Relational Dataset Analytics for Clear Customer Insights

May 2023

Bob Foreman
Software Engineer Lead
LexisNexis Risk Solutions

BOSTON | MAY 9–11
HYNES CONVENTION CENTER

The Goal of this Workshop

This workshop will introduce our open source HPCC Systems platform, and the underlying Enterprise Control Language (ECL) used for Big Data Transformation and Analytics.

We will look at the power of the open data model, transforming normalized data into a denormalized relational dataset, and use the implicit relationality power to analyze relationships to provide better customer insight.

What you will need…

- ✓ Your favorite browser
- ✓ An internet connection
- ✓ GitHub Account (free and open source)
- ✓ We will be using **GitPods** to fast track an environment that you can use to examine, modify, and run the workshop code.

The GitPods and Cluster link:

The Environment:

https://gitpod.io/#https://github.com/hpccsystems-solutions-lab/ODSCEast2023

The Workshop Cluster:

http://training.us-hpccsystems-dev.azure.lnrsg.io:8010/

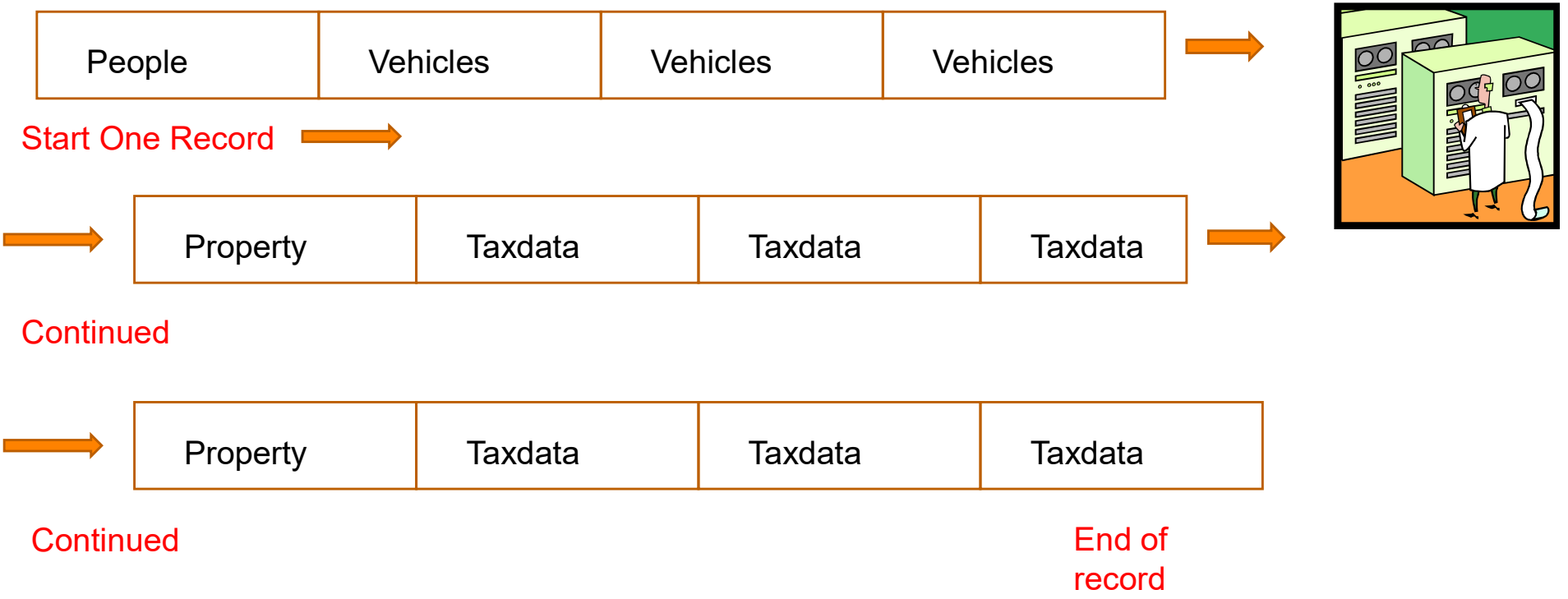
Running today through Thursday 6 P.M.

