WHITE PAPER

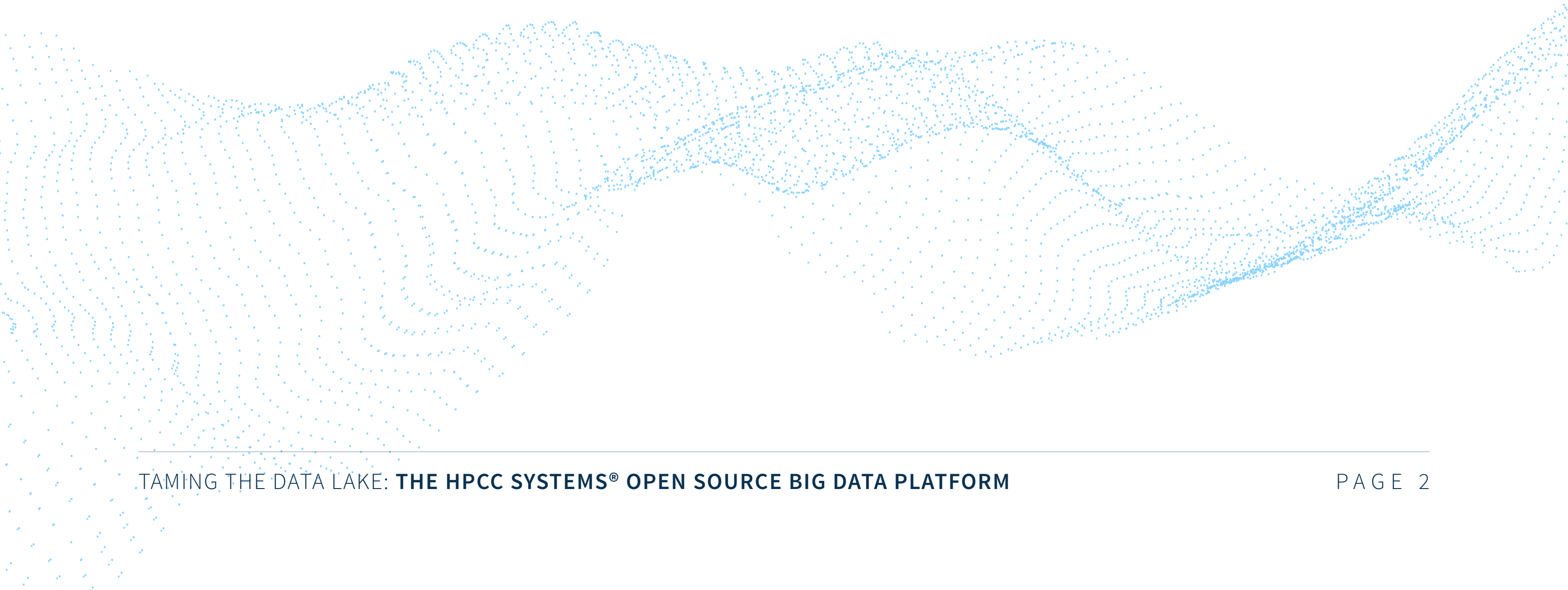
Taming the Data Lake: The HPCC Systems® Open Source Big Data Platform

OVERVIEW

A “Data Lake” is an architecture and methodology for the continuous management of complex data that stores data on raw format for increased agility on data exploration. As it enters the lake, each piece of data is readily available for manipulations and insights via a unique identifier and a set of extended metadata tags. In contrast, a “Data Warehouse” stores data in a predefined format for faster delivery of data analysis results. HPCC Systems offers the best of both worlds by combining the fast performance of a Data Warehouse for information delivery with the ability to treat data as if it were in a Data Lake when it comes to data exploration. HPCC Systems uses distributed data architecture and a parallel processing methodology in order to work with large datasets. Enterprises are adopting data lake technology to manage their rapidly growing internal datasets and to solve complex problems through data analysis to improve their relationships with customers and suppliers.

> HPCC Systems is an open source platform for big data implementations, whether as a data lake or data warehouse.

HPCC Systems is an open source platform for big data implementations, whether as a data lake or data warehouse. Specifically designed to facilitate data discovery and experimentation, HPCC Systems provides users with a clear path from data discovery to production.



HPCC Systems: A Brief History

The big data platform that would become HPCC Systems was developed in 2001 by an in-house engineering team at LexisNexis® Risk Solutions. The beginnings of the technological foundation for the HPCC Systems platform were first developed at Seisint in 1999 in order to manage significant numbers of datasets. In 2000, the team at Seisint needed to collect and analyze massive amounts of raw consumer financial data from a wide variety of sources for a customer responsible for determining consumer credit scores in the United States. This led to the creation of the programming language, known as ECL (Enterprise Control Language), that HPCC Systems uses today. In 2004, LexisNexis® Risk Solutions (LNRS) acquired Seisint along with its foundational technology including the ECL language that programs Thor and Roxie clusters. During this time, HPCC Systems mainly served as an internal tool for LexisNexis Risk Solutions and had not yet reached its full potential. In 2008, LexisNexis Risk Solutions acquired ChoicePoint, an insurance analytics provider, and over the next three years, ChoicePoint's product portfolio was integrated into the HPCC Systems platform. Combining the HPCC Systems platform with the ChoicePoint insurance products created powerful business advantages -- an extremely efficient big data processing solution capable of managing massive amounts of data driving much more efficient data products into the already massive Insurance market.

In 2011, LexisNexis decided to release HPCC Systems under an opensource license. Since its launch, HPCC Systems has developed a rich user community and a global network of customers. Additionally, HPCC Systems continues to innovate the HPCC Systems platform by adding support for cloud-based data centers; developing new data cataloging, governance, and orchestration tools; and adding new administration and dashboard reporting capabilities. Current HPCC Systems users that can be mentioned publicly include Quod, the new credit bureau for Brazil that LNRS helped to create, and many prominent universities: Clemson University, Florida Atlantic University, Kennesaw State University, RV College of Engineering and several other universities globally.

Since its beginning, HPCC Systems has given its users a platform consisting of a single homogenous data pipeline. This significantly minimizes the amount of effort spent on platform management, installation and maintenance for users, which is a crucial benefit of this platform. Perfect for both data lakes and warehouses, HPCC Systems is extremely capable and efficient in processing large amounts of data due to infrastructure servers including Thor and Roxie which manage the platform's various functions.

What is a Data Lake?

Technologies like social media, e-commerce, and the Internet of Things (IoT) are fueling massive growth in the amount of data organizations need to store and analyze. According to a 2022 report from Statista, global data creation is projected to grow to over 180 zettabytes by 2025.

This kind of dramatic growth creates problems for organizations as they work to capture, classify, and analyze the data generated by customers and their own employees and operations. Further complicating the situation is the variety of data formats organizations now collect: structured and semi-structured data, unstructured data (emails, documents, PDFs), and binary data (images, audio, video). Finally, some of this data will include information that must remain private and protected to meet various legal and service level agreement (SLA) requirements for the use and storage of sensitive data.

Trying to manage data in real time as it pours into an organization's datacenter is challenging. Taking new data and adjusting it to meet the format requirements of existing databases uses up valuable resources, both in terms of servers and personnel. Respondents to a McKinsey 2019 Global Data Transformation Survey reported that an average of 30 percent of their total enterprise time was spent on non-value-added tasks because of poor data quality and availability. Legacy data management systems also tend to create data silos: collections of separate databases that don't communicate with each other due to mismanagement. Data silos can result in flawed data analytics as algorithms attempt to create data-driven insights without access to complete and/or correct data assets.

Data Lakes support extremely large, complex, and diverse datasets, and they easily accommodate new data sources such as IoT. They allow IT groups to quickly create new applications that support changing business needs by unlocking the power of complex data for all users within the organization. They also scale much more easily and cost-effectively than relational databases, and offer the huge storage and compute resources needed for data analytics. As a result, Data Lakes enable greater responsiveness for users and external customers, reduced costs, and greater scalability.

