Chapter 1 : Extend system partition, Extend C drive - EaseUS Partition Master Guide

To do this we need to first look at the second way to extend the Common Data Model using existing entities. Extending the Model with Existing Entities Before we can complete our previous Relationship example we will need to extend our new Application by adding the existing entity of Contact to it.

Project Open Data Metadata Schema v1. Federal CFO-Act agencies are expected to complete the transition to the v1. To see changes from v1. Updates to the metadata schema can be found in the changelog. The challenge is to define and name standard metadata fields so that a data consumer has sufficient information to process and understand the described data. The more information that can be conveyed in a standardized regular format, the more valuable data becomes. Metadata can range from basic to advanced, from allowing one to discover the mere fact that a certain data asset exists and is about a general subject all the way to providing detailed information documenting the structure, processing history, quality, relationships, and other properties of a dataset. Making metadata machine readable greatly increases its utility, but requires more detailed standardization, defining not only field names, but also how information is encoded in the metadata fields. Establishing a common vocabulary is the key to communication. The metadata schema specified in this memorandum is based on DCAT , a hierarchical vocabulary specific to datasets. This specification defines three types of metadata elements: Required, Required-if conditionally required , and Expanded fields. These elements were selected to represent information that is most often looked for on the web. To assist users of other metadata standards, field mappings to equivalent elements in other standards are provided. What to Document â€" Datasets and Web APIs A dataset is an identifiable collection of structured data objects unified by some criteria authorship, subject, scope, spatial or temporal extentâ€¦. A catalog is a collection of descriptions of datasets; each description is a metadata record. The intention of a data catalog is to facilitate data access by users who are first interested in a particular kind of data, and upon finding a fit-for-purpose dataset, will next want to know how to get the data. For example, a dataset of farmers markets may be made available for download as a single file e. Please also see the extended guidance on documenting Web APIs in your data. A quick primer on the file format involved: JSON is a lightweight data-exchange format that is very easy to read, parse and generate. Based on a subset of the JavaScript programming language, JSON is a text format that is optimized for data interchange. JSON is built on two structures: Where optional fields are included in a catalog file but are unpopulated, they may be represented by a null value. They should not be represented by an empty string "". Any object may be described by title, description, format, or mediaType, though when an object contains downloadURL, it must be accompanied by mediaType. The Project Open Data schema is case sensitive. The schema uses a camel case convention where the first letter of some words within a field are capitalized usually all words but the first one. While it may seem subtle which characters are uppercase and lowercase, it is necessary to follow the exact same casing as defined in the schema documented here. Tools to help agencies produce and maintain their data inventories are available on GitHub and hosted at Labs. Publishers can also use the describedBy field to reference the default JSON Schema file used to define the schema https:

Chapter 2 : Tutorial: Extend Data Model relationships using Excel, Power Pivot, and DAX - Excel

The work described in this report was accomplished during the third year of the Triservice program for extending the missile aerodynamic data base and incorporating the new data into the predictive program known as MISSILE.

