

# Solving Big Data Problems with the Open Source HPC Systems Platform

Invited talk at Wright State University

Presenter: Dr. John Holt  
Sr Architect

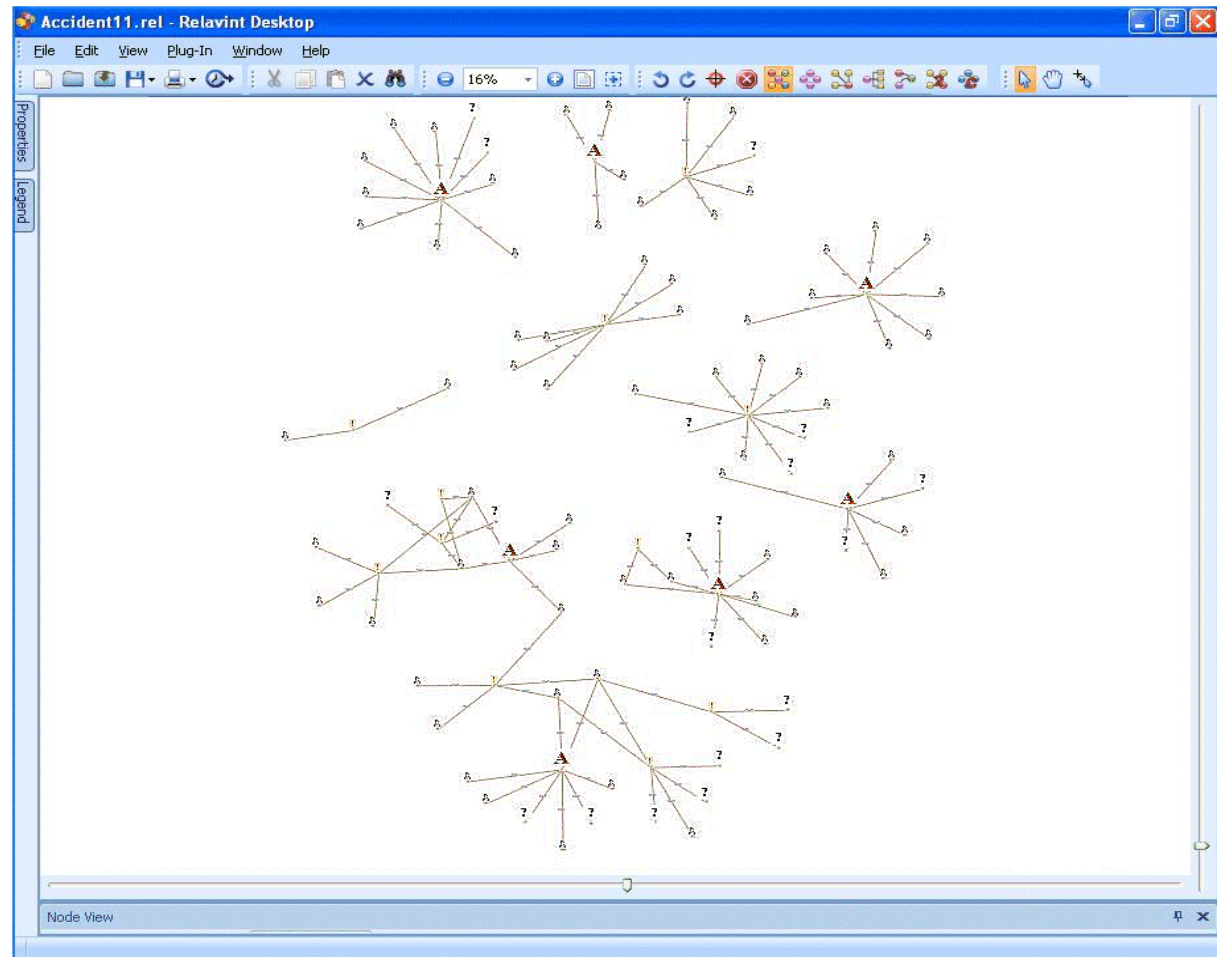


[hpcsystems.com](http://hpcsystems.com)

## Scenario

This view of carrier data shows seven known fraud claims and an additional linked claim.

The Insurance company data **only** finds a connection between two of the seven claims, and only identified one other claim as being weakly connected.



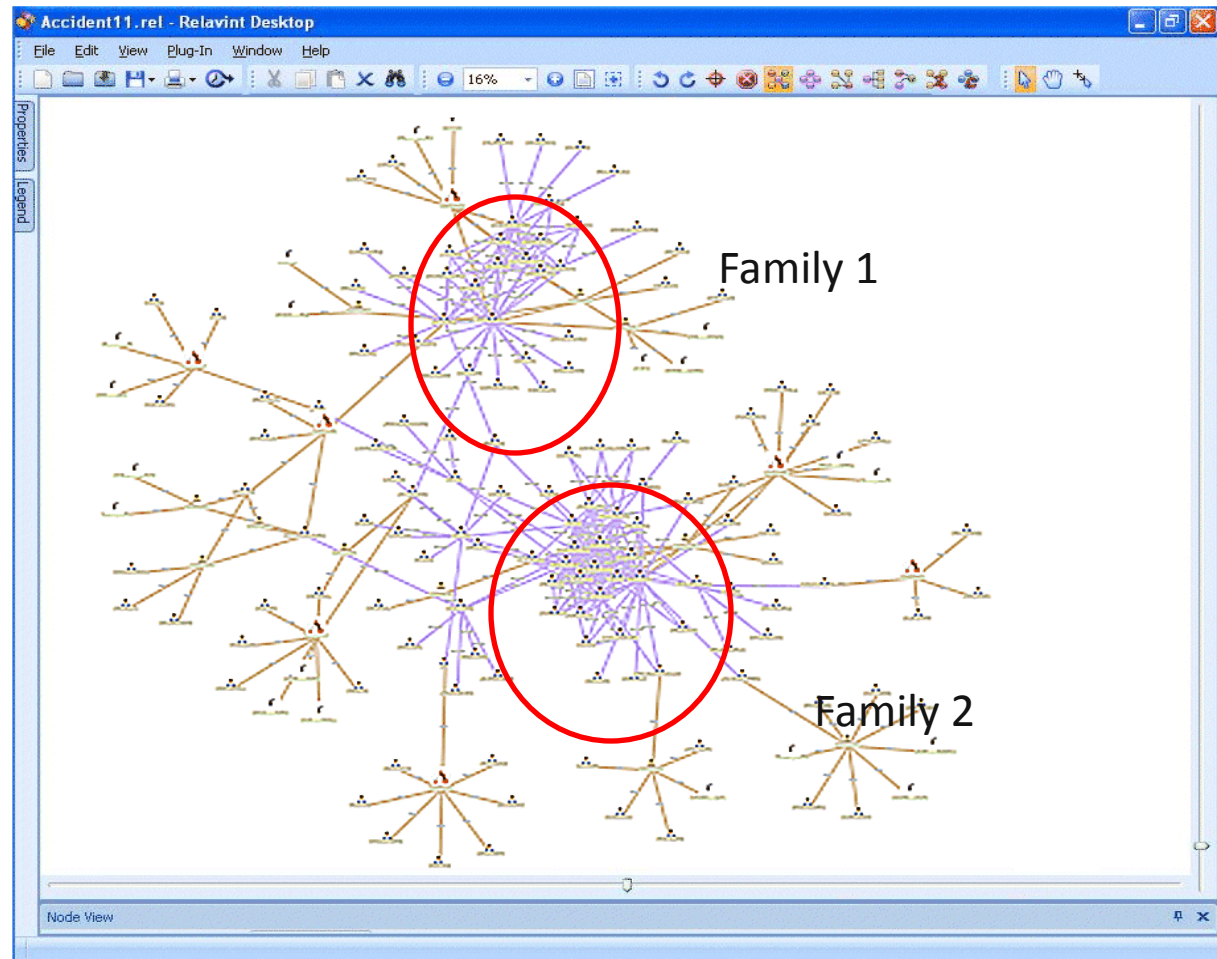
## Task

After adding the LexID to the carrier Data, LexisNexis HPCC technology then explored 2 additional degrees of relative separation