Training Data:

Our Workshop Database:
(a 3-level hierarchical relational database)

- **People**
  - **Vehicle**
  - **Property**
    - **Taxdata**

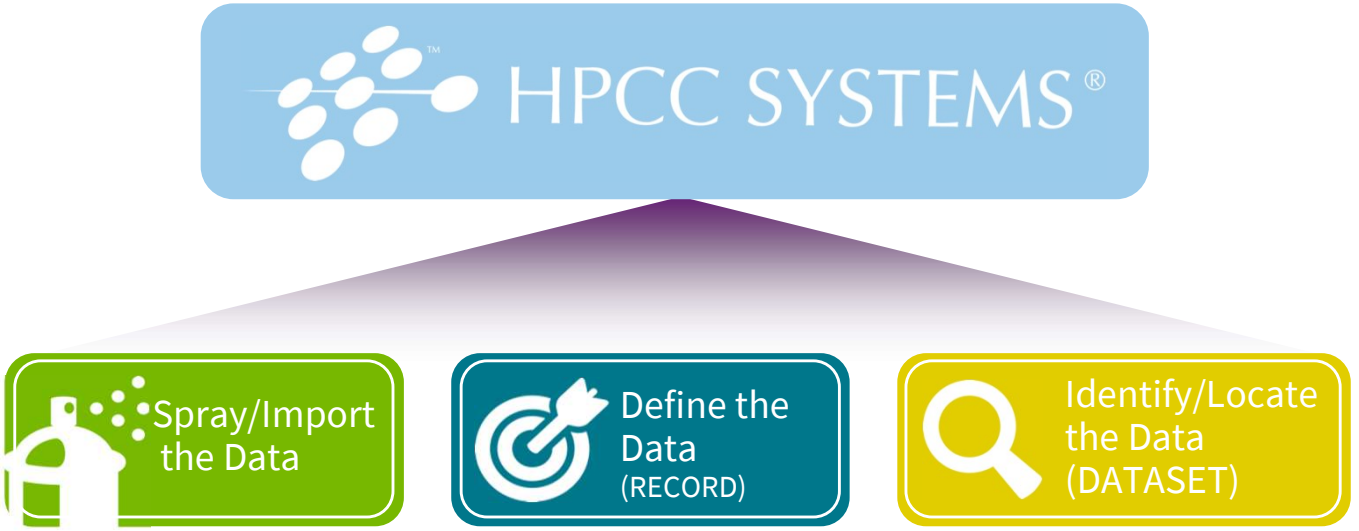
Denormalizing Related Data:



The Data “Donut” (Normalized Data)

Three ECL Data Rules

Before you begin to work on any data in the HPCC cluster, you must always do three things:



Normalized RECORD and DATASET:

```
EXPORT File_People := MODULE
EXPORT Layout := RECORD
  UNSIGNED8 id;
  STRING15  firstname;
  STRING25  lastname;
  STRING15  middlename;
  STRING2   namesuffix;
  STRING8   filedate;
  STRING1   gender;
  STRING8   birthdate;
  END;
EXPORT File := DATASET('~ODSCEast::People',Layout,THOR);
END;
```