These tutorials use Excel with Power Pivot enabled. For more information on Excel , click here. For guidance on enabling Power Pivot, click here. First, you need to make sure you have the Power Pivot add-in enabled. To enable Power Pivot, follow these steps. We imported Hosts by copying it and pasting it into Excel, then formatted the data as a table. To add the Hosts table to the Data Model, we need to establish a relationship. In Excel, click the Hosts tab to make it the active sheet. This step adds the Hosts table to the Data Model. It also opens the Power Pivot add-in, which you use to perform the remaining steps in this task. Notice that the Power Pivot window shows all the tables in the model, including Hosts. Click through a couple of tables. Use the slide bar to resize the diagram so that you can see all objects in the diagram. Notice that four tables are unrelated to the rest of the tables: You notice that both the Medals table and the Events table have a field called DisciplineEvent. Upon further inspection, you determine that the DisciplineEvent field in the Events table consists of unique, non-repeated values. The DisciplineEvent field represents a unique combination of each Discipline and Event. In the Medals table, however, the DisciplineEvent field repeats many times. Create a relationship between the Medals table and the Events table. A line appears between them, indicating a relationship has been established. Click the line that connects Events and Medals. The highlighted fields define the relationship, as shown in the following screen. To connect Hosts to the Data Model, we need a field with values that uniquely identify each row in the Hosts table. Then we can search our Data Model to see if that same data exists in another table. With Hosts selected, switch back to Data View. Extend the Data Model using calculated columns To establish a relationship between the Hosts table and the Data Model, and thereby extend our Data Model to include the Hosts table, Hosts must have a field that uniquely identifies each row. In addition, that field must correspond to a field in the Data Model. You can, however, create new columns by using calculated fields based on the existing data. By looking through the Hosts table, then looking at other Data Model tables, we find a good candidate for a unique field we could create in Hosts, and then associate with a table in the Data Model. Both tables will require a new, calculated column in order to meet the requirements necessary to establish a relationship. In Hosts, we can create a unique calculated column by combining the Edition field the year of the Olympics event and the Season field Summer or Winter. In the Medals table there is also an Edition field and a Season field, so if we create a calculated column in each of those tables that combines the Edition and Season fields, we can establish a relationship between Hosts and Medals. The goal is to create a calculated column in the Hosts table, and then in the Medals table, which can be used to establish a relationship between them. Select the Hosts table in Power Pivot. Adjacent to the existing columns is an empty column titled Add Column. Power Pivot provides that column as a placeholder. There are many ways to add a new column to a table in Power Pivot, one of which is to simply select the empty column that has the title Add Column. In the formula bar, type the following DAX formula. As you type, AutoComplete helps you type the fully qualified names of columns and tables, and lists the functions that are available. Use tab to select AutoComplete suggestions. You can also just click the column while typing your formula, and Power Pivot inserts the column name into your formula. Values are populated for all the rows in the calculated column. Such fields are called a primary key. You can rename any column by double-clicking it, or by right-clicking the column and choosing Rename Column. When completed, the Hosts table in Power Pivot looks like the following screen. The Hosts table is ready. Start by creating a new column in the Medals table, like we did for Hosts. Notice that Add Column is selected. This has the same effect as simply selecting Add Column. The Edition column in Medals has a different format than the Edition column in Hosts. Before we combine, or concatenate, the Edition column with the Season column to create the EditionID column, we need to create an intermediary field that gets Edition into the right format. In the formula bar above the table, type the following DAX formula. Values are populated for all the rows in the calculated column, based on the formula you entered. Rename the column by right-clicking CalculatedColumn1 and selecting Rename Column. Type Year, and then press Enter. When you created a new column, Power Pivot added another placeholder column called Add Column. In the formula bar, type the following DAX formula and press Enter. Sort the column in ascending order. The Medals table in Power Pivot now looks like the following screen. Notice many values are repeated in the Medals table EditionID field. What is unique in the Medals table is each awarded medal. The unique identifier for each record in the Medals table, and its designated primary key, is the MedalKey field. The next step is to create a relationship between Hosts and Medals. You can also switch between Grid view and Diagram view using the buttons at the bottom of the PowerView window, as shown in the following screen. Expand Hosts so you can view all of its fields. We created the EditionID column to act as the Hosts table primary key unique, non-repeated field , and created an EditionID column in the Medals table to enable establishment of a relationship between them. We need to find them both, and create a relationship. Power Pivot provides a Find feature on the ribbon, so you can search your Data Model for corresponding fields. Position the Hosts table so that it is next to Medals. Power Pivot creates a relationship between the tables based on the EditionID column, and draws a line between the two columns, indicating the relationship. In this section, you learned a new technique for adding new columns, created a calculated column using DAX, and used that column to establish a new relationship between tables. You can also use the associated data to create additional PivotTables, PivotCharts, Power View reports, and much more. Create a hierarchy Most Data Models include data that is inherently hierarchical. Common examples include calendar data, geographical data, and product categories. Creating hierarchies within Power Pivot is useful because you can drag one item to a report â€” the hierarchy â€" instead of having to assemble and order the same fields over and over. The Olympics data is also hierarchical. For each sport, there is one or more associated disciplines sometimes there are many. And for each discipline, there is one or more events again, sometimes there are many events in each discipline. The following image illustrates the hierarchy. You then use these hierarchies to see how hierarchies make organizing data easy in PivotTables and, in a subsequent tutorial, in Power View. Expand the Events table so that you can more easily see all of its fields. Press and hold Ctrl, and click the Sport, Discipline, and Event fields. With those three fields selected, right-click and select Create Hierarchy. A parent hierarchy node, Hierarchy 1, is created at the bottom of the table, and the selected columns are copied under the hierarchy as child nodes. Verify that Sport appears first in the hierarchy, then Discipline, then Event. Double-click the title, Hierarchy1, and type SDE to rename your new hierarchy. You now have a hierarchy that includes Sport, Discipline and Event. Your Events table now looks like the following screen. Create a Location hierarchy Still in Diagram View in Power Pivot, select the Hosts table and click the Create Hierarchy button in the table header, as shown in the following screen. An empty hierarchy parent node appears at the bottom of the table. Type Locations as the name for your new hierarchy. There are many ways to add columns to a hierarchy. Ensure that your hierarchy child nodes are in order. From top to bottom, the order should be: If your child nodes are out of order, simply drag them into the appropriate ordering in the hierarchy. Your table should look like the following screen. Your Data Model now has hierarchies that can be put to good use in reports. In the next section, you learn how these hierarchies can make your report creation faster, and more consistent. Use hierarchies in PivotTables Now that we have a Sports hierarchy and Locations hierarchy, we can add them to PivotTables or Power View, and quickly get results that include useful groupings of data.

Chapter 3 : GTAP Resources: Resource Display: Extending the GTAP Data Base and Model to Cover Do

The most comprehensive visualization of U.S. public data. Data USA provides an open, easy-to-use platform that turns data into knowledge.

