AtomicDB API in *Mathematica*

Professional Version

By Athanassios I. Hatzis, PhD - (C) April 2015

Test Add and Get Commands with Pro Version

Test Preparation

Load Application

```
ClearAll["Global`*"]
```

```
<< AtomicDBAddOn`
```

Shortened Commands

Primitive Commands

```
login = ADBloginToServer;
getAny = ADBgetAnything;
setAny = ADBsetAnything;
addAny = ADBaddAnything;
assAny = ADBassAnything;
impAny = ADBimpAnything;
```

Enhanced Commands

```
getMod = ADBgetModels;
getCon = ADBgetConcepts;
getCol = ADBgetCollections;
getGrp = ADBgetGroups;
getItm = ADBgetItems;
getSrcD = ADBgetDataSources;
getSrcT = ADBgetTables;
getSrcC = ADBgetColumns;
```

```
addMod = ADBaddModel;
addCon = ADBaddConcepts;
addCol = ADBaddCollections;
addRec = ADBaddRecords;
```

Output Commands

```
prnObj = PrintADBobj;
prnObjOut = PrintADBobjOut;
nout = PrintOut;
ntext = PrintOutText;
```

Transformation Commands

```
toLists = ADBobjToLists;
toRules = ADBobjToRules;
toAssocs = ADBobjToAssocs;

toKey = WLlistToADBkey;
toKvp = WLruleToADBkvp;
toRec = WLrecToADBrec ;
```

Test Predicates Commands

```
keyQ = ADBkQ;
keyLQ = ADBkLQ;
keyLLQ = ADBkLLQ;

kvpQ = ADBkvpQ;
kvpLQ = ADBkvpLQ;
kvpLLQ = ADBkvpLLQ;

assocsQ = WLassocsQ;
rulesQ = WLrulesQ;
```

GetBy Commands

```
getByKey = WLgetByKey;
getByVal = WLgetByVal;
```

Titles

```
ntext["AtomicDB Add-On in Mathematica", "Title"]
ntext["Professional Version", "Subtitle"]
ntext["By Athanassios I. Hatzis" <> " - (C) " <> DateString[], "Subtitle"]
ntext["This output has been generated automagically. " <> "©", "Subsubtitle"]
```

Description of this Demo

```
nout["In this demo we build first  a simple relational data model using the Wolfram List
   structure.Our relational model example includes two main tables STOCK and ORDER that
   are joined with a third junction table STOCK-ORDER. Then we convert this to AtomicDB
   data model by adding a new Model, then Concepts (columns) and Records (rows)."]
```

Relational Model

```
ntext["Relational Model", "Subchapter"];
```

Headers of the Tables

Headers are lists of column names, i.e. attribute names.

```
stockHeader = {"StockID", "StockNameEN", "StockPrice", "StockNameGR"};

orderHeader = {"OrderID", "OrderKey"};

soHeader = {"SOID", "SOOrderID", "SOStockID", "SOQuantity"};
```

Body of the Tables

The body of the table is the relation data set and it is represented with a list of records. Each record is represented with a list of values.

```
stockRelData = {{991, "Pinto Beans", 11.1`, "Φασόλια Πίντο"},
    {992, "Kidney Beans", 9.85`, "Φασόλια Κόκκινα"}, {993, "White Beans",
     13.45`, "Φασόλια Άσπρα"}, {994, "Wax Beans", 18.72`, "Φασόλια Καναρίνια"}};

orderRelData = {{441, "1111-BZ"}, {442, "1117-CM"}, {443, "1118-SA"}, {444, "1119-TT"}};

soRelData =
  {{224, 441, 991, 1}, {225, 442, 992, 3}, {226, 443, 994, 2}, {227, 444, 993, 1}, {228, 441, 993, 3}};
```

Relation Sets

```
ntext["Relations", "Section"];

ntext["STOCK Table", "Subsection"];

(stockRelSet = Insert[stockRelData, stockHeader, 1]) // TableForm // nout

ntext["ORDER Table", "Subsection"];

(orderRelSet = Insert[orderRelData, orderHeader, 1]) // TableForm // nout

ntext["STOCK-ORDER Table", "Subsection"];

(soRelSet = Insert[soRelData, soHeader, 1]) // TableForm // nout
```

AtomicDB Model

```
ntext["AtomicDB Model", "Subchapter"];
```

Login To Server

```
ntext["Login To Server", "Section"]

ntext["Existing Models", "Subsection"];

(modelKeys = login["localhost", "System Administrator", "Wind0ws7", "ManageIT"]) // prnObjOut
```

Add A Model

```
ntext["Concept Map System", "Section"]

modelName = "Beans Stock-Order Model Added with ADBAddOn Pro Version";

ntext["Add A New Model", "Subsection"];

(res1 = addMod[modelName]) // prnObjOut
```

Get Command

```
ntext["Get All Models", "Subsection"];

getMod[] // prnObjOut
```

Add Concepts to the Model

```
ntext["Add Concepts to the Model", "Subsection"];

stockConceptsNames = Insert[stockHeader, "StockNEXUS", 1];

orderConceptsNames = Insert[orderHeader, "OrderNEXUS", 1];

soConceptsNames = {"SONEXUS", "SOID", "OrderID", "StockID", "SOQuantity"};
```

Add STOCK Group Concepts

```
ntext["Add STOCK Group Concepts", "Subsubsection"]
```

```
addCon[modelName, stockConceptsNames] // prnObjOut
```

Add ORDER Group Concepts

```
ntext["Add ORDER Group Concepts", "Subsubsection"]
```

```
addCon[modelName, orderConceptsNames] // prnObjOut
```

Add STOCK-ORDER Group Concepts

```
ntext["Add STOCK-ORDER Group Concepts", "Subsubsection"]
```

```
addCon[modelName, soConceptsNames] // prnObjOut
```

Add Collections Auto-generated from Concepts

```
ntext["Data Holder System", "Section"]
```

```
ntext["Add Collections", "Subsection"];
```

STOCK Group Collections

```
ntext["Add STOCK Group Collections", "Subsubsection"]
```

```
addCol[modelName, stockConceptsNames] // prnObjOut
```

ORDER Group Collections

```
ntext["Add ORDER Group Collections", "Subsubsection"]
```

```
addCol[modelName, orderConceptsNames] // prnObjOut
```

STOCK-ORDER Group Collections

```
ntext["Add STOCK-ORDER Group Collections", "Subsubsection"]
```

```
addCol[modelName, soConceptsNames] // prnObjOut
```

Add Records

```
ntext["Add Records", "Subsection"]
```

Add Records to the STOCK Group

```
ntext["Add STOCK Group Records", "Subsubsection"]
```

```
addRec[modelName, stockConceptsNames, stockRelSet] // prnObjOut
```

Add Records to the ORDER Group

```
ntext["Add ORDER Group Records", "Subsubsection"]
```

```
addRec[modelName, orderConceptsNames, orderRelSet] // prnObjOut
```

Add Records to the STOCK-ORDER Group

```
ntext["Add STOCK-ORDER Group Records", "Subsubsection"]
```

```
addRec[modelName, soConceptsNames, soRelSet] // prnObjOut
```