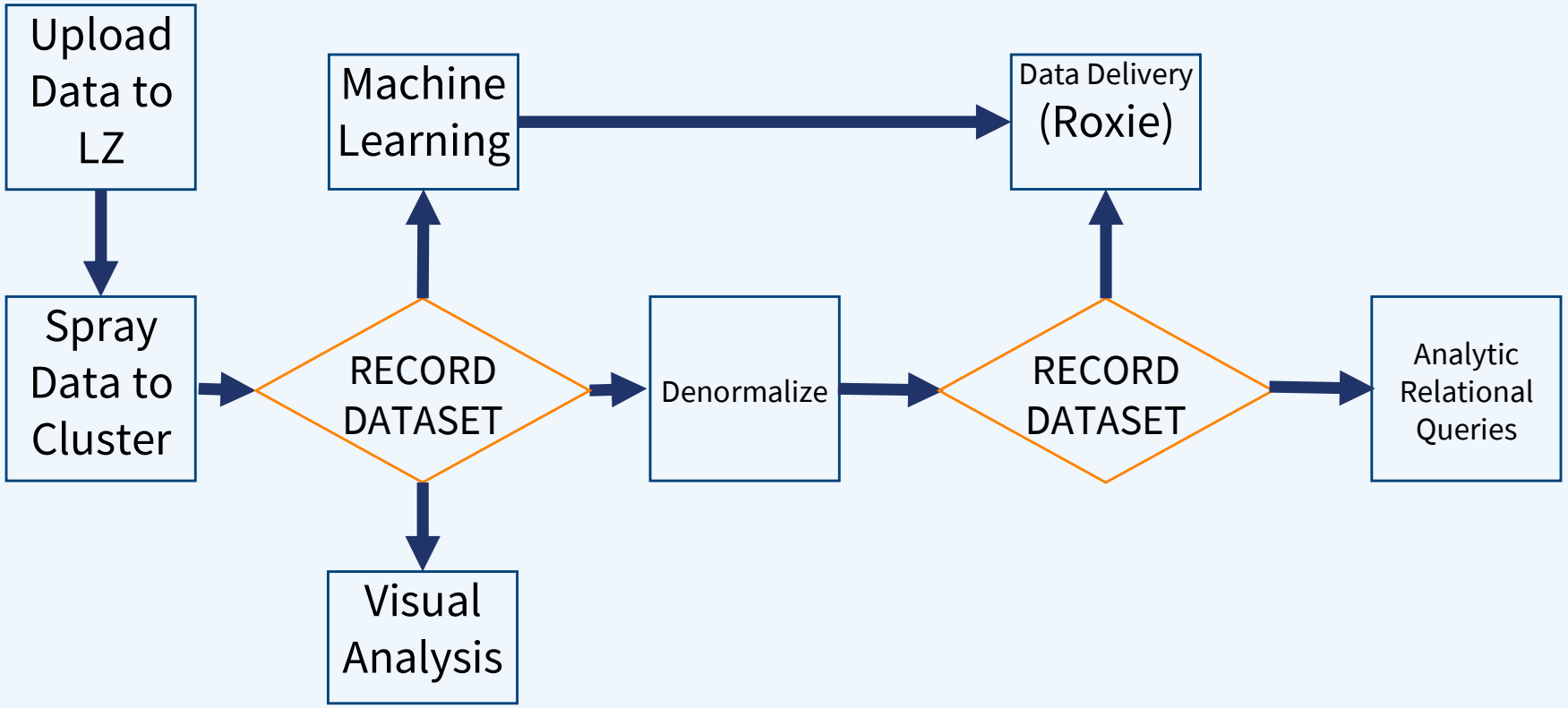
Nested Child Dataset RECORD:

```
EXPORT PropTax := RECORD
  $.File_Property.Layout;
  UNSIGNED1 ChildTaxCount;
  DATASET($.File_Taxdata.Layout) TaxRecs{MAXCOUNT(20)};
END;

EXPORT PeopleAll := RECORD
  $.File_People.Layout;
  UNSIGNED1 ChildVcount;
  UNSIGNED1 ChildPcount;
  DATASET($.File_Vehicle.Layout) VehicleRecs{MAXCOUNT(20)};
  DATASET(PropTax) PropRecs{MAXCOUNT(20)};
 END;

 EXPORT People   := DATASET('~ODSCEast::OUT::PeopleAll',$.Layouts.PeopleAll,FLAT);
```

Workshop Workflow



Working with Child Datasets

- ✓DATASET fields in RECORD Structures
- ✓TRANSFORM Function
- ✓DENORMALIZE Function

ECL TRANSFORM Structure:

```
resulttype  funcname( parameterlist ) := TRANSFORM
   SELF.outfield := transformation;
END;
```

✓*resulttype* – The name of a RECORD structure attribute specifying the output format of the function.

✓*funcname* – The name of the function the TRANSFORM structure defines.

✓*parameterlist* – The value types and labels of the parameters that will be passed to the TRANSFORM function.

✓**SELF** – Indicates the *resulttype* structure.

✓*outfield* – The name of a field in the *resulttype* structure.

✓*transformation* – An expression specifying how to produce the value assigned to the *outfield*.

DENORMALIZE Function:

DENORMALIZE(*parentoutput*,*childrecset*,*condition*,*transform***)**

✓*parentoutput* – The set of parent records already formatted as the result of the combination.

✓*childrecset* – The set of child records to process.

✓*condition* – An expression that specifies how to match records between the parent and child records.

✓*transform* – The TRANSFORM function to call.

The **DENORMALIZE** function forms flat file records from a parent and any number of children.

The *transform* function must take at least 2 parameters: a LEFT record of the same format as the resulting combined parent and child records, and a RIGHT record of the same format as the *childrecset*. An optional integer COUNTER parameter can be included which indicates the current iteration of child record.