- Facilitating the rapid development of new data sources and analytic algorithms
- Allowing multiple users to access data and create customized applications and reports
- Supporting the easy addition of new data sources to provide a continuous value enhancement to the data available to users

What is HPCC Systems?

HPCC Systems is an opensource data lake platform designed to continuously acquire data from many data sources in both structured and unstructured formats. Data is usually stored in simple flat files like basic disk files, or in object stores like Amazon S3 and Azure BLOB storage. In other words, there is no predefined schema into which data needs to be fitted before storage. A typical HPCC Systems implementation begins with just a few data sources and some initial analytical and reporting tools, but the size, complexity, and capability of the HPCC Systems data lake can grow quickly. Once data is added to the data lake, the process of data enrichment begins. Data enrichment is an evolving, iterative process that extracts as much knowledge as possible from data sources. Once that knowledge is extracted, it's available to other data lake users that need it via a process known as data delivery. During data delivery, HPCC Systems ensures that data is transferred to data lake users in a responsive, secure, and reportable manner. An analogy can help illustrate what happens to data as it enters through an HPCC Systems data lake. In this analogy, water (raw data) is collected in a reservoir (data lake) where it is then processed to make it fit for consumption by the public.

In the illustration below, water (1. Multiple types of raw data) is collected from a variety of sources: rain, snow, and melt water runoff. That water is then collected (via 2. Batch, real-time, and streaming ingestion) in a reservoir for distribution to the local populace (5. Information delivery). However, before that water is ready for public consumption, it must be processed (3. Integrate, profile, and clean). Furthermore, other chemicals can be added to the water to improve its taste or health benefits prior to consumption (4. Enrichment).

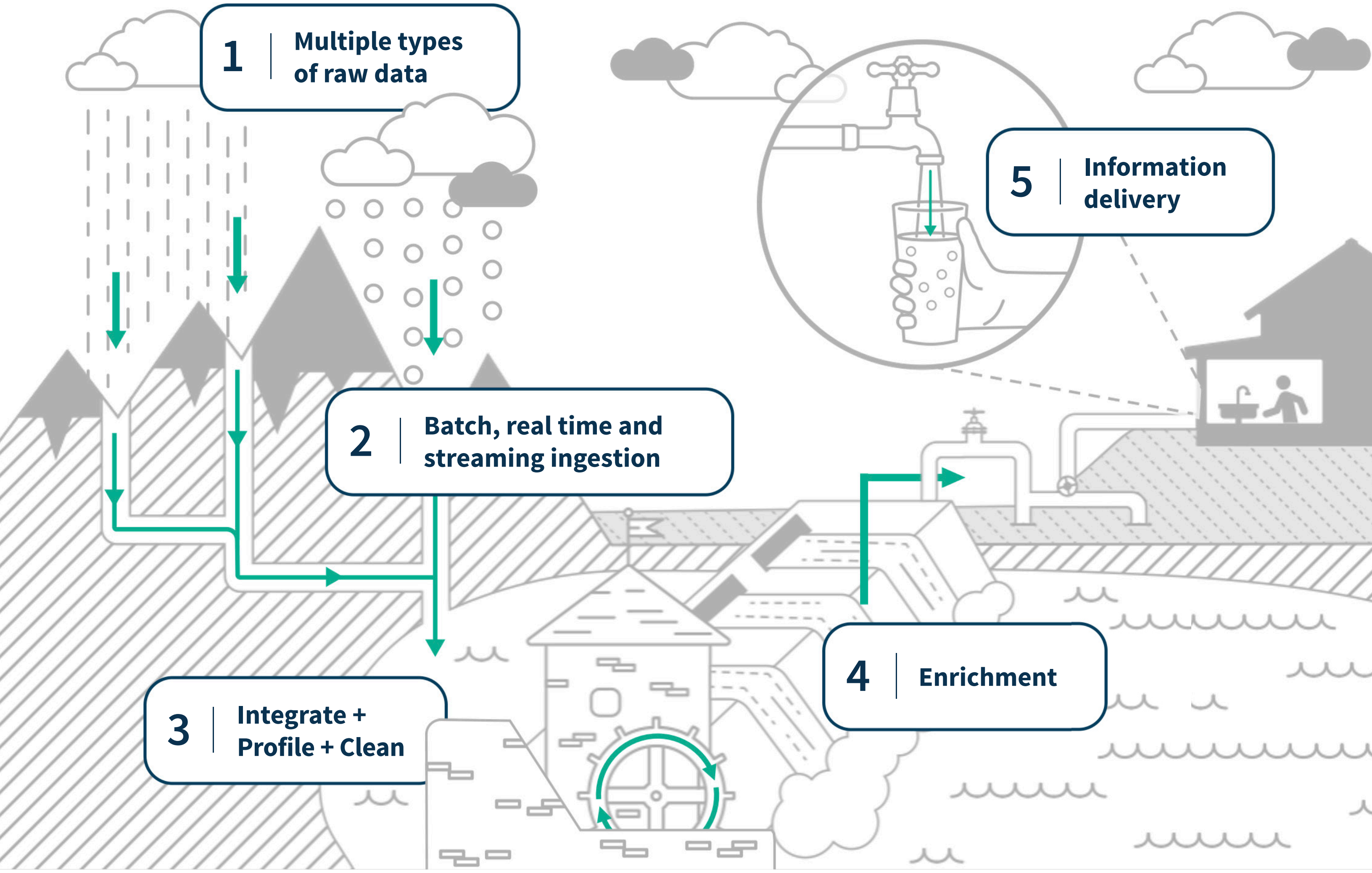

1 Multiple types of raw data
5 Information delivery
2 Batch, real time and streaming ingestion
4 Enrichment
3 Integrate + Profile + Clean

Illustration 1: A data lake must be able to ingest, format, and enrich data with 24/7 availability.

In order to keep delivering water to the public, the processing plant can't stop either gathering or processing water; the process must reliably deliver water 24 hours a day, seven days a week to keep up with consumer demand. A data lake must offer the same level of availability. No matter how much data is added to the lake, the process of cataloging and analyzing that data must continuously operate at "five nines" levels of service (the data lake must be up and operational at least 99.999 percent of the time).

The illustration below captures the data lifecycle in an actual HPCC Systems data lake currently in use by an HPCC Systems customer. Moving left to right, the data sources deliver data to the HPCC Systems data lake for refinement, enrichment, indexing, and analysis. HPCC Systems can generate reports or dashboards about the data at any step in the process, depending on what reports the consumer needs. All of these processes occur within the data lake environment to produce consistent results.