## Result

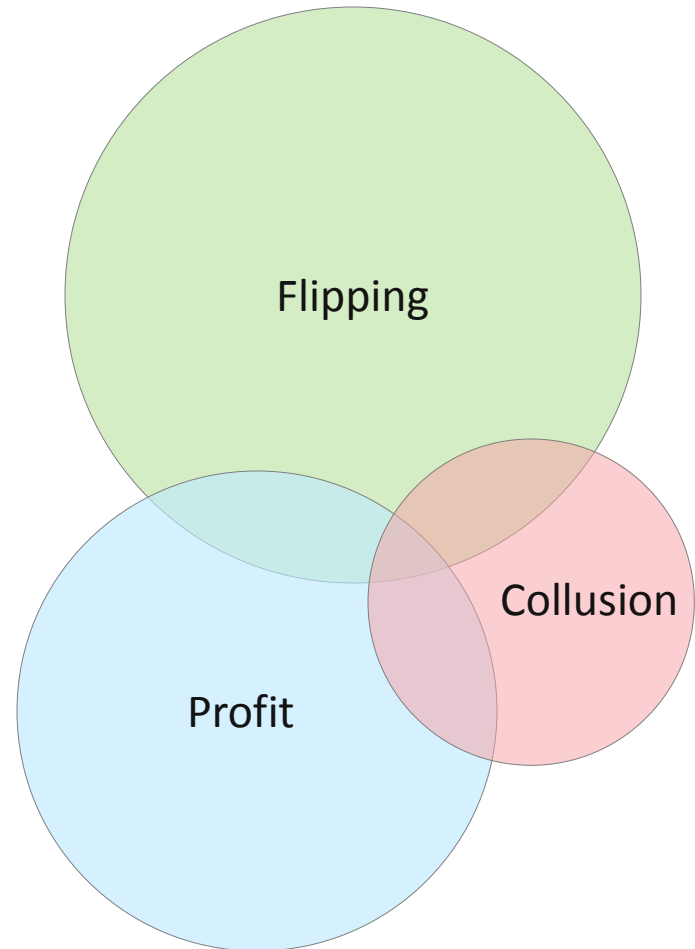
The results showed **two family groups interconnected on all of these seven claims.**

The links were much stronger than the carrier data previously supported.

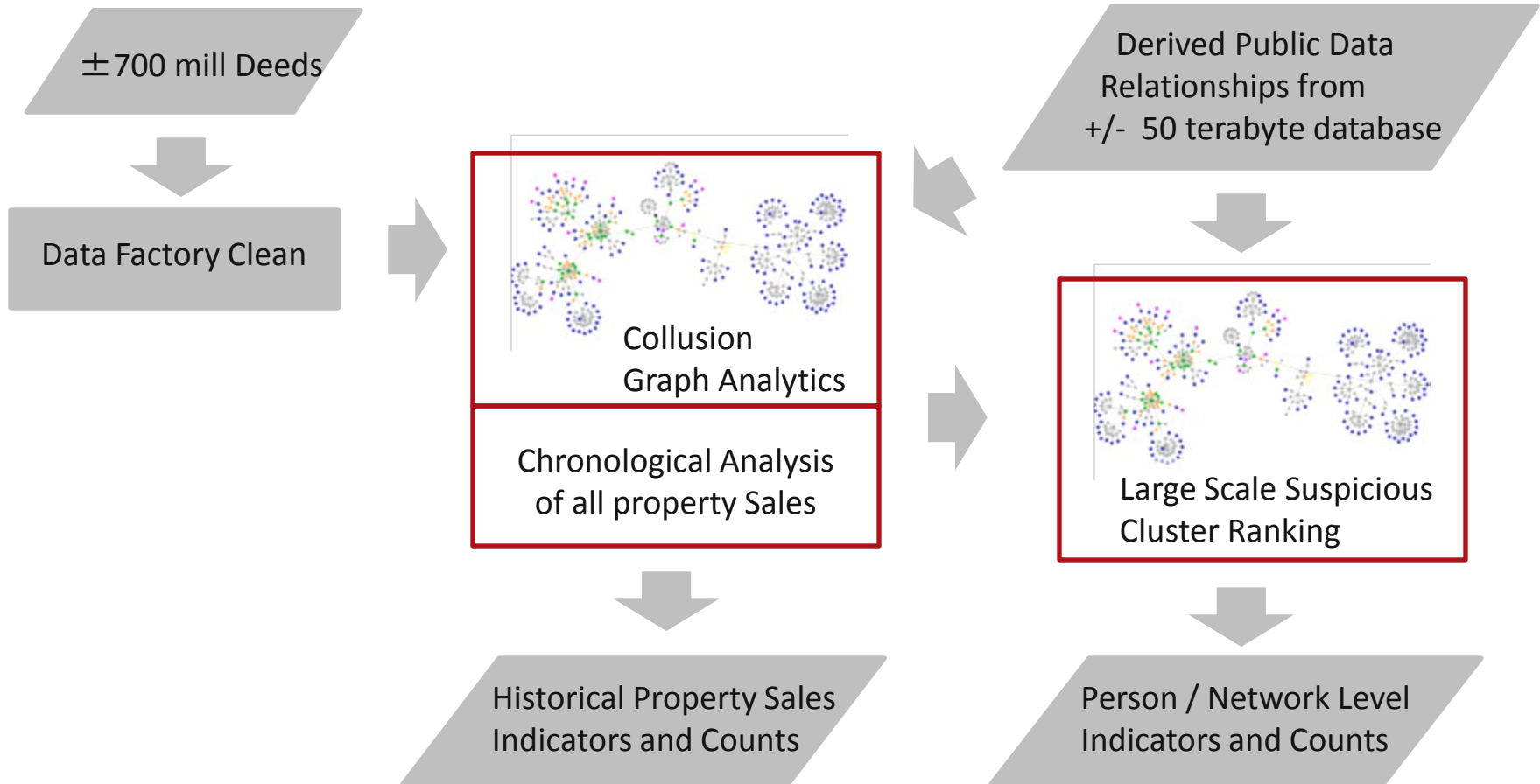


## Three core transaction variables measured

- Velocity
- Profit (or not)
- Buyer to Seller Relationship Distance  
(Potential of Collusion)