DENORMALIZE Workshop Example:

✓DENORMALIZE the Data:

BWR_DenormalizePeople

Querying Relational Data:

Implicit Dataset Relationality
(Nested child datasets):

- ✓ Parent record fields are always in memory when operating at the level of the Child
- ✓ You may only reference the related *set* of Child records when operating at the level of the Parent

- People
  - Vehicle
  - Property
    - Taxdata

Helper Functions:

ThisYear

```
EXPORT ThisYear := 2023; //base year
```

IsValidAmount

```
EXPORT IsValidAmount(integer amt) := amt BETWEEN 1 AND 9999998;
```

IsValidYear

```
IMPORT $;
EXPORT IsValidyear(integer pyear) := pyear > 1900 and pyear <= $.ThisYear ;
```

YearsOld

```
IMPORT $;
EXPORT YearsOld(integer4 datex) :=
            (INTEGER4)  IF($.IsValidYear(datex), ($.ThisYear - datex), datex);
```

Relational Queries: Property Focused

1. Calculate the total number of properties that have, or have *ever* had, 3 or more bedrooms. (**PropTaxBeds3**)
2. Calculate the number of Property records with 3 or more bathrooms, *ever*. This will be used to calculate number of the specified properties for each person. For the purpose of this definition, the number of "bathrooms" is defined as:

   The number of full baths, plus, the rounded number of half baths divided by two. (**PropBath_3**)

- People
  - Vehicle
  - Property
    - Taxdata

Relational Queries: Property Focused

4. Calculate the aggregate Property values for all properties on “small” streets (CT, LN, WAY, CIR, PL,or TRL). (**PropValSmallStreet**)
5. Calculate the number of Properties with exactly 3 bedrooms in a year acquired within 5-15 years ago, or exactly 3 bedrooms in a tax year within 5-15 years ago. (**PropTaxBedYearRange**)
6. Calculate the assessed total value from the most recently reported Taxdata record for the most recently acquired Property that is not an apartment.

   If there are multiple properties for the same year, use the one with the highest property value. Output the result as an 8-character string. (**PropTaxdataHomeAssess**)

Relational Queries: Vehicle Focused

1. Calculate the price of the newest truck and output it as a 6 character string. (**VehicleNewTruckPrice**)
2. Calculate the number of unique vehicles owned, based on their make code. If no vehicles exist, output a -9 value. (**CountUniqueMakeVehicles**)
3. Calculate the number of Ford Vehicles purchased within 90 days of purchasing a Chevrolet Vehicle. If no Vehicles exist , output a –9 value.(**FordChevWithin90**)

- People
  - Vehicle
  - Property
    - Taxdata

Relational Queries: Vehicle *and* Property!

1. Calculate the sum of Property and Vehicle records for each person that meet the below-specified criteria, limiting the final result to a maximum of 5.

   To be included, the Vehicle must be a Passenger car made in the last three years whose price is > 15,000.

   To be included, the Property must be a "non-apartment" built in the last ten years whose total value is > 150,000. (**PropVehSumEx**)

- People
  - Vehicle
  - Property
    - Taxdata

Normalizing your data

- ✓NORMALIZE Function
- ✓Using COUNTs or CHILDREN (NORMALIZE forms)

NORMALIZE Function

NORMALIZE(*recordset*, *expression*, *transform***)**

✓*recordset* – The set of records to process.

✓*expression* – An numeric expression specifying the total number of times to call the *transform* for that record.

✓*transform* – The TRANSFORM function to call for each record in the *recordset*.

The **NORMALIZE** function processes through all the records in the *recordset* performing the *transform* function the *expression* number of times on each record in turn to produce relational child records of the parent.

The *transform* function must take two parameters: A LEFT record of the same format as the *recordset*, and an integer COUNTER specifying the number of times to call the *transform* for that record. The format of the resulting recordset can be different from the input.

NORMALIZE Workshop Example:

✓NORMALIZE the Data:

BWR_NormalizePeople
BWR_Renorm_People

Built-In Visualization Tools

HPCC Systems provides built-in Visualization of your output data in a variety of charts and graphs. You can visualize your data in three ways:

- Using the Chart Tool in the ECL Playground.
- Accessing the Visualize tab in all ECL workunits
- Using the Resources tab in conjunction with the ECL Visualizer bundle.