HPCC Data Enrichment Pipeline

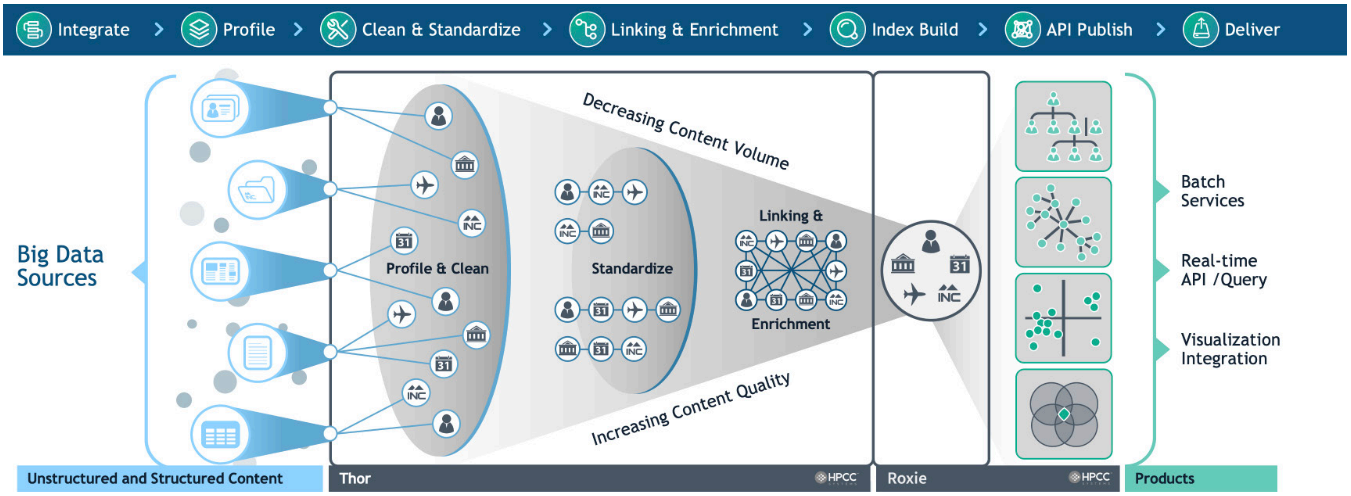


Illustration 2: The HPCC Systems data pipeline above follows data from the source to its ingestion into the HPCC Systems cluster, where it is formatted, enriched, and then made available to applications hosted in the cluster.

COMPONENTS OF THE HPCC SYSTEMS PLATFORM

An HPCC Systems Data Lake is comprised of the following components:

- The **ECL (Enterprise Control Language)** programming language is a data-oriented, declarative programming language developed for use on HPCC Systems data lakes.
- **Thor** is a bulk data processing cluster that cleans, standardizes, and indexes inbound data for use by the data lake. Once data has been refined by Thor it can then be used by the Roxie cluster.
- **Roxie** is a real-time API/Query cluster for querying data after refinement by Thor. Roxie queries execute in sub-second times and provide very high concurrency.

HPCC Systems Platform

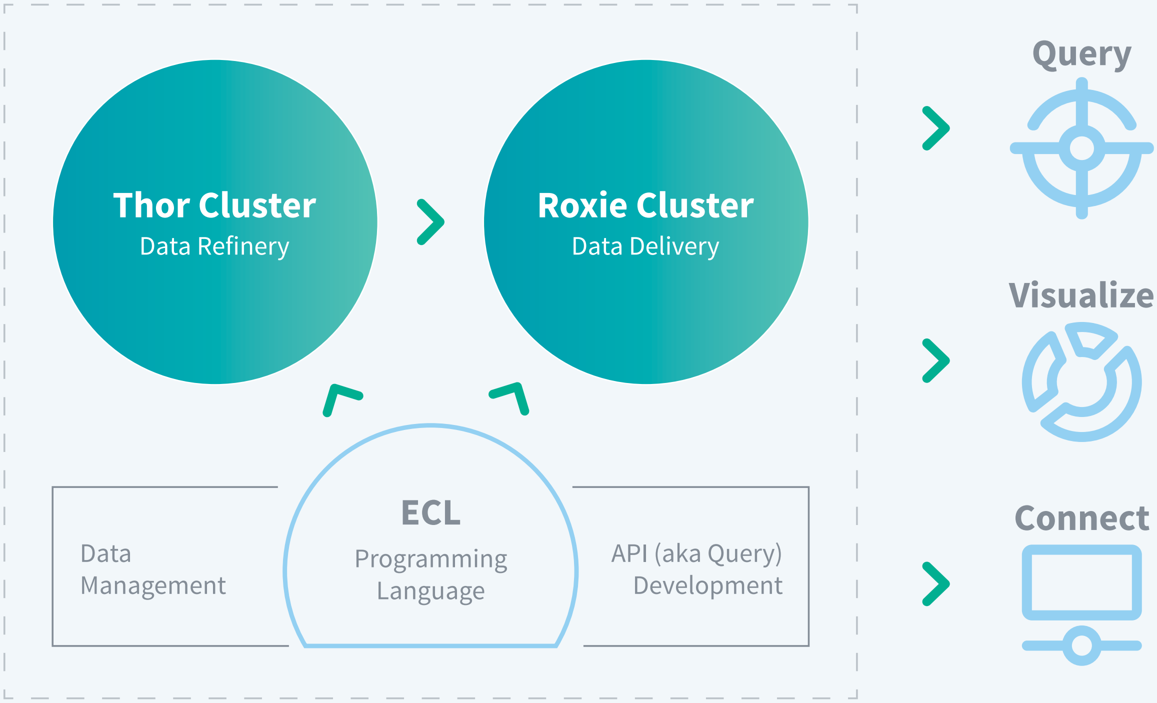
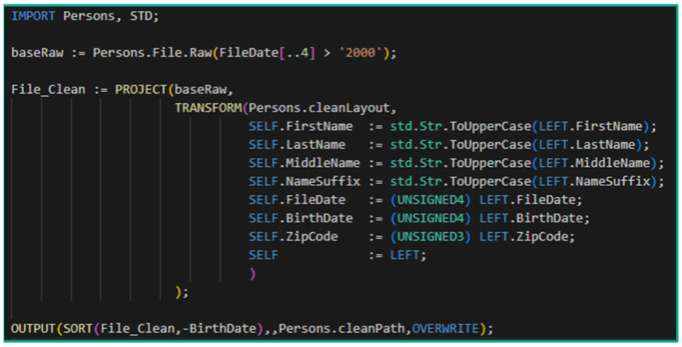


Illustration 3: An HPCC Systems system diagram featuring a Thor cluster (for bulk data processing) and a Roxie cluster (for handling data queries)

About ECL

ECL is a declarative programming language, which presents numerous advantages over the more conventional imperative programming model. Declarative programming allows the programmer to express the logic of a computation without describing its flow control. In layman's terms, ECL lets developers tell the system what they need, but leaves it up to the system to determine the best way to go about doing it. In addition to simplifying the design and implementation of complex algorithms, it also improves the quality of the code by minimizing or eliminating side effects in the data lake's code, which makes code testing easier and simplifies code maintenance. ECL code is easier to understand, verify and extend, even by people who are not familiar with the original design, which helps shorten the learning curve for new programmers.

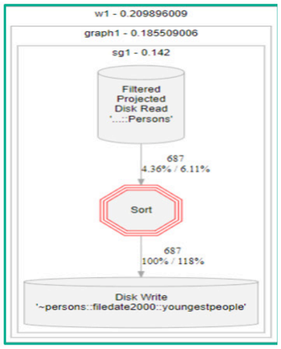


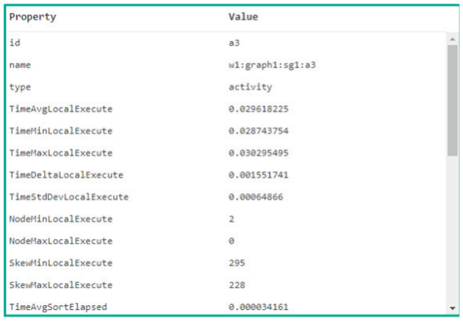
```
IMPORT Persons, STD;

baseRaw := Persons.File.Raw(FileDate[..4] > '2000');

File_Clean := PROJECT(baseRaw,
                      TRANSFORM(Persons.cleanLayout,
                                SELF.FirstName  := std.Str.ToUpperCase(LEFT.FirstName);
                                SELF.LastName   := std.Str.ToUpperCase(LEFT.LastName);
                                SELF.MiddleName := std.Str.ToUpperCase(LEFT.MiddleName);
                                SELF.NameSuffix := std.Str.ToUpperCase(LEFT.NameSuffix);
                                SELF.FileDate   := (UNSIGNED4) LEFT.FileDate;
                                SELF.BirthDate  := (UNSIGNED4) LEFT.BirthDate;
                                SELF.ZipCode    := (UNSIGNED3) LEFT.ZipCode;
                                SELF            := LEFT;
                                )
                      );

OUTPUT(SORT(File_Clean,-BirthDate),,Persons.cleanPath,OVERWRITE);
```


w1 - 0.209896009
graph1 - 0.185509006
sg1 - 0.142
Filtered Projected Disk Read '...::Persons'
687
4.36% / 6.11%
Sort
687
100% / 118%
Disk Write '~persons::filedate2000::youngestpeople'