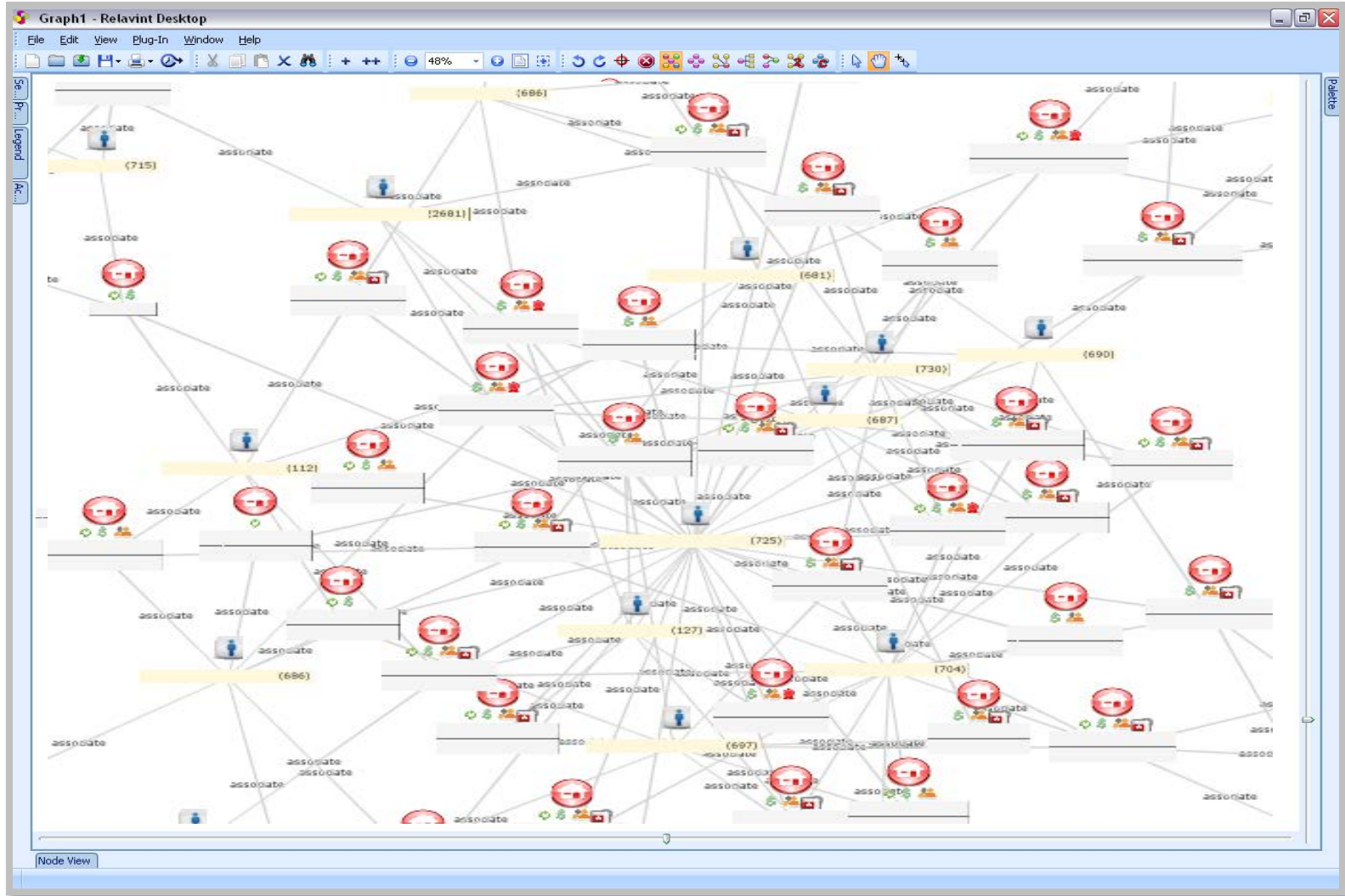


# Property Transaction Risk





# Suspicious Equity Stripping Cluster



## Large scale measurement of influencers strategically placed to potentially direct suspicious transactions.

- All BIG DATA on one supercomputer measuring over a decade of property transfers nationwide.
- BIG DATA Products to turn other BIG DATA into compelling intelligence.
- Large Scale Graph Analytics allow for identifying known unknowns.
- Florida Proof of Concept
  - Highest ranked influencers
    - Identified known ringleaders in flipping and equity stripping schemes.
    - Typically not connected directly to suspicious transactions.
  - Known ringleaders not the Highest Ranking.
- Clusters with high levels of potential collusion.
- Clusters offloading property, generating defaults.
- Agile Framework able to keep step with emerging schemes in real estate.



## Scenario

Healthcare insurers need better analytics to identify drug seeking behavior and schemes that recruit members to use their membership fraudulently.

Groups of people collude to source schedule drugs through multiple members to avoid being detected by rules based systems.

Providers recruit members to provide and escalate services that are not rendered.

## Task

Given a large set of prescriptions. Calculate normal social distributions of each brand and detect where there is an unusual socialization of prescriptions and services.

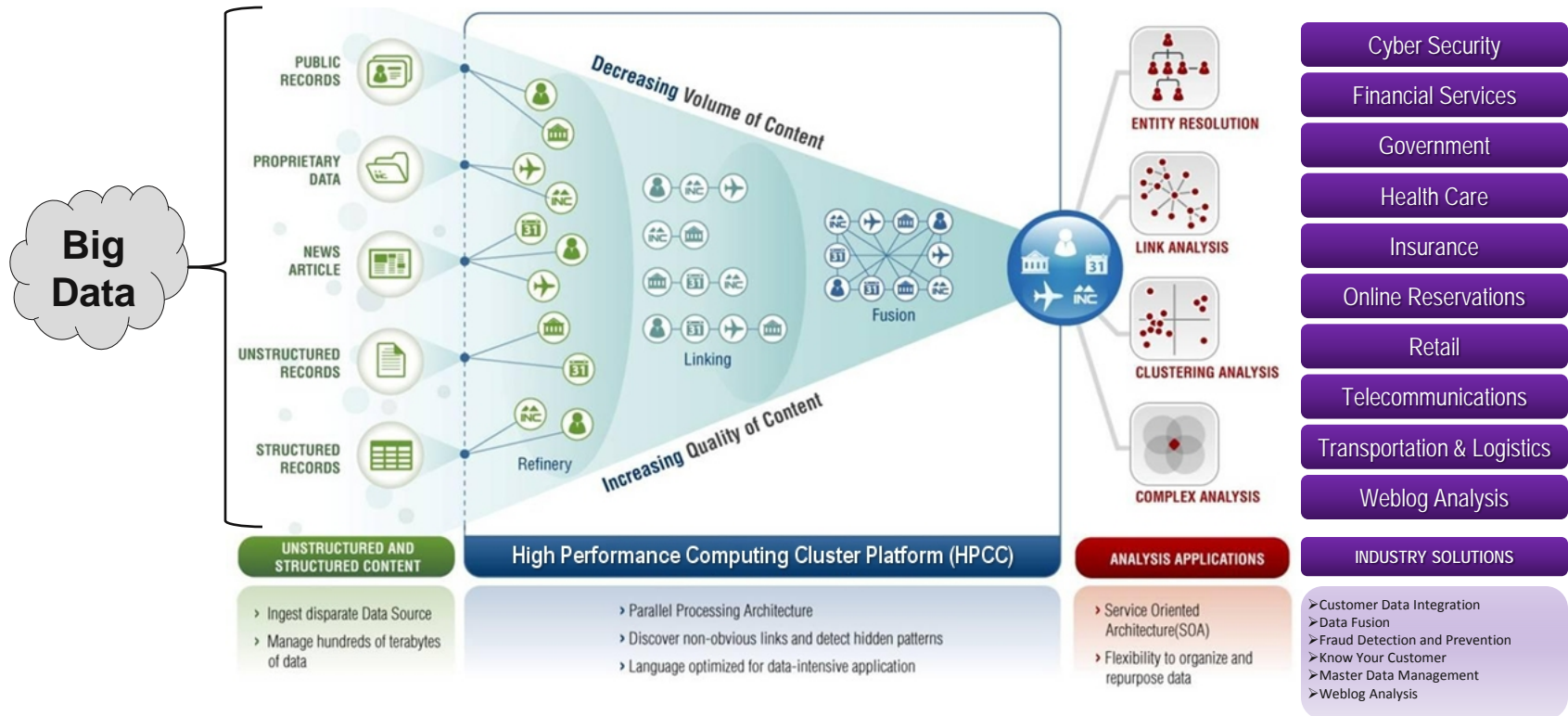
## Result

The analysis detected social groups that are sourcing Vicodin and other schedule drugs. Identifies prescribers and pharmacies involved to help the insurer focus investigations and intervene strategically to mitigate risk.





# The Data/Information flow



- **High Performance Computing Cluster Platform (HPCC)** enables data integration on a scale not previously available and real-time answers to millions of users. Built for big data and proven for 10 years with enterprise customers.
- **Offers a single architecture**, two data platforms (query and refinery) and a consistent data-intensive programming language (ECL)
- **ECL Parallel Programming Language** optimized for business differentiating data intensive applications

- Open Source distributed data-intensive computing platform
- Share-nothing architecture
- Runs on commodity computing/storage nodes
- Binary packages available for the most common Linux distributions
- Provides end-to-end Big Data workflow management , scheduler, integration tools, etc.
- Originally designed circa 1999 (predates the original paper on MapReduce from Dec. '04)
- Improved over a decade of real-world Big Data analytics
- In use across critical production environments throughout LexisNexis for more than 10 years

- The HPCC Systems platform includes:
  - Thor: batch oriented data manipulation, linking and analytics engine
  - Roxie: real-time data delivery and analytics engine
- A high level declarative dataflow language: ECL
  - Implicitly parallel
  - No side effects
  - Code/data encapsulation
  - Extensible
  - Highly optimized
  - Builds graphical execution plans
  - Compiles into C++ and native machine code
  - Common to Thor and Roxie
- An extensive library of ECL modules, including data profiling, linking and Machine Learning

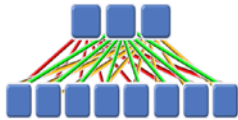
# The Three HPCC components

## 1 HPCC Data Refinery (Thor)



- Massively Parallel data processing engine
- Enables data integration on a scale not previously available
- Programmable using ECL

## 2 HPCC Data Delivery Engine (Roxie)



- A massively parallel, high throughput, query engine
- Low latency, highly concurrent and highly available
- Several advanced strategies for efficient retrieval
- Programmable using ECL

