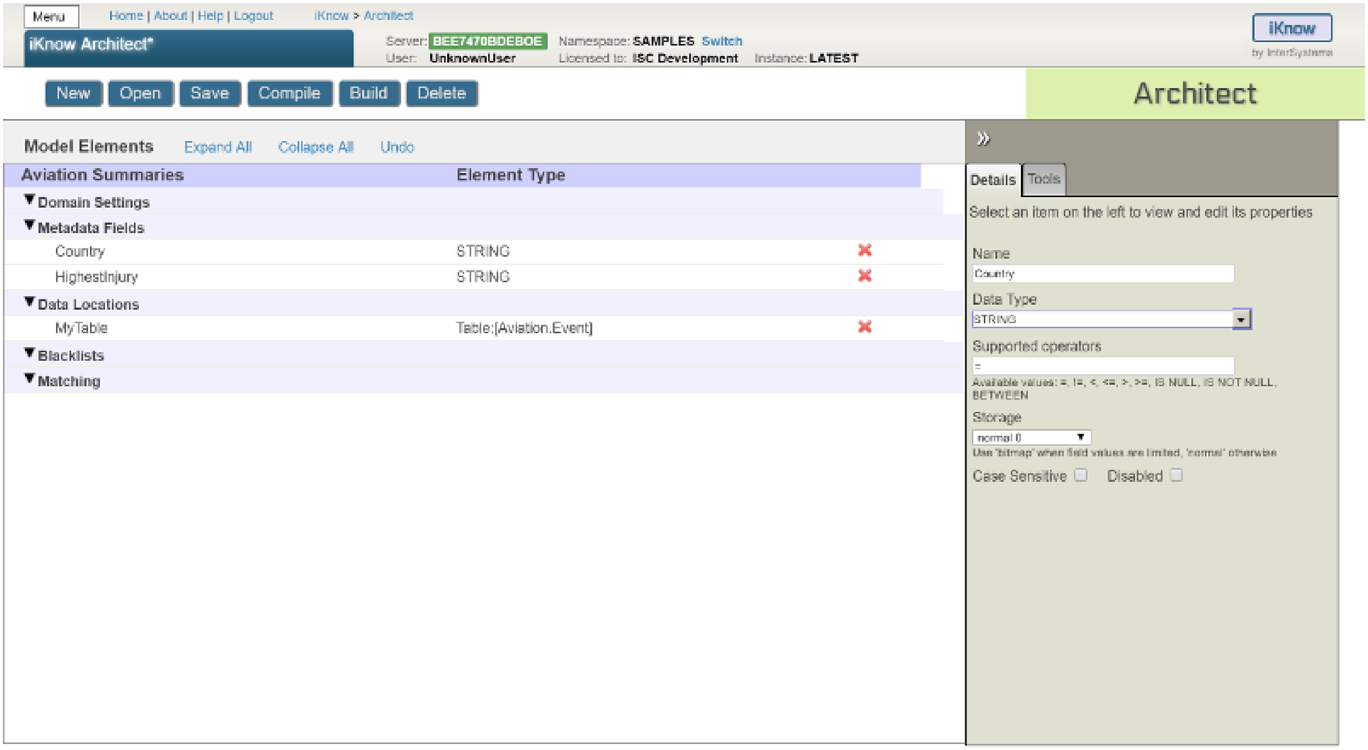

Home | About | Help | Logout
iKnow > Architect
Build Domain
Build Domain
Build Domain 'Aviation Summaries'
Domain
OK

```
11:35:54: Loading data...
11:36:00: Finished loading 1200 sources
11:36:00: Creating dictionaries and profiles...
11:36:00: Finished creating 0 dictionaries, 0 items, 0 terms and
0 formats
11:36:00: Matching sources...
11:36:02: Finished matching sources
11:36:02: Successfully built domain 'Aviation Summaries'
```

Now, after closing this window, when you click the "Tools" tab in the details panel, you can click through to the Knowledge Portal to browse the indexing results for the aviation event summaries column. Check out this article if you want to know more about the Knowledge Portal.

Adding metadata

As you may have seen going through the Knowledge Portal introduction, it's often convenient to have a couple of metadata fields available to organize your data and filter against. To add these in the iKnow Architect, click the "Metadata Fields" bar on the left and then the "Add Metadata" button on the right. Use this button and the corresponding details section to add a couple of metadata fields to your domain. You'll want to choose fields that make sense for filtering your queries that you have data to populate for. When selecting the storage type, take bitmap for fields that will only have a limited number of possible values (like country) or normal otherwise.