Property	Value
id	a3
name	w1:graph1:sg1:a3
type	activity
TimeAvgLocalExecute	0.029618225
TimeMinLocalExecute	0.028743754
TimeMaxLocalExecute	0.030295495
TimeDeltaLocalExecute	0.001551741
TimeStdDevLocalExecute	0.00064866
NodeMinLocalExecute	2
NodeMaxLocalExecute	0
SkewMinLocalExecute	295
SkewMaxLocalExecute	228
TimeAvgSortElapsed	0.000034161

Illustration 4: An ECL code sample

ECL is the one language needed to express data algorithms across the entire HPCC Systems platform. In Thor, ECL expresses data workflows consisting of data loading, transformation, linking, indexing, etc. In Roxie, it defines data queries (the HPCC Systems equivalent to stored procedures in traditional RDBMS). This means HPCC Systems data analysts and programmers only need to learn one language to define the complete data lifecycle.

ECL is highly extensible: as new attributes are defined, they become primitives that other programmers can benefit from. Thanks to code and data encapsulation (which data-oriented languages like Pig and Hive don't have), programmers can reuse existing ECL attributes without worrying about the internal implementation of those attributes. This makes ECL code easier to understand, more compact, and simplifies future development.

ECL is implicitly parallel, so the same ECL code developed to run on a one-node cluster can just as easily run on a cluster with thousands of nodes. The programmer doesn't need to worry about implementing parallelization, and ECL has an optimizer function that ensures the best performance for the specific hardware platform.

As mentioned above, ECL can eliminate or reduce code side effects, making ECL code more succinct, reliable, and easier to troubleshoot. ECL code composed of dozens of lines typically express algorithms that would require thousands if written using Java or Hadoop. And unlike imperative programming languages like Java, which allow (and even encourage) side effects and make programs that are hard to read, understand and debug, ECL code is easily readable, and the lack of side effects makes them easily verifiable.

ECL code can express algorithms that span across the entire data workflow. Unlike Pig and Hive, which were originally designed to write small code snippets, ECL provides for a mature programming paradigm that encourages collaboration, code reuse, encapsulation, extensibility, and readability.

ECL was designed from the ground up to be a data-oriented programming language. Unlike Java, high-level data primitives such as JOIN, TRANSFORM, PROJECT, SORT, DISTRIBUTE, MAP, NORMALIZE, etc. are first class functions, so basic data operations can be implemented in a single line of code. This makes ECL an ideal programming language for data analysis as it can be used to express data algorithms directly, removing the need to write the specifications software developers need to write their programs. In essence, because they use ECL, HPCC Systems data lakes need fewer programmers to deliver more projects in shorter amounts of time.

About Thor

Thor is a distributed data processing cluster technology that imports and processes data in bulk. To do this effectively at scale (i.e., billions of records), Thor supports data partitioning and parallel processing. For example, consider a file that contains people's names and property addresses. Now, you want to develop a program that imports this file and then appends the purchase value of each property. You can import this file into Thor, write code to resolve the addresses, link it to another dataset that contains the property values, and produce a file containing both the addresses and property values. Thor can execute this kind of logic with billions of records in just a few minutes.

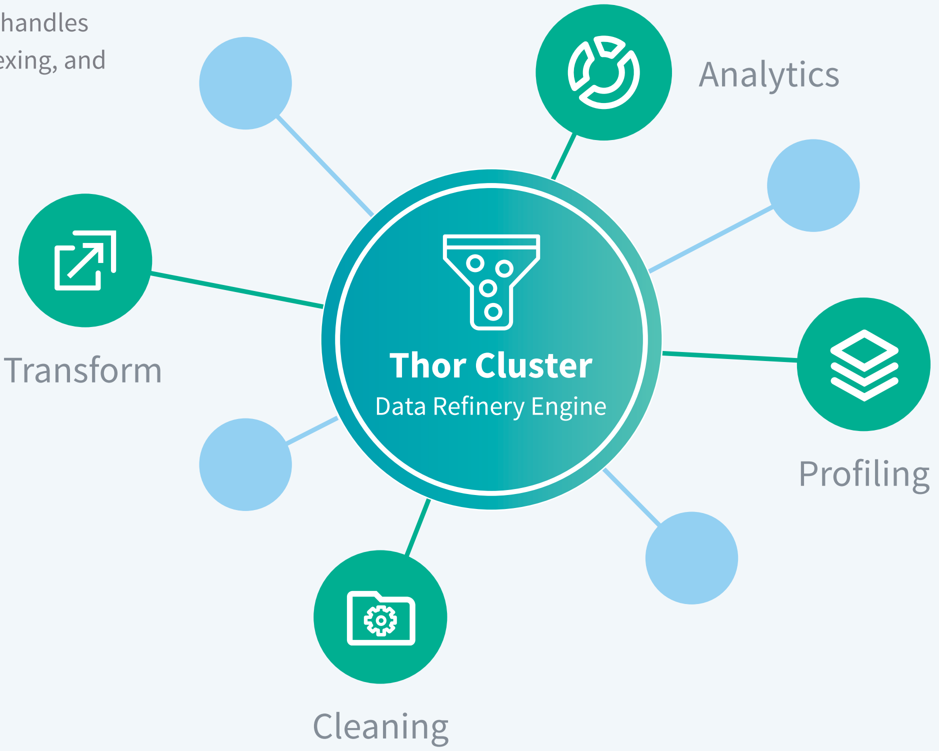


Illustration 5: A Thor cluster handles the formatting, cleaning, indexing, and enrichment of inbound data

The Thor cluster is based on a manager/worker design. In effect, you have one manager process overseeing many different worker processes. Each worker process processes one portion of an input data file, while the manager process acts as the delegator and coordinator of all worker processes. By dividing a large data file into multiple smaller parts, data processing can happen at a much faster rate as each of the worker processes are operating in parallel.

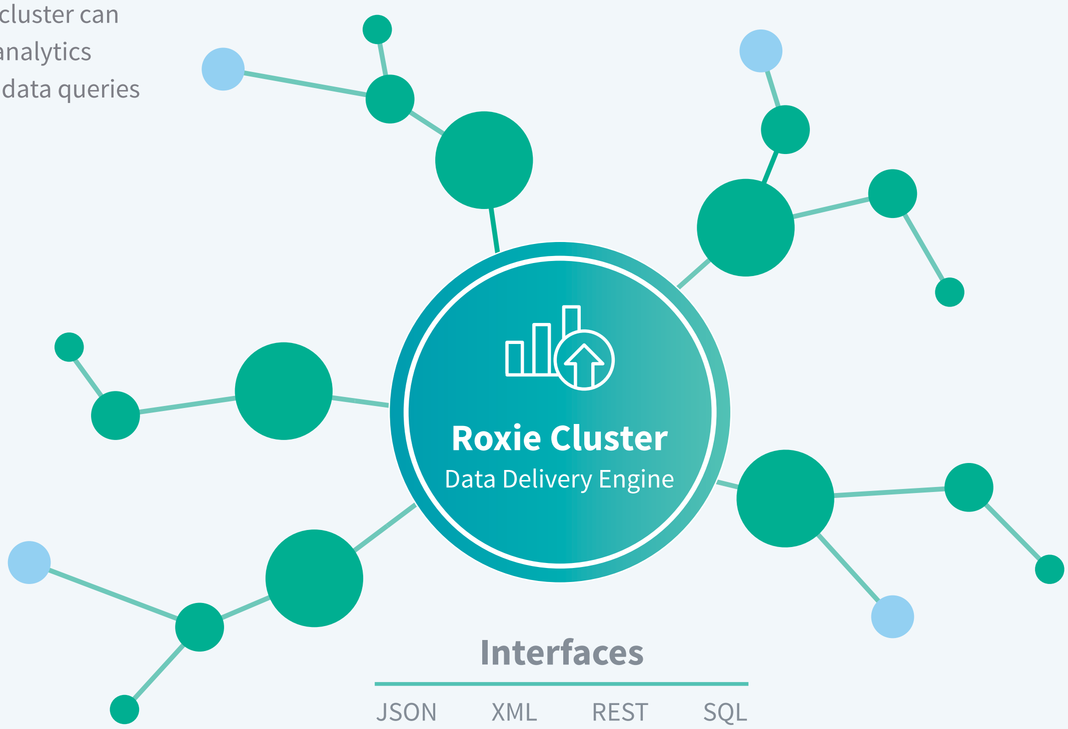
Manage data, more quickly and efficiently