## 3 Enterprise Control Language (ECL)

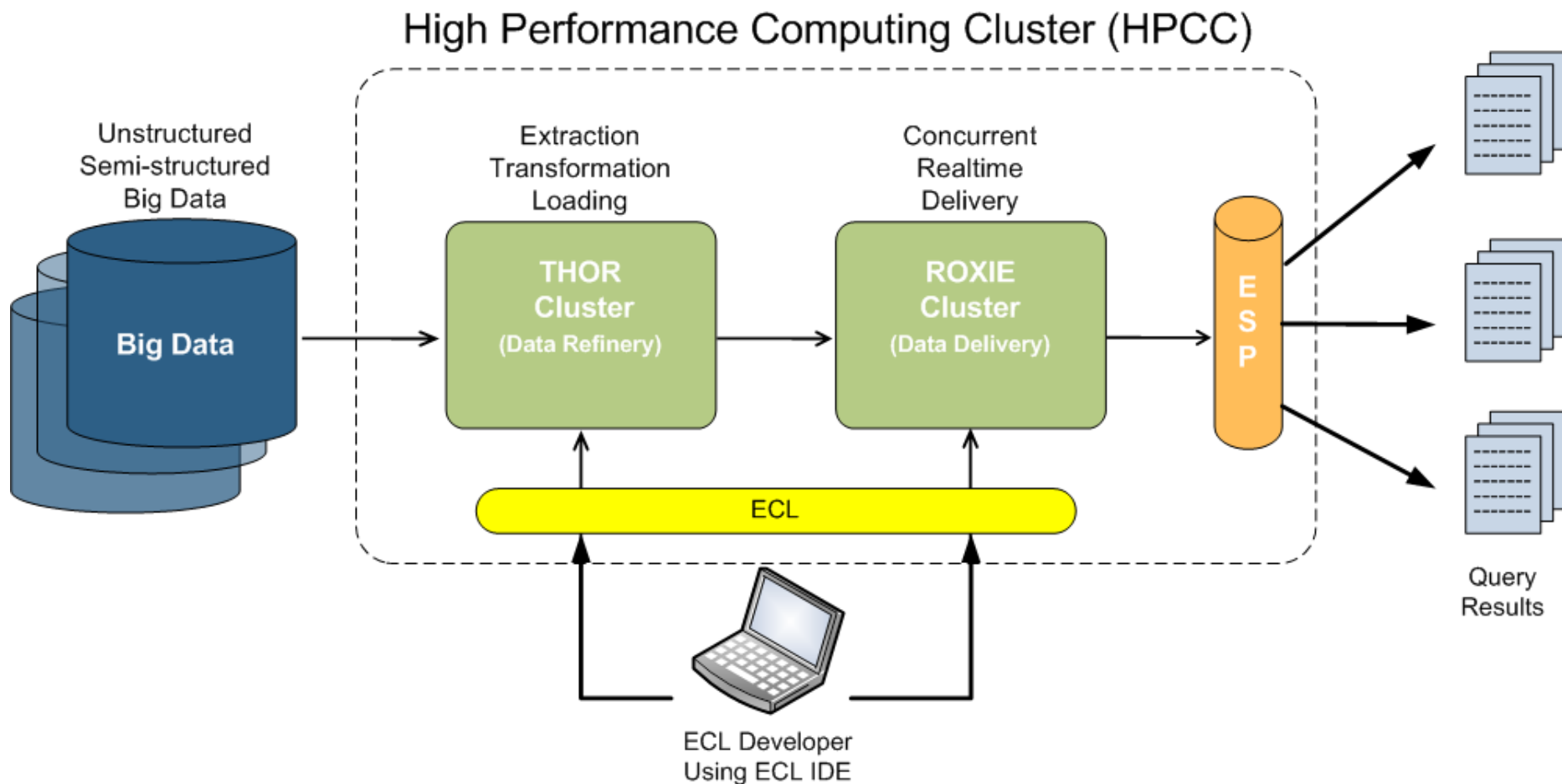


- An easy to use, declarative data-centric programming language optimized for large-scale data management and query processing
- Highly efficient; automatically distributes workload across all nodes; compiles to native machine code.
- Automatic parallelization and synchronization

## Conclusion: End to End platform

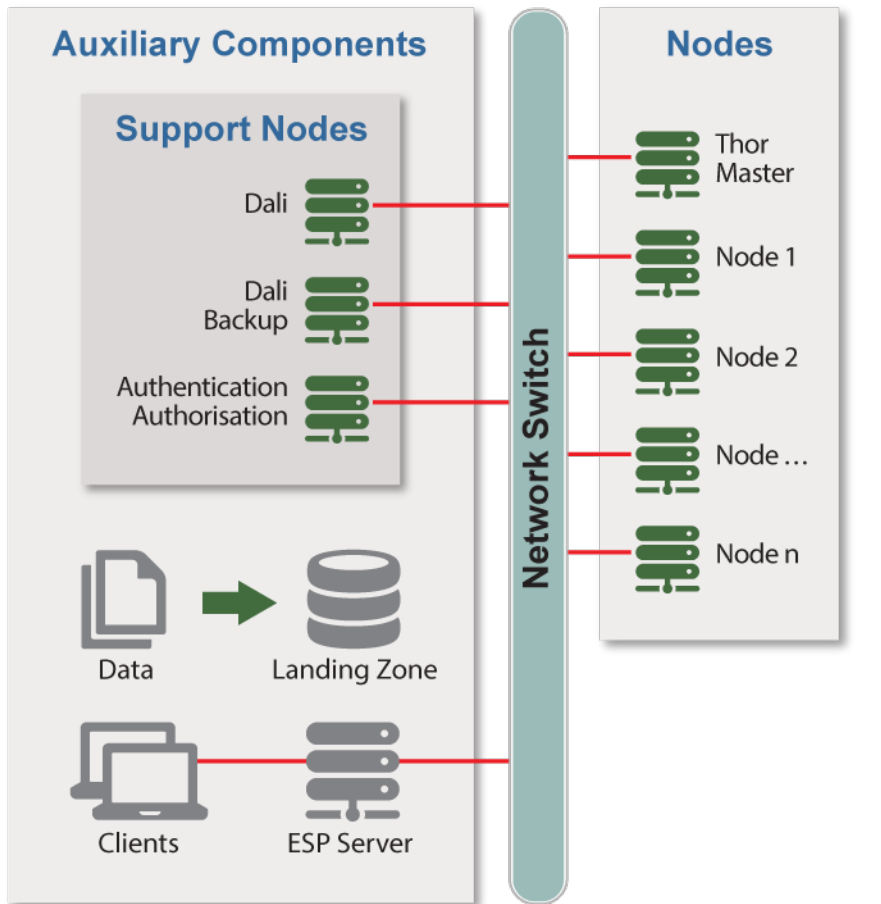
- No need for any third party tools

# The HPCC Systems platform



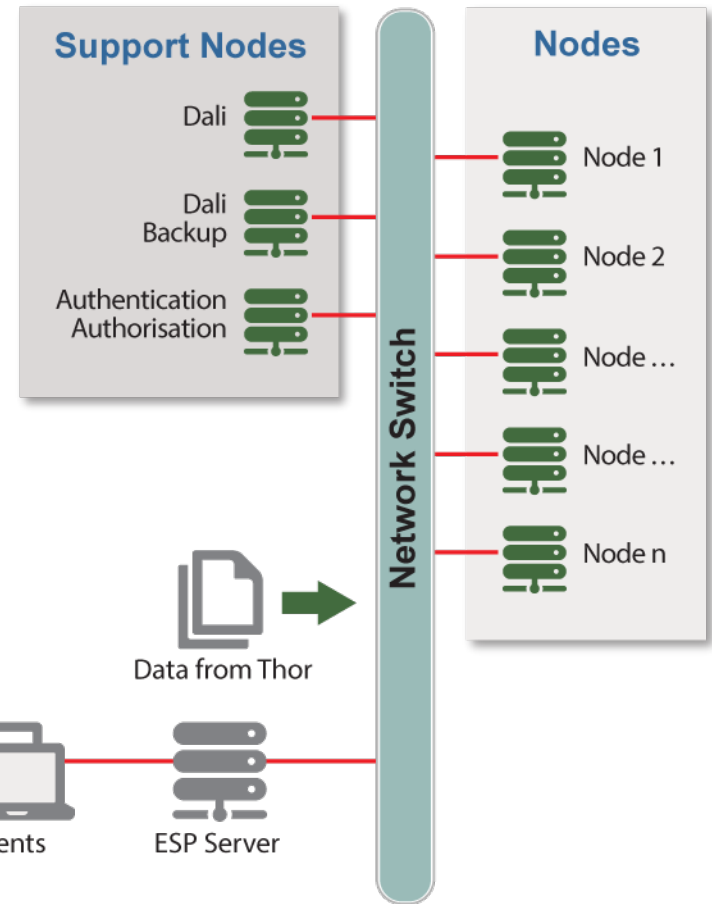


# Detailed HPCC Architecture



## Thor

(Batch Job Execution Engine + DFS)  
Physical Layout Schematic Diagram



## Roxie

(Rapid Data Delivery Engine)  
Physical Layout Schematic Diagram

# Enterprise Control Language (ECL)

- Declarative programming language:** Describe **what** needs to be done and **not how** to do it
- Powerful:** High level data activities like JOIN, TRANSFORM, PROJECT, SORT, DISTRIBUTE, MAP, etc. are available.
- Extensible:** Modular and extensible, it can shape itself to adapt to the type of problem at hand
- Implicitly parallel:** Parallelism is built into the underlying platform. The programmer needs not be concerned with data partitioning and parallelism
- Maintainable:** High level programming language, without side effects and with efficient encapsulation; programs are more succinct, reliable and easier to troubleshoot
- Complete:** ECL provides a complete data programming paradigm
- Homogeneous:** One language to express data algorithms across the entire HPCC platform: data integration, analytics and high speed delivery