A Visual Studio project with C source code is available to accompany this tutorial series. This tutorial describes how to display data items and data item details using ASP. This tutorial builds on the previous tutorial "UI and Navigation" and is part of the Wingtip Toy Store tutorial series. How to add a data control to display products from the database. How to connect a data control to the selected data. How to add a data control to display product details from the database. These are the features introduced in the tutorial: Model Binding Value providers Adding a Data Control to Display Products When binding data to a server control, there are a few different options you can use. The most common options include adding a data source control, adding code by hand, or using model binding. Using a Data Source Control to Bind Data Adding a data source control allows you to link the data source control to the control that displays the data. This approach allows you to declaratively connect server-side controls directly to data sources, rather than using a programmatic approach. Coding By Hand to Bind Data Adding code by hand involves reading the value, checking for a null value, attempting to convert it to the appropriate type, checking whether the conversion was successful, and finally, using the value in the query. You would use this approach when you need to retain full control over your data-access logic. Using Model Binding to Bind Data Using model binding allows you to bind results using far less code and gives you the ability to reuse the functionality throughout your application. Model binding aims to simplify working with code-focused data-access logic while still retaining the benefits of a rich, data-binding framework. The data control calls the method at the appropriate time in the page life cycle and automatically binds the returned data. In Solution Explorer, open the ProductList. Replace the existing markup with the following markup: It is useful for data in any repeating structure. This ListView example simply shows data from the database, however you can enable users to edit, insert, and delete data, and to sort and page data, all without code. By setting the ItemType property in the ListView control, the data-binding expression Item is available and the control becomes strongly typed. As mentioned in the previous tutorial, you can select details of the Item object using IntelliSense, such as specifying the ProductName: In addition, you are using model binding to specify a SelectMethod value. This value GetProducts will correspond to the method that you will add to the code behind to display products in the next step. The code will support showing products by individual category, as well as showing all products. In Solution Explorer, right-click ProductList. Replace the existing code in the ProductList. To limit the results to a specific category in the database, the code sets the categoryId value from the query string value passed to the ProductList. The QueryStringAttribute class in the System. ModelBinding namespace is used to retrieve the value of the query string variable id. This instructs model binding to try to bind a value from the query string to the categoryId parameter at run time. When a valid category is passed as a query string to the page, the results of the query are limited to those products in the database that match the categoryId value. If no query string is included when navigating to the ProductList. The sources of values for these methods are referred to as value providers such as QueryString , and the parameter attributes that indicate which value provider to use are referred to as value provider attributes such as "id". NET includes value providers and corresponding attributes for all of the typical sources of user input in a Web Forms application, such as the query string, cookies, form values, controls, view state, session state, and profile properties. You can also write custom value providers. Running the Application Run the application now to see how you can view all of the products or just a set of products limited by category. In the Solution Explorer, right-click the Default. The browser will open and show the Default. Select Cars from the product category navigation menu. Later in this tutorial, you will display product details. Select Products from the navigation menu at the top. Close the browser and return to Visual Studio. In Solution Explorer, open the ProductDetails. This markup uses methods like those that are used to display data in the ProductList. The FormView control is used to display a single record at a time from a data source. When you use the FormView control, you create templates to display and edit data-bound values. The templates contain controls, binding expressions, and formatting that define the look and functionality of the form. To connect the above markup to the database, you must add additional code to the ProductDetails. In Solution Explorer, right-click ProductDetails. Replace the existing code with the following code: If a valid query-string value is found, the matching product is displayed. If no query-string is found, or the query-string value is not valid, no product is displayed on the ProductDetails. Running the Application Now you can run the application to see an individual product displayed based on the id of the product. Press F5 while in Visual Studio to run the application. Select "Boats" from the category navigation menu. Select the "Paper Boat" product from the product list. Summary In this tutorial of the series you have add markup and code to display a product list and to display product details. During this process you have learned about strongly typed data controls, model binding, and value providers.

Chapter 4 : How to: Create an Extended Data Type | Microsoft Docs

IntroS/RExtendingRcppExamplesSummary Programming with Data: Using and extending R Dirk Eddelbuettel, Ph.D. edd@calendrierdelascience.com, calendrierdelascience.comuettel@calendrierdelascience.com

Chapter 5 : US stocks extend gains on positive reports

Let us assume you are working on the "Fleet Management Extension" package and want to first add a new field group to the FMRental table. The FMRental table is in the fleet management model. Here's what you would do.

Chapter 6 : Display Data Items and Details | Microsoft Docs

Extending Knowledge Bases Using Images Vincent P. A. Lonij, Ambrish Rawat, Maria-Irina Nicolae IBM Research - Ireland Mulhuddart, Dublin 15, Ireland.

Chapter 7 : Charts & Graphs

research project, 'Multidimensional child poverty and disadvantage: tackling "data exclusion" and extending the evidence base on "missing" and "invisible" children'. The.

Chapter 8 : How to auto populate other cells when selecting values in Excel drop down list?

Longitudinal studies of the application of a paraphrasing model to to month-olds indicated that mean length of utterance was significantly correlated with realized and paraphrased frequencies of several linguistic items in the subjects' corpora. The model was productive for examining children.

Chapter 9 : GTAP Resources: Resource Display: The GMig2 Data Base: Extending GTAP 8 to Include G

US three main indices end month lower US equities extended gains on Wednesday supported by better than expected corporate reports and economic data. The S&P gained % to