So, when ECL code is submitted to Thor, the code is compiled into an execution graph that is then deployed to every worker. Every worker and their manager have this execution graph, so the steps in the execution graph are individually executed on each dataset partition assigned to each worker. Steps that need consolidation and coordination are executed by the manager. In essence, Thor gives programmers the ease-of-use characteristic of relational databases, but with the additional power of parallel data processing at scale, automatically.

About Roxie

While Thor acts as a bulk database server, it isn't optimized to be queried in a highly concurrent or real-time environment. Data queries on the HPCC Systems platform are handled by one or more Roxie clusters.

Illustration 6: A Roxie cluster can perform the real-time analytics required to respond to data queries



Thor’s principles are based on the manipulation of raw data files and a manager/worker design, which works well for bulk processing, but has limited capabilities for handling end-user queries. This is because those queries are inherently concurrent and need sub-second response times. Accordingly, Roxie leverages indexed data and employs a server/agent design. The Roxie workflow is:

1. Roxie **receives** the query request, then
2. **Identifies** which server can fulfill the query request
3. The designated server **messages** all the agents for the data, then
4. **Consolidates** the result data from the agents and returns the results to the calling process

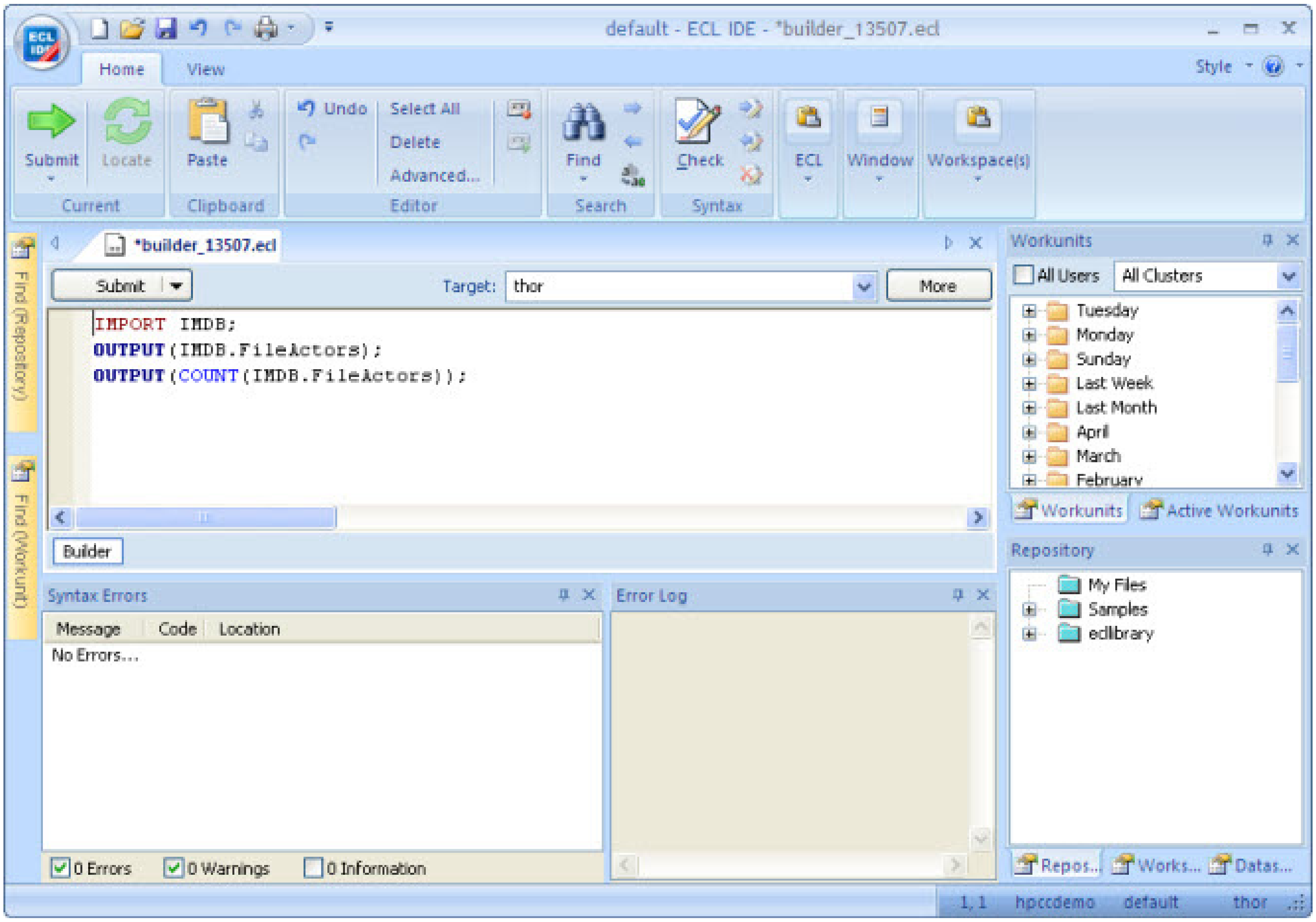
Perform real-time queries with highly concurrent delivery

A single Roxie cluster can have one or many server/agent combinations. Which combination is used depends on query workload and whether using one or multiple data partitions will make the processing efficient. APIs developed for Roxie are written in ECL and writing queries for Roxie is nearly identical to writing a query for Thor. Once the query is compiled, it is published to Roxie as an endpoint API.

Other Tools in the HPCC Systems Platform

It is important to have a process for the curation, visualization, organizing, and governing of data in the data lake. HPCC Systems has a suite of tools currently available that support these processes and new tools are in continuous development.

HPCC Systems provides an integrated development environment (IDE) for developers to facilitate ECL code development, called the ECL IDE. It is a Windows desktop application. There is also an ECL Language Extension available for VS Code that some developers prefer to use.