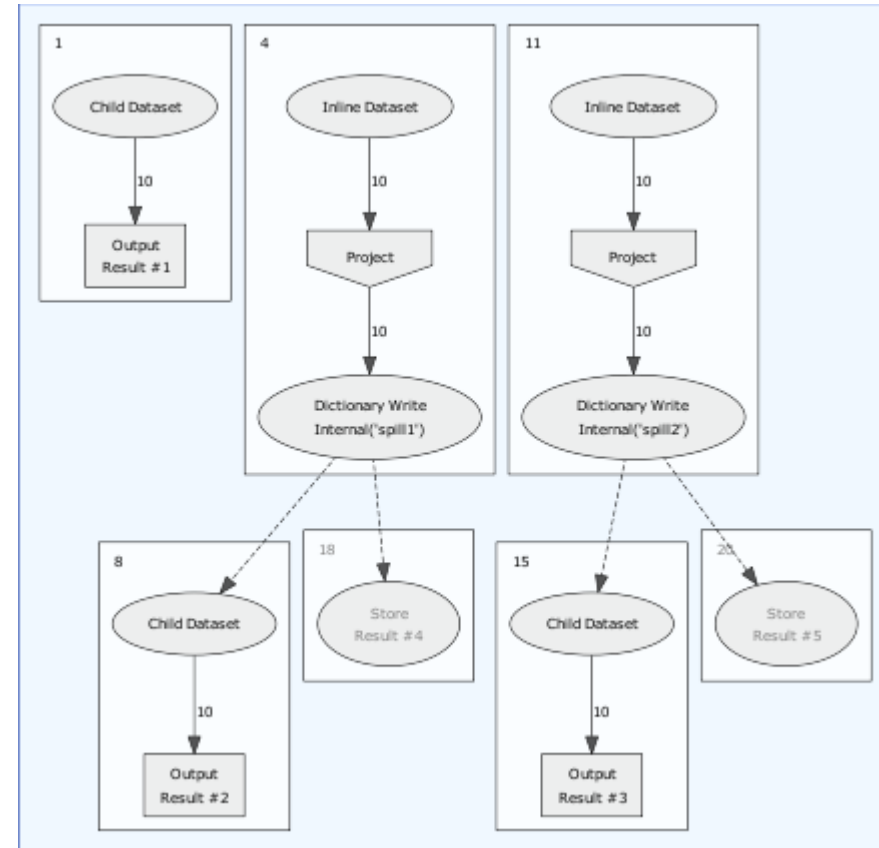
```
// Initialize output log
log_out_init := project(log_init,
                        transform(layout_logout,
                                self := left,
                                self := []));

// Create error log
outerrorfile := join(log_seq,
                    log_out_init,
                    left.linenum = right.linenum,
                    transform(recordof(log_seq),
                                self := left),
                    left only,
                    hash);

// Denormalize key value pairs
outlogfile := sort(denormalize(distribute(log_seq),
                              sort(distribute(key_val,
                                                left.linenum = right.linenum,
                                                transform(layout_logout,
                                                            self.keyvals := left,
                                                            row({right.linenum,
                                                            self := left}),
                                                            self := left)),
                              left.linenum = right.linenum,
                              transform(layout_logout,
                                          self.keyvals := left,
                                          row({right.linenum,
                                          self := left})),
                              left only,
                              hash);
```

# Enterprise Control Language (ECL)

- The ECL optimizer generates an optimal execution plan
- Sub-graphs run activities in parallel, automatically persist the results and spill to disk as needed
- Lazy evaluation avoids re-computing results when data and code haven't changed
- Graphs are dynamic and display number of records traversing the edges and data skew percentages



- Extensible Machine Learning Library developed in ECL
- Fully distributed across the cluster
- Supports PB-BLAS (Parallel Block BLAS)
- General statistical functions
- Supports supervised, semi-supervised and unsupervised learning methods
- Document manipulation, tokenization and statistical Natural Language Processing
- A consistent and standard interface to classification (“pluggable classifiers”)
- Efficient handling of iterative algorithms (for example, for numeric optimization and clustering)
- Open Source and available at: <http://hpccsystems.com/ml>

## General aspects

- Based on a distributed ECL linear algebra framework
- New algorithms can be quickly developed and implemented
- Common interface to classification (pluggable classifiers)

## ML algorithms

- Linear regression
- Several Classifiers
- Multiple clustering methods
- Association analysis

## Document manipulation and statistical grammar-free NLP

- Tokenization
- CoLocation

## Statistics

- General statistical methods
- Correlation
- Cardinality
- Ranking



## Linear Algebra library

- Support for sparse matrices
- Standard underlying matrix/vector data structures
- Basic operations (addition, products, transpositions)
- Determinant/inversions
- Factorization/SVD/Cholesky/Rank/UL
- PCA
- Eigenvectors/Eigenvalues
- Interpolation (Lanczos)
- Identity
- Covariance
- KD Trees

- ML on a general-purpose Big Data platform means effective analytics in-situ
- The combination of Thor and Roxie is ideal when, for example, training a model on massive amounts of labeled historical records (Thor), and providing real-time classification for new unlabeled data (Roxie)

## REMEMBER!

**When applying Machine Learning methods to Big Data: data profiling, parsing, cleansing, normalization, standardization and feature extraction represent 85% of the problem!**

## **KEL: an abstraction for network/graph processing**

- Declarative model: describe what things are, rather than how to execute
- High level: vertices and edges are first class citizens
- A single model to describe graphs and queries
- Leverages Thor for heavy lifting and Roxie for real-time analytics
- Compiles into ECL (and ECL compiles into C++, which compiles into assembler)

<b>ECL</b>	<b>KEL</b>
Generates C++ (1->100)	Generates ECL (1->12)
Files and Records	Entities and associations
Detailed control of data format	Loose control of input format; none of processing
Can write graph and statistical algorithms	Major algorithms built in
Thor/Roxie split by human design	Thor/Roxie split by system design
Solid, reliable and mature	R&D

- Actor := ENTITY( FLAT(UID(ActorName),Actor=ActorName) )
- Movie := ENTITY( FLAT(UID(MovieName),Title=MovieName) )
- Appearance := ASSOCIATION( FLAT(Actor Who,Movie What) )
  
- USE IMDB.File\_Actors(FLAT,Actor,Movie,Appearance)
  
- CoStar := ASSOCIATION( FLAT(Actor Who,Actor WhoElse) )
  
- GLOBAL: Appearance(#1,#2) Appearance(#3,#2) => CoStar(#1,#3)
  
- QUERY:FindActors(\_Actor) <= Actor(\_Actor)
- QUERY:FindMovies(\_Actor) <= Movie(UID IN Appearance(Who IN Actor(\_Actor){UID}){What})
- QUERY:FindCostars(\_Actor) <= Actor(UID IN CoStar(Who IN Actor(\_Actor){UID}){WhoElse})
- QUERY:FindAll(\_Actor) <= Actor(\_Actor),Movie(UID IN Appearance(Who IN \_1{UID}){What}),Actor(UID IN CoStar(Who IN \_1{UID}){WhoElse})

# Assorted examples

- Data integration using HPCC Flow
- Record linkage, entity disambiguation, entity resolution using SALT
- Data Visualization
- Exploratory data analysis
- Recommendation systems
- Search and retrieval
- Full text classifiers
- Scoring/statistical models
- Attribute manager
- Sentiment Analysis example
- Graph traversal problem handling (IMDB/Kevin Bacon example)
- Smart View
- SQL/JDBC integration
- Data Streaming/Kafka
- R integration
- Java integration
- Python integration