Aviation Summaries	Element Type	
▼Domain Settings		
▼Metadata Fields		
Country	STRING	✖
HighestInjury	STRING	✖
▼Data Locations		
MyTable	Table:[Aviation.Event]	✖
▼Blacklists		
▼Matching		

Select an item on the left to view and edit its properties

Name: Country

Data Type: STRING

Supported operators: =

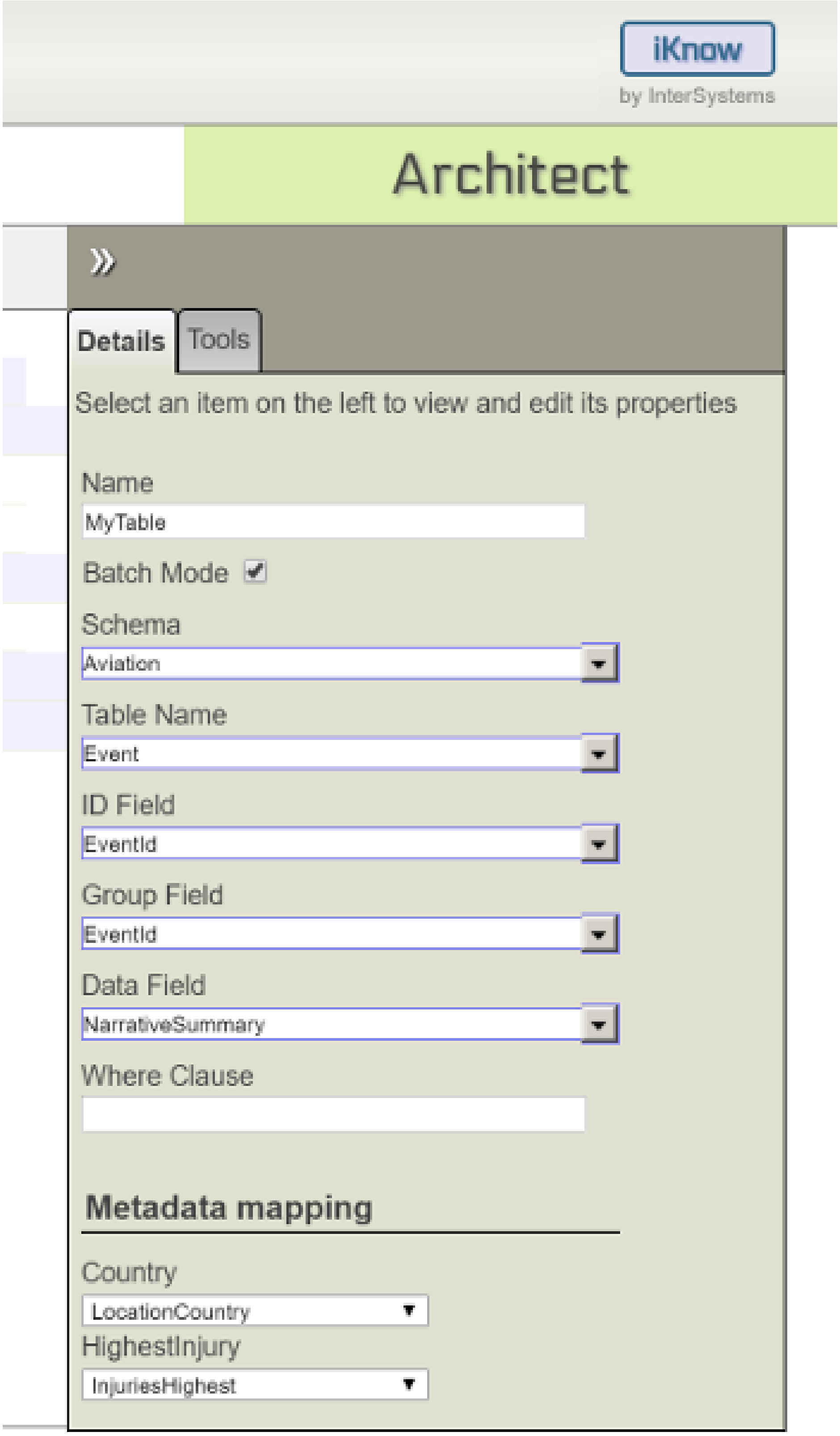
Available values: =, !=, <, <=, >, >=, IS NULL, IS NOT NULL, BETWEEN

Storage: normal 0

Use 'bitmap' when field values are limited, 'normal' otherwise

Case Sensitive □ Disabled □

Now, go back to your data location entry. You'll now see a "Metadata mappings" section at the bottom that allows you to select which columns to load data from for populating the fields:



Select an item on the left to view and edit its properties

Name: MyTable

Batch Mode ☑

Schema: Aviation

Table Name: Event

ID Field: EventId

Group Field: EventId

Data Field: NarrativeSummary

Where Clause:

Metadata mapping

Country: LocationCountry

HighestInjury: InjuriesHighest

After mapping these fields, recompile and rebuild your domain in order to be able to take advantage of them in the Knowledge Portal.

What's next?

This article introduced the primary capabilities of the iKnow Architect for managing and building iKnow domains. There's more options and settings that can be controlled through the interface, which you can learn more about in the documentation, or just by lots of trial and hopefully little error!

If you want to go beyond the mere exploration offered through the Knowledge Portal, you can take a look at the Set Analysis, Dictionary Builder and Rules Builder demo apps as a source of inspiration to put the concepts and context identified by iKnow to good use.

#Best Practices #InterSystems Natural Language Processing (NLP, iKnow)

Source URL:https://community.intersystems.com/post/creating-domain-iknow-domain-architect