Target: thor

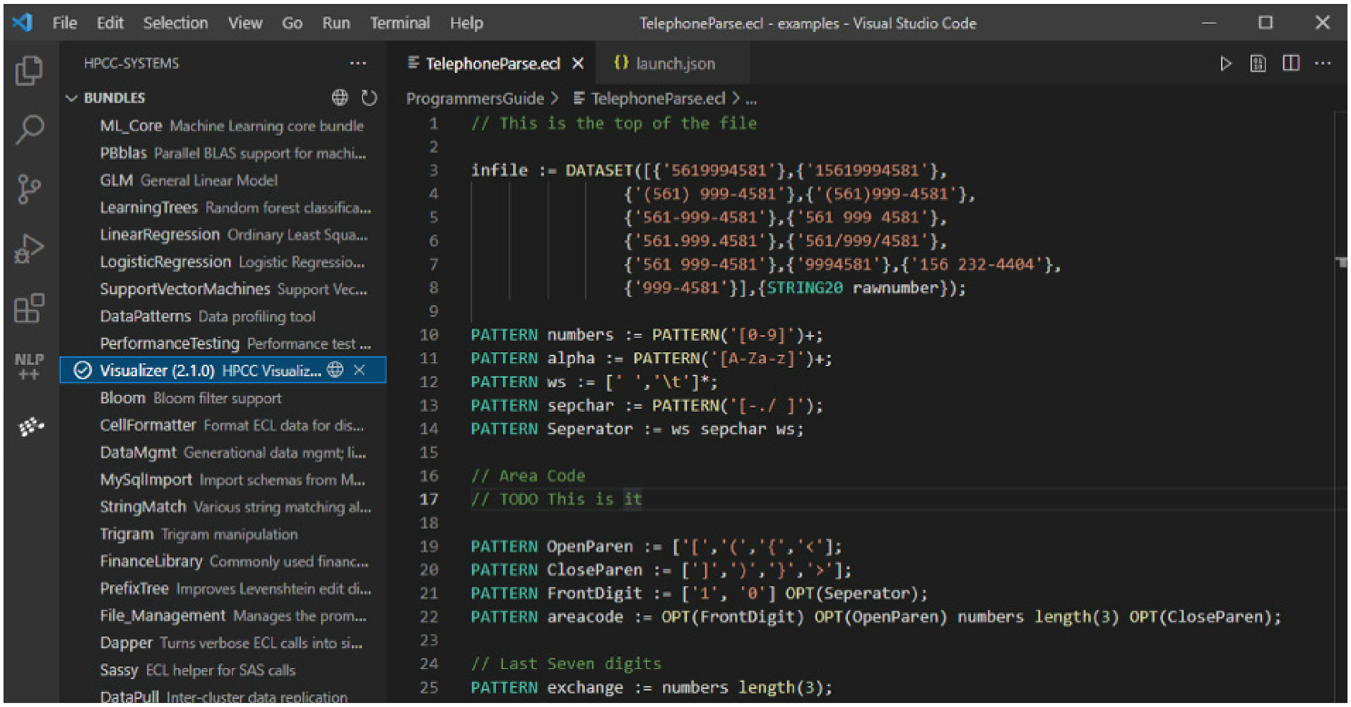
```
IMPORT IMDB;
OUTPUT(IMDB.FileActors);
OUTPUT(COUNT(IMDB.FileActors));
```

No Errors...

☑ 0 Errors ☑ 0 Warnings □ 0 Information

1, 1 hpccdemo default thor

Illustration 7: ECL integrated development environment (IDE)


File
Edit
Selection
View
Go
Run
Terminal
Help
TelephoneParse.ecl - examples - Visual Studio Code
HPCC-SYSTEMS
BUNDLES
ML_Core Machine Learning core bundle
PBblas Parallel BLAS support for machi...
GLM General Linear Model
LearningTrees Random forest classifica...
LinearRegression Ordinary Least Squa...
LogisticRegression Logistic Regressio...
SupportVectorMachines Support Vec...
DataPatterns Data profiling tool
PerformanceTesting Performance test ...
Visualizer (2.1.0) HPCC Visualiz...
Bloom Bloom filter support
CellFormatter Format ECL data for dis...
DataMgmt Generational data mgmt; li...
MySqlImport Import schemas from M...
StringMatch Various string matching al...
Trigram Trigram manipulation
FinanceLibrary Commonly used financ...
PrefixTree Improves Levenshtein edit di...
File_Management Manages the prom...
Dapper Turns verbose ECL calls into si...
Sassy ECL helper for SAS calls
DataPull Inter-cluster data replication
NLP ++
TelephoneParse.ecl
launch.json
ProgrammersGuide > TelephoneParse.ecl > ...

```
// This is the top of the file

infile := DATASET([{'5619994581'},{'15619994581'},
                    {'(561) 999-4581'},{'(561)999-4581'},
                    {'561-999-4581'},{'561 999 4581'},
                    {'561.999.4581'},{'561/999/4581'},
                    {'561 999-4581'},{'9994581'},{'156 232-4404'},
                    {'999-4581'}],{STRING20 rawnumber});

PATTERN numbers := PATTERN('[0-9]')+;
PATTERN alpha := PATTERN('[A-Za-z]')+;
PATTERN ws := [' ','\t']*;
PATTERN sepchar := PATTERN('[-./ ]');
PATTERN Seperator := ws sepchar ws;

// Area Code
// TODO This is it

PATTERN OpenParen := ['[','(','{','<'];
PATTERN CloseParen := [']',')','}','>'];
PATTERN FrontDigit := ['1', '0'] OPT(Seperator);
PATTERN areacode := OPT(FrontDigit) OPT(OpenParen) numbers length(3) OPT(CloseParen);

// Last Seven digits
PATTERN exchange := numbers length(3);
```

Illustration 8: VS Code IDE

Tombolo is the HPCC Systems platform’s tool for data curation, governance, and job orchestration. Data curation is an active and ongoing process of data management that turns independently created data sources into unified data sets ready for user consumption. By curating HPCC Systems data, Tombolo helps users keep an accurate catalog of data as it enters the data lake, is formatted and/or enriched, and as it’s used to inform queries implemented in Roxie.