---

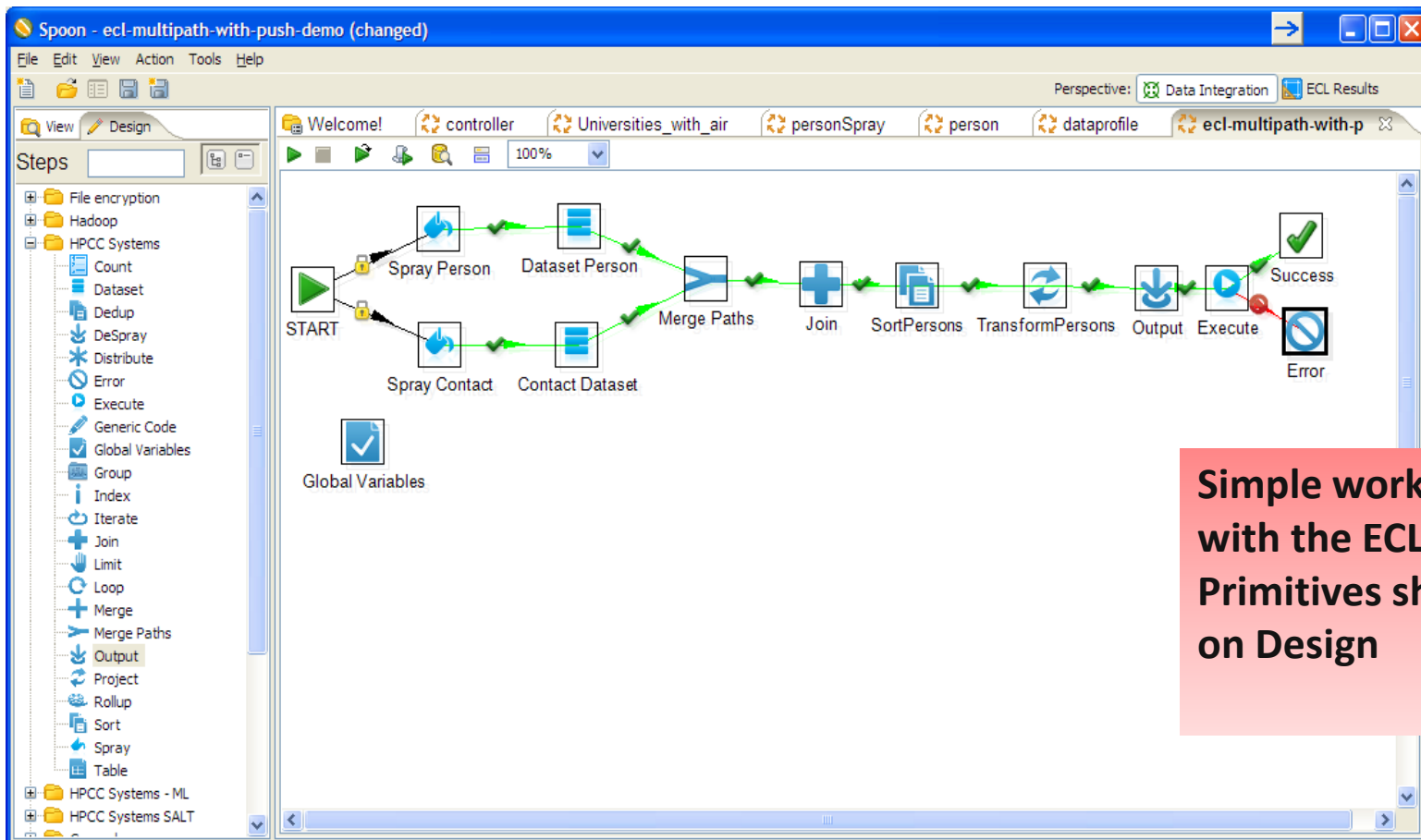
## Data Integration

---

# Data Integration using HPCC Flow

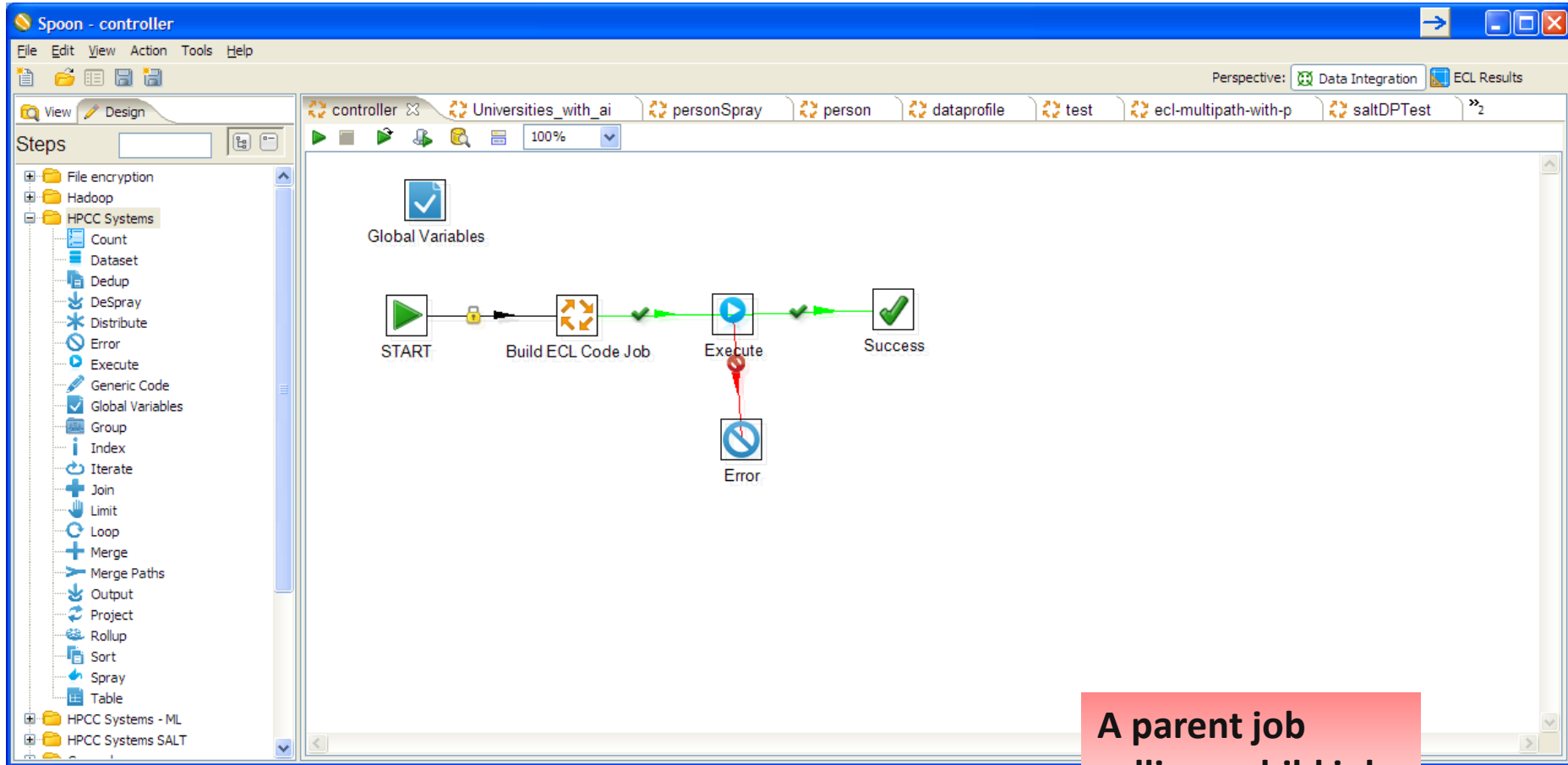
Drag and drop interface to data workflows on HPCC