Installing the Visualizer Bundle:

```
ecl bundle install https://github.com/hpcc-systems/Visualizer.git
```

Visualization Workshop Example:

✓Visualize the Data:

BWR_StateChoropleth

Built-in Machine Learning Support:

Classical Machine Learning

Unsupervised

Clustering
DBSCAN
K-Means

Pattern Search
Text Vectors
Levenshtein Deletion Neighborhood

Dimension Reduction
PCA

Supervised

Classification
SVM
Decision Trees
Logistic Regression
Classification Forest
Latent Dirichlet Allocation (Topic Modeling)

Regression
Linear Regression
Regression Forest

https://hpccsystems.com/download/free-modules/hpcc-systems-machine-learning-library/

Neural Nets & Deep Learning

Autoencoders

Convolutional Neural Networks

Recurrent Neural Networks

Perceptrons

Ensemble Methods

Random Forest

Gradient Boosted Forest

Gradient Boosted Trees

Machine Learning Basics:

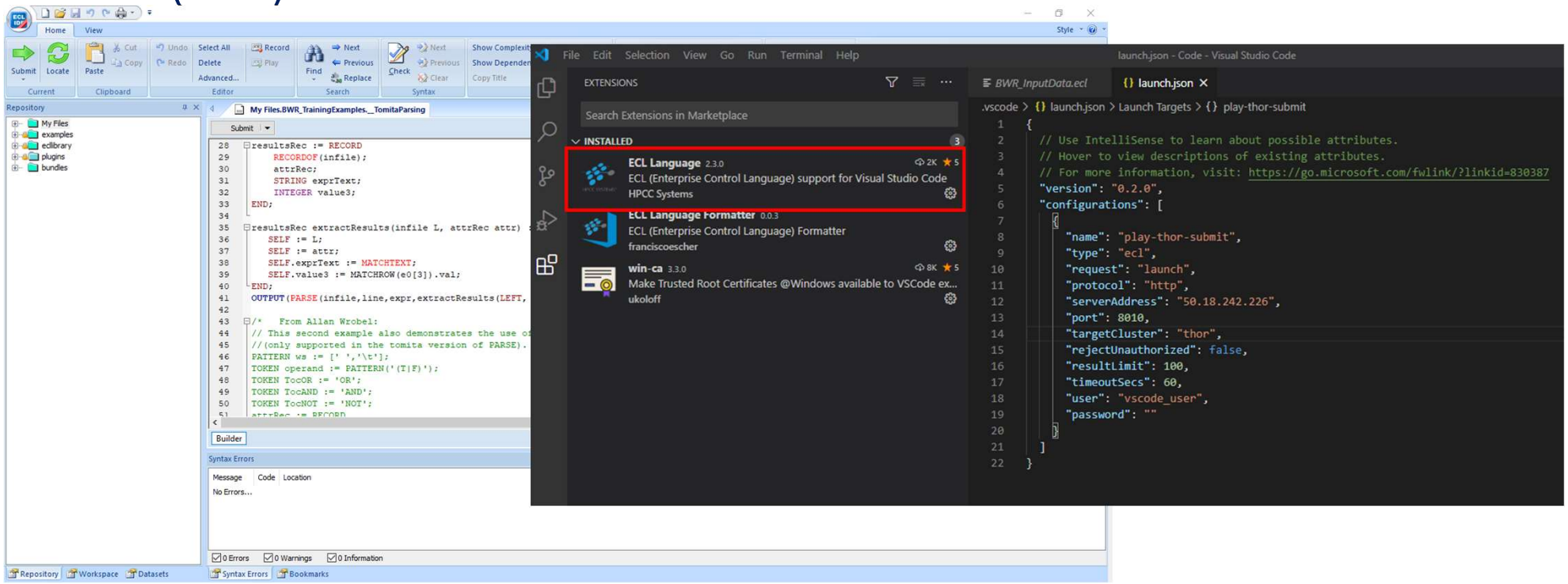
Steps for using ML with HPCC Systems:

1. Install the target ML bundles
2. Prepare Your Data (Clean, Train, Test)
3. Train the Model
4. Assess the Model (Accuracy of Test)
5. Make Predictions (Apply new data)

Integrated Development Environments

ECL IDE (Win)

Visual Studio Code (Ux/MacOS)



And CLI too! ECL.EXE

Want to know more?

Portal:

https://hpccsystems.com

Free online training (138 classes and counting!):

https://learn.lexisnexis.com/hpcc

Free HPCC Clusters (try it out!):

https://play.hpccsystems.com:18010

Please email our training team:

training@hpccsystems.com

Join our Community

Help us make HPCC Systems better. Register on our community portal.