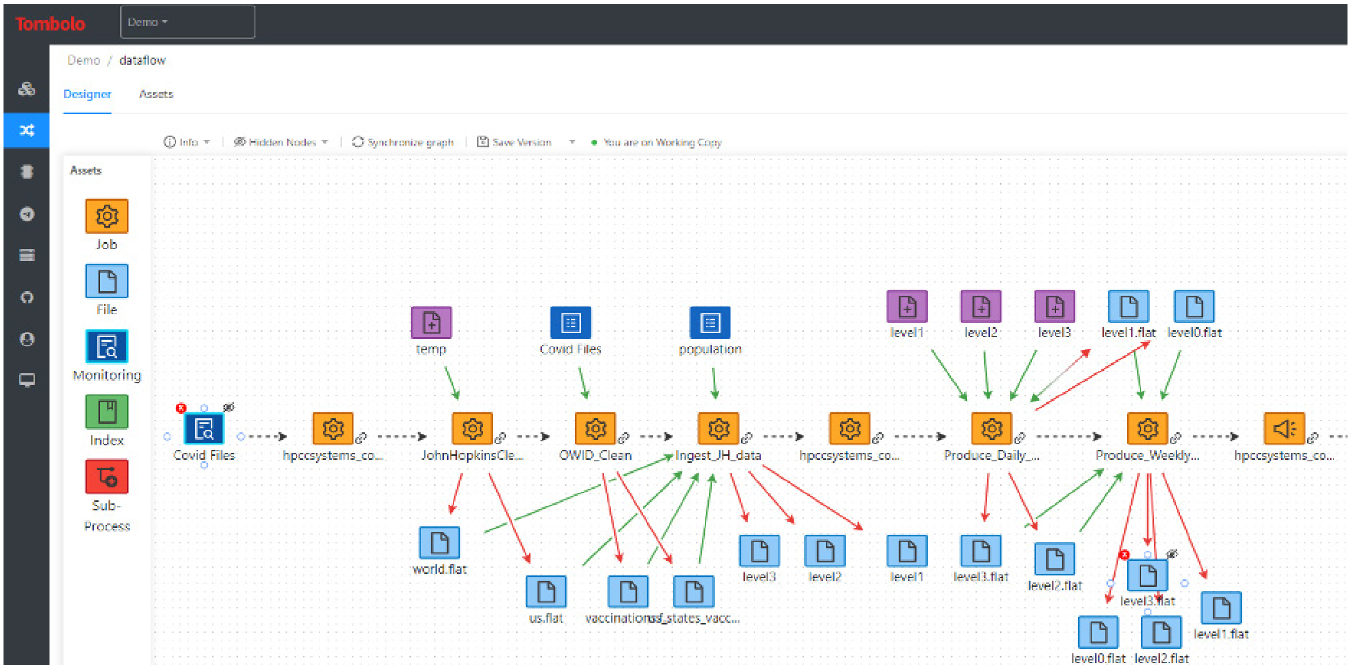
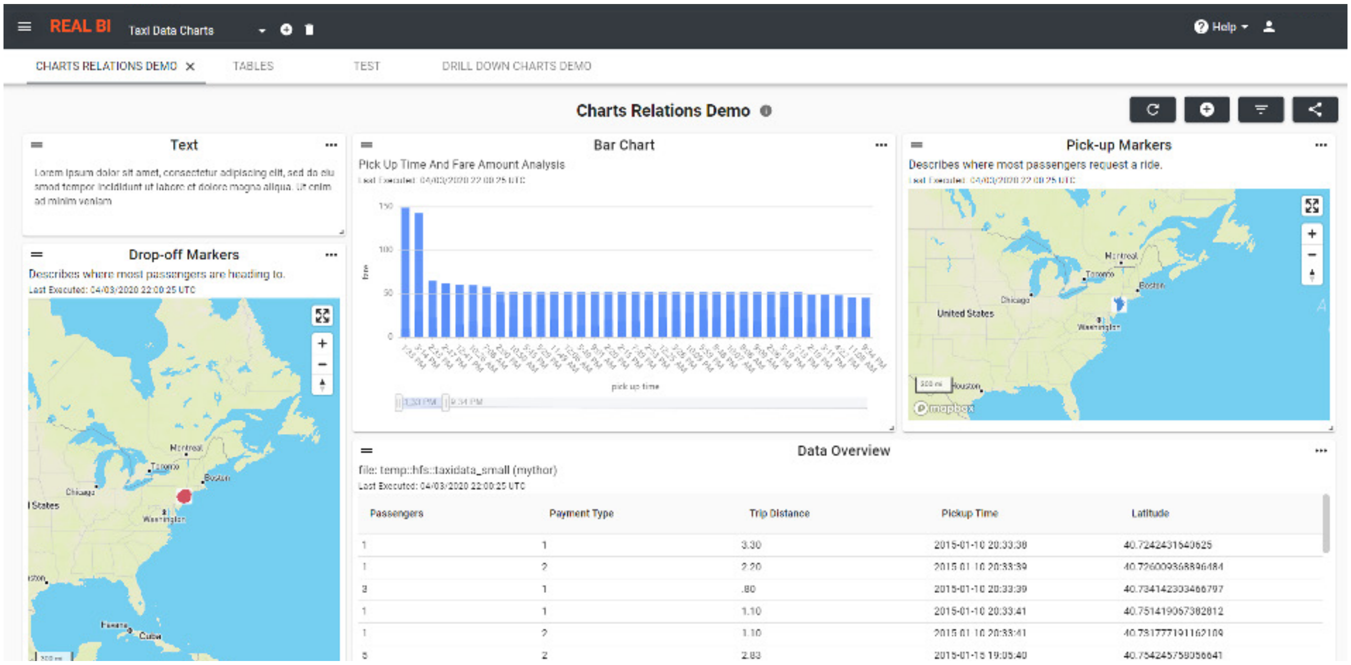


Illustration 9: Tombolo, the data curation tool for HPCC Systems

One of the options for Data Visualizations is Real BI, a customizable data dashboard application that can quickly pull needed data from the HPCC Systems environment to test query concepts and confirm data accuracy.



Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam

Pick Up Time And Fare Amount Analysis
Last Executed: 04/03/2020 22:00:25 UTC

Describes where most passengers request a ride.
Last Executed: 04/03/2020 22:00:25 UTC

Describes where most passengers are heading to.
Last Executed: 04/03/2020 22:00:25 UTC

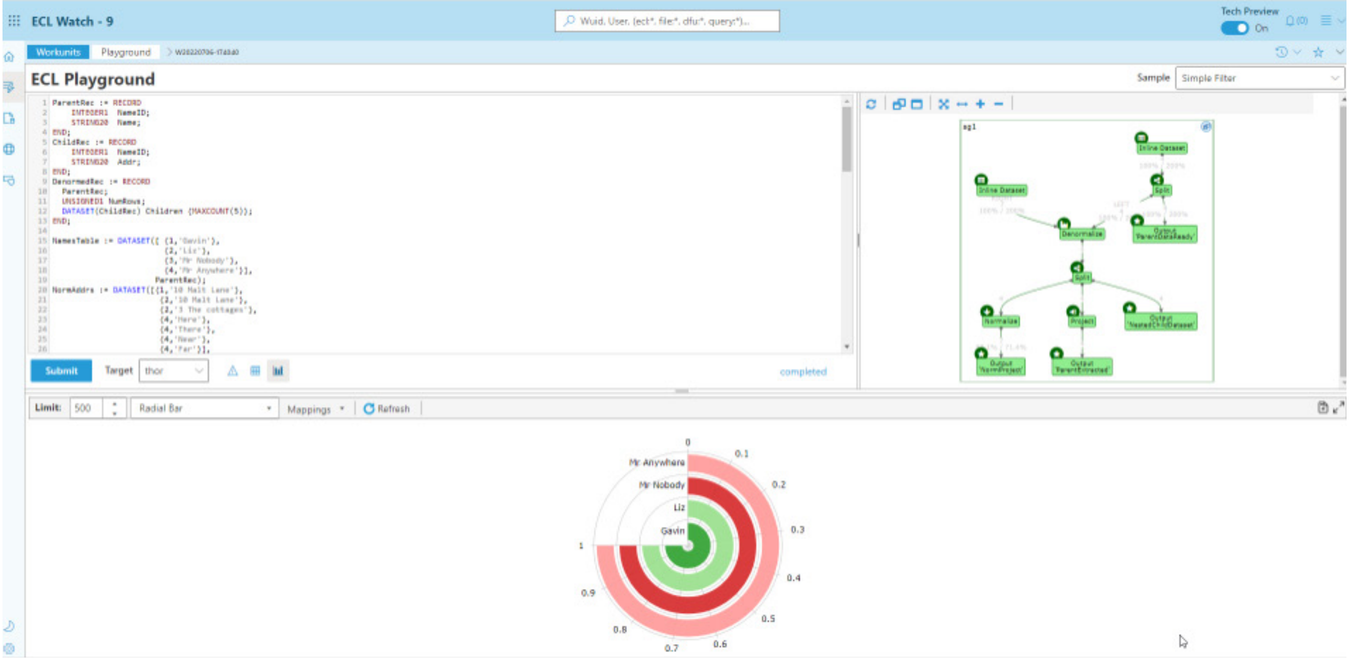
Data Overview

file: temp::hfs::taxidata_small (mythor)
Last Executed: 04/03/2020 22:00:25 UTC

Passengers	Payment Type	Trip Distance	Pickup Time	Latitude
1	1	3.30	2015-01-10 20:33:38	40.7242431640625
1	2	2.20	2015 01 10 20:33:39	40.726009368896484
3	1	.80	2015-01-10 20:33:39	40.734142303466797
1	1	1.10	2015-01-10 20:33:41	40.751419067382812
1	2	1.10	2015 01 10 20:33:41	40.731777191162109
5	2	2.83	2015-01-15 19:05:40	40.754245758056641

Illustration 10: A sample report created by Real BI

ECL Watch helps users monitor all aspects of an HPCC Systems cluster. It also provides a sandbox tool for testing ECL code prior to implementation in the data lake. The sandbox tool includes a sample dataset for code testing, and a dashboard reporting feature to visualize test results in graphical or relational layouts.

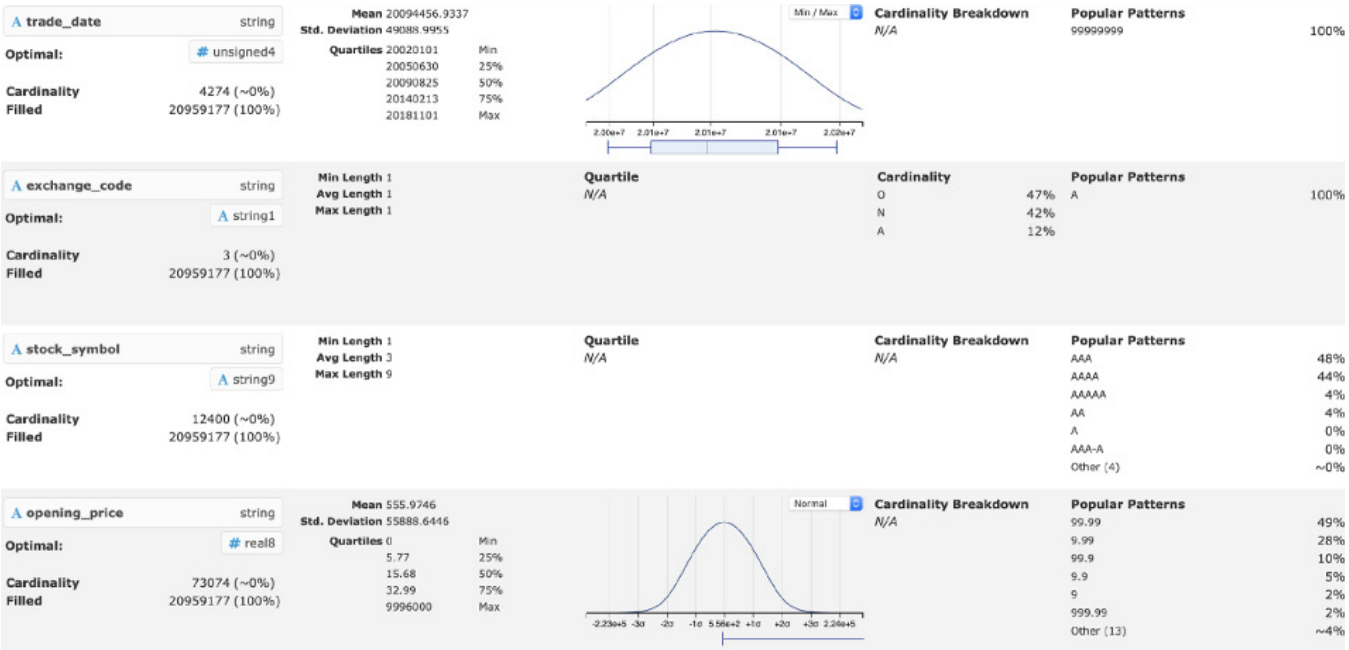


```
ParentRec := RECORD
    INTEGER1  NameID;
    STRING20  Name;
END;
ChildRec := RECORD
    INTEGER1  NameID;
    STRING20  Addr;
END;
DenormedRec := RECORD
  ParentRec;
  UNSIGNED1 NumRows;
  DATASET(ChildRec) Children {MAXCOUNT(5)};
END;

NamesTable := DATASET([ {1,'Gavin'},
                        {2,'Liz'},
                        {3,'Mr Nobody'},
                        {4,'Mr Anywhere'}],
                       ParentRec);
NormAddrs := DATASET([{1,'10 Malt Lane'},
                      {2,'10 Malt Lane'},
                      {2,'3 The cottages'},
                      {4,'Here'},
                      {4,'There'},
                      {4,'Near'},
                      {4,'Far'}],
```

Illustration 11: The ECL Watch digital sandbox lets users test new ECL code before implementing it in the data lake.

Data Profiling provides data profiling and research tools to an ECL programmer, a wealth of information regarding the "shape" of a dataset from a simple function call and detailed metrics per dataset field.



A trade_date string
Optimal: # unsigned4
Cardinality 4274 (~0%)
Filled 20959177 (100%)

Mean 20094456.9337
Std. Deviation 49088.9955
Quartiles 20020101 Min
20050630 25%
20090825 50%
20140213 75%
20181101 Max

Cardinality Breakdown
N/A

Popular Patterns
99999999 100%

A exchange_code string
Optimal: A string1
Cardinality 3 (~0%)
Filled 20959177 (100%)

Min Length 1
Avg Length 1
Max Length 1

Quartile
N/A

Cardinality
O 47%
N 42%
A 12%

Popular Patterns
A 100%

A stock_symbol string
Optimal: A string9
Cardinality 12400 (~0%)
Filled 20959177 (100%)

Min Length 1
Avg Length 3
Max Length 9

Quartile
N/A

Cardinality Breakdown
N/A

Popular Patterns
AAA 48%
AAAA 44%
AAAAA 4%
AA 4%
A 0%
AAA-A 0%
Other (4) ~0%

A opening_price string
Optimal: # real8
Cardinality 73074 (~0%)
Filled 20959177 (100%)

Mean 555.9746
Std. Deviation 55888.6446
Quartiles 0 Min
5.77 25%
15.68 50%
32.99 75%
9996000 Max

Cardinality Breakdown
N/A

Popular Patterns
99.99 49%
9.99 28%
99.9 10%
9.9 5%
9 2%
999.99 2%
Other (13) ~4%

Illustration 12: Data Profiling

Data governance is the process of implementing a set of rules and policies to manage and protect potentially sensitive data, like credit card numbers or medical records. As sensitive data must often remain private and protected to meet various legal and service level agreement (SLA) requirements, future releases of the HPCC Systems platform will support data governance capabilities that can:

- Assign accountability to employees responsible for data assets
- Grant or restrict data access per usage and/or by need
- Implement data privacy and data protection protocols
- Provide reports that track the lifecycle of all sensitive data in a data lake for auditing and compliance purposes

HPCC Systems Training and Support

HPCC Systems offers a variety of training and support options to customers interested in applying HPCC Systems to their own data management and analytics needs. Training is offered in a variety of formats including online and in-person as well as multiple day workshops. Most online training is available on demand and most courses are available for free to the open source and academic communities. Learning paths for ECL Core, Machine Learning, Administration, and Managers can be found online at hpccsystems.com. On the ‘Training’ page of the HPCC Systems website, you will also find links to short video tutorials, a wiki with ECL training resources and Tips and Tricks to get you up to speed quickly.

As a fully open source project, HPCC Systems is supported by a worldwide network of developers working either directly for HPCC Systems or as a member of the HPCC Systems community. There are an estimated 2,000 ECL developers globally that are actively writing code for the HPCC Systems platform. Further, HPCC Systems has been leveraged globally across academia through research, student internships, and events. Many of these developers attend the HPCC Systems annual community event, where developers from around the world meet to learn more about HPCC Systems and to be inspired by how other developers use the platform.

For more information, call 877.316.9669 or visit www.hpccsystems.com

HPCC™ SYSTEMS

® LexisNexis®
RISK SOLUTIONS

About HPCC Systems®

HPCC Systems® from LexisNexis® Risk Solutions is a proven, comprehensive, dedicated data lake platform that makes combining different types of data easier and faster than competing platforms — even data stored in massive, mixed schema data lakes . It’s also open source, free to use, and easy to learn. You can acquire, enrich, deliver and curate information faster using HPCC Systems — and the automation of Kubernetes in our cloud-native architecture makes it easy to set-up, manage and scale your data to save time and money, now and in the future. HPCC Systems offers a consistent data-centric programming language, two processing platforms and a single, complete end-to-end architecture for efficient processing. To learn more, visit us at hpccsystems.com.

About LexisNexis® Risk Solutions

LexisNexis® Risk Solutions harnesses the power of data and advanced analytics to provide insights that help businesses and governmental entities reduce risk and improve decisions to benefit people around the globe. We provide data and technology solutions for a wide range of industries including insurance, financial services, healthcare and government. Headquartered in metro Atlanta, Georgia, we have offices throughout the world and are part of RELX (LSE: REL/NYSE: RELX), a global provider of information-based analytics and decision tools for professional and business customers. For more information, please visit www.risk.lexisnexis.com and www.relx.com.