Generates ECL code so it can be used also as a learning tool



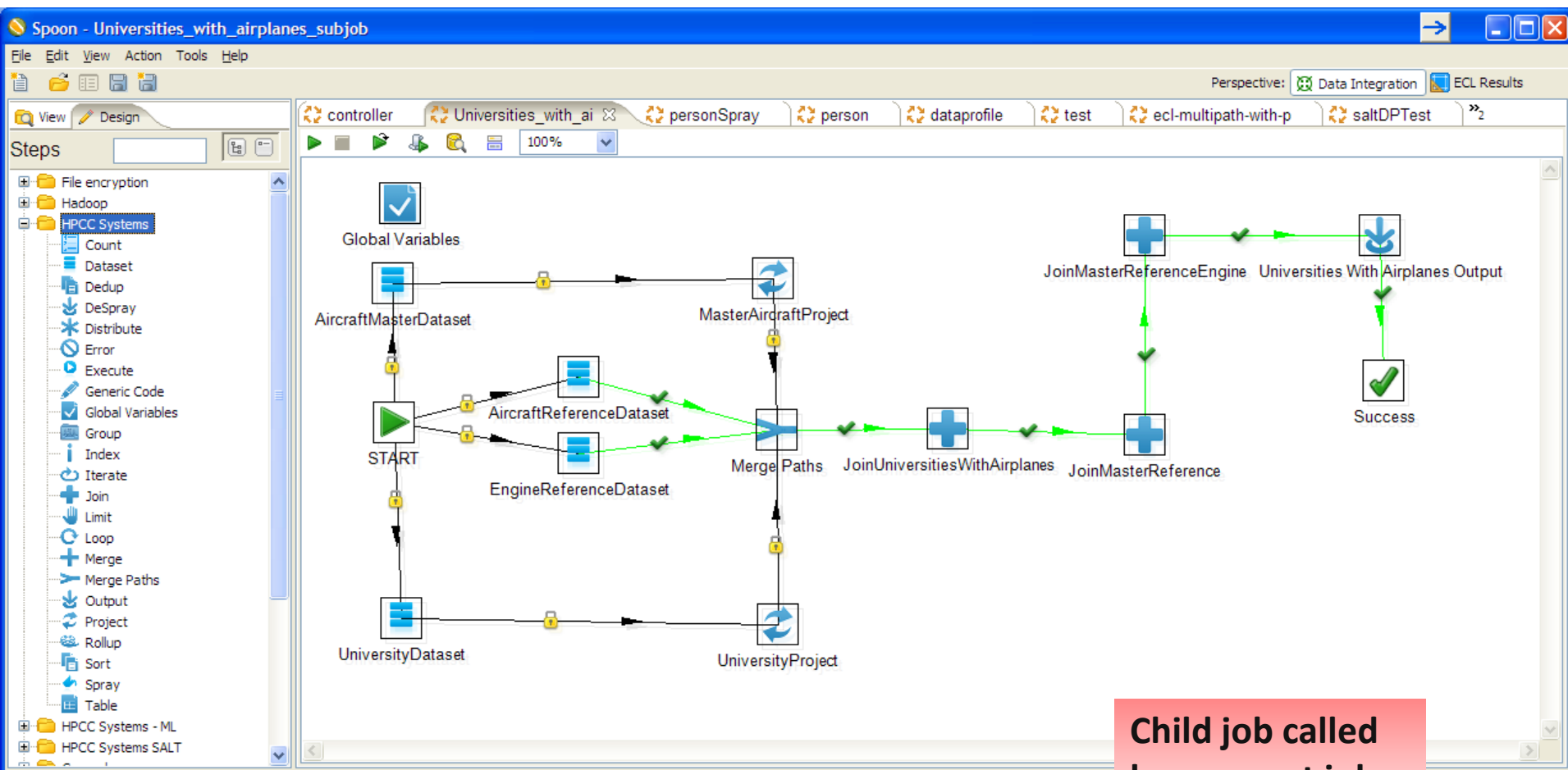
**Simple workflow,  
with the ECL  
Primitives shown  
on Design**

# Data Integration using HPC Flow



**A parent job  
calling a child job**

# Data Integration using HPC Flow



**Child job called by a parent job**

Get the source code: <https://github.com/hpcc-systems/spoon-plugins>

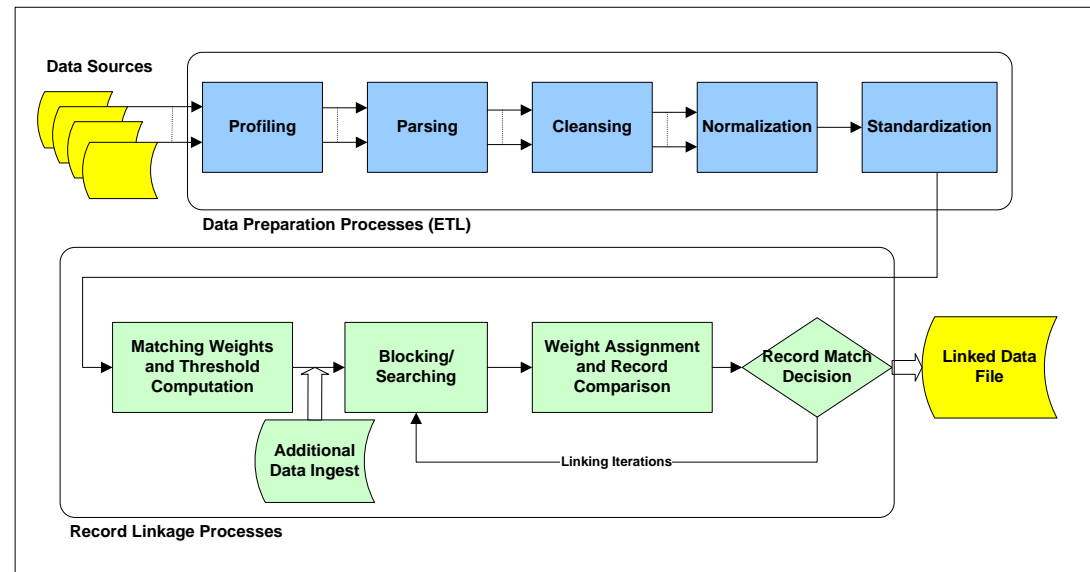
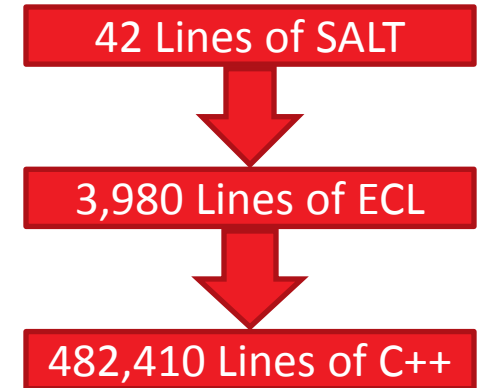
---

Record linkage, entity disambiguation, entity resolution

---

# Scalable Automated Linking Technology (SALT)

- The acronym stands for “Scalable Automated Linking Technology”
- Template based ECL code generator
- Provides for automated data profiling, QA/QC, parsing, cleansing, normalization and standardization
- Sophisticated specificity based linking and clustering
- Significant productivity boost!



# Rules based vs. probabilistic record linkage

Thoroughly employing statistical algorithms, SALT has the two significant advantages over rules:

## 1. SALT sufficiently explore all credible matches

INPUT

- Flavio Villanustre, Atlanta — Record 1
- Javio Villanustre, Atlanta — Record 2

SALT

**Match**, because the system has learnt that “Villanustre” is specific because the frequency of occurrence is small and there is only one present in Atlanta

Error

RULES

**NO MATCH**, because the rules determine that “Flavio” and “Javio” are not the same

## 2. SALT effectively minimizes false matches

INPUT

- John Smith, Atlanta
- John Smith, Atlanta

SALT

**NO Match**, because the system has learnt that “John Smith” is **not** specific because the frequency of occurrence is large and there are many present in Atlanta

Error

RULES

**MATCH**, because the rules determine that “John Smith” and the city for both the records match

# Record Linkage: SALT – Scalable Automated Linking Technology

Another purpose of SALT is to perform record linking





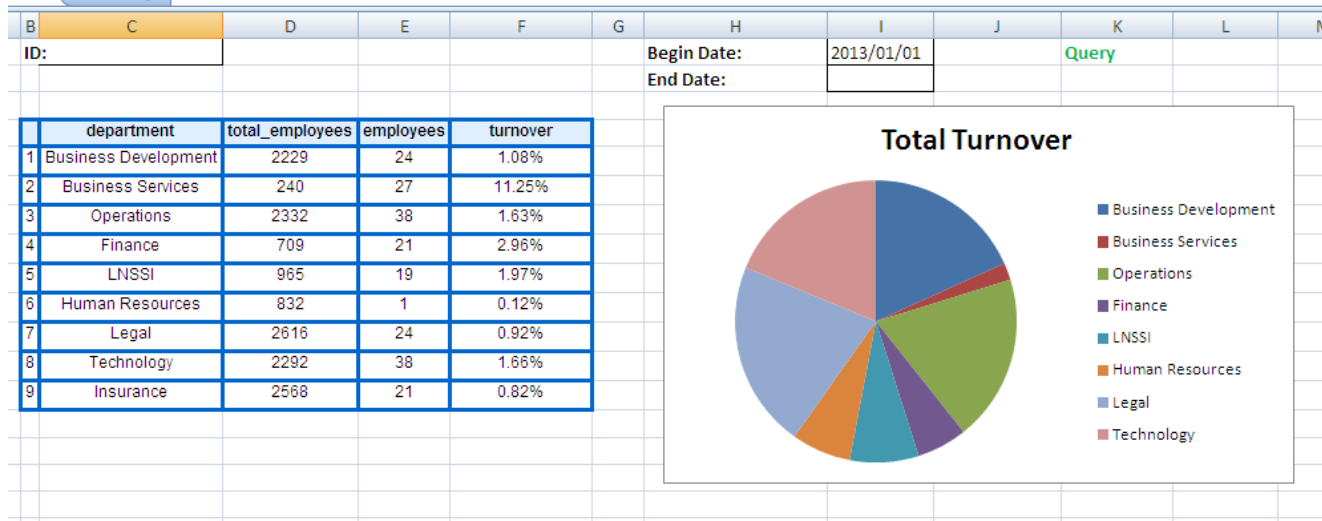
---

## Data Visualization

---

# Data Visualization: Excel Integration

Access a Roxie query to display a dynamic data dashboard from within Excel:



The Excel Visual Basic code to retrieve data:

```
Function refreshdata()  
    Application.EnableEvents = False  
    [Turnover].QueryTable.Connection = "URL;http://10.242.99.99:8002/WsEcl/xslt/query/roxie/excelqueries? asofdate=" + [qrydatecell].Text  
    Call [Turnover].QueryTable.Refresh  
    Application.EnableEvents = True  
End Function
```

# Data Visualization: Excel Integration...

Selecting the published Roxie Query in Excel:

The screenshot shows the Microsoft Excel interface with the 'Data' tab selected. A green arrow points to the 'From Web' button in the 'Data' group. The 'New Web Query' dialog box is open, showing the URL 'http://10.242.20.22:8002/WS/Ecl/xslt/query/roxie/turnoverqueries'. The dialog box displays a table of data with the following columns: department, totalturnover, employees, and employeepercent. The data is as follows:

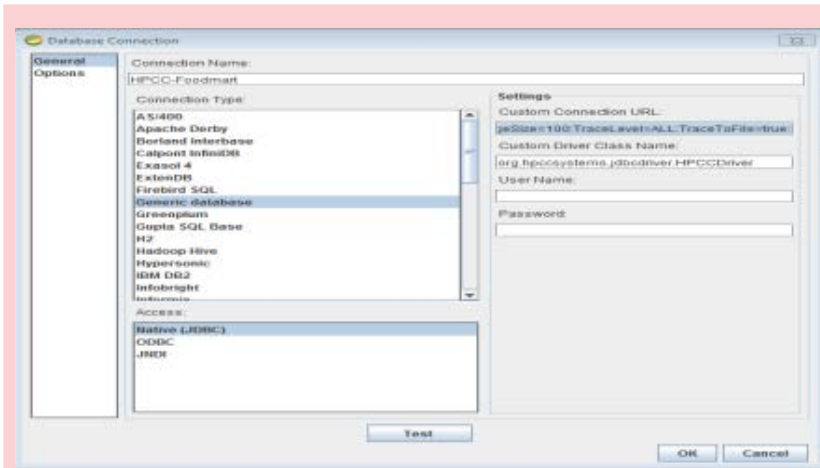
	department	totalturnover	employees	employeepercent
1	Business Development			
2	Business Services			
3	Operations			
4	Finance			
5	LNSSI			
6	Human Resources			
7	Legal			
8	Technology			
9	Insurance			

The dialog box also shows a 'Total Turnover' pie chart in the background. The 'Import' button is visible at the bottom right of the dialog box.

# Data Visualization: Pentaho Report Designer

Desktop visual report designer

Reports can be saved locally or published to a **Pentaho BI Server**



Copy the HPCC JDBC Driver in directory `\report-designer\lib\jdbc` and specify the following values:

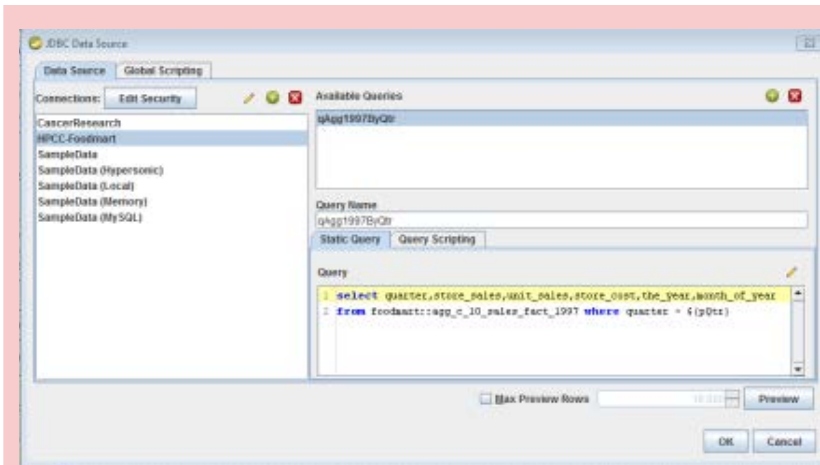
**Connection Type:** Generic Database

**Custom Connection URL:**

`jdbc:hppc:ServerAddress=99.999.999.9;Cluster=default:WsECLDirectPort=8008:EclResultLimit=100:QuerySet=thor:LazyLoad=true:PageSize=100:TraceLevel=ALL:TraceToFile=true`

**Custom Driver Class Name:**

`org.hpccsystems.jdbcdriver.HPCCDriver`



Reports/Charts can be created by writing a custom query or by calling a Roxie query.

**Adhoc SQL Query:**

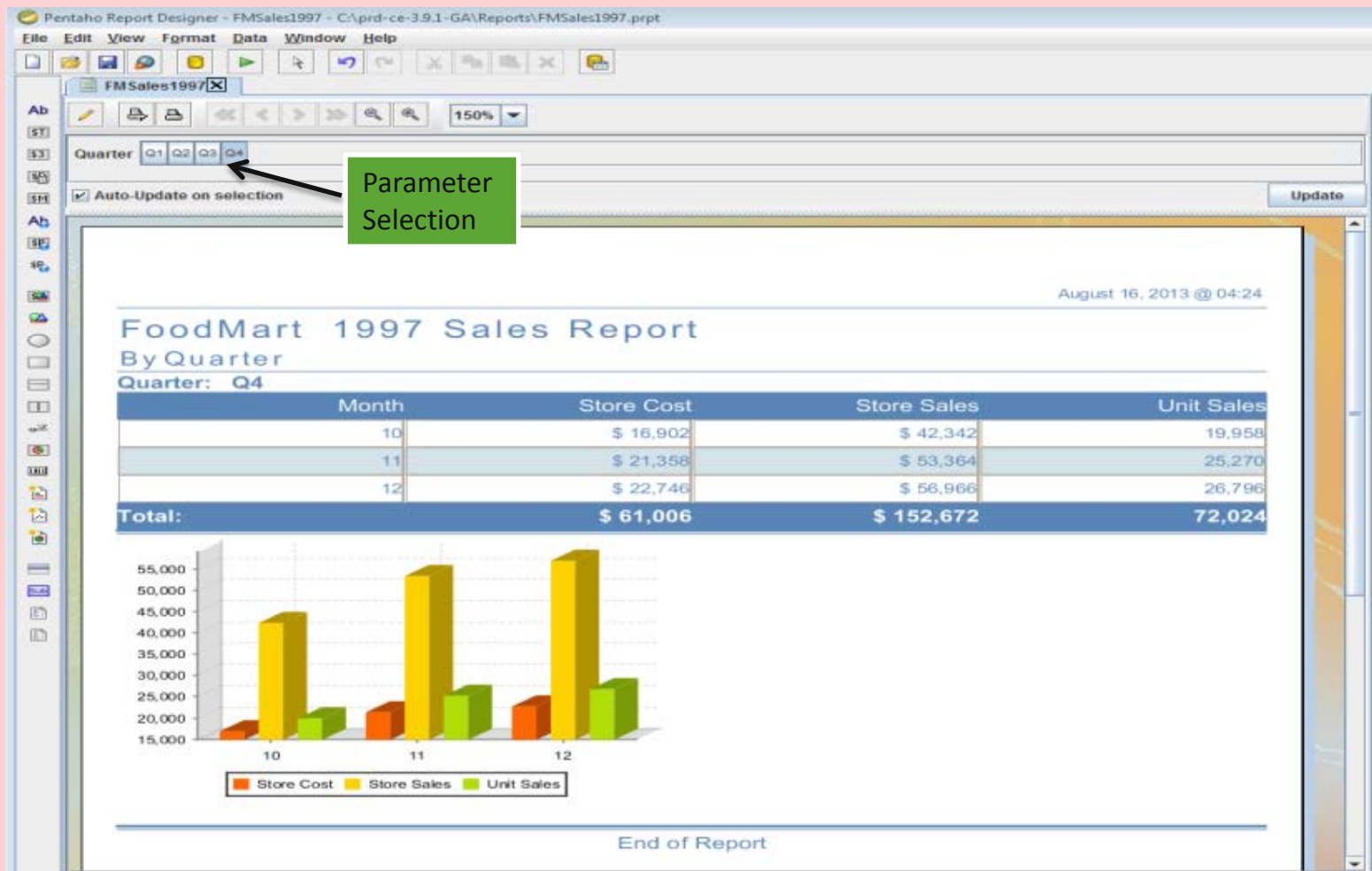
`select quarter,store_sales,unit_sales,store_cost,the_year,month_of_year from foodmart::agg_c_10_sales_fact_1997 where quarter = ${pQtr}`

Parameter

**Roxie Query:**

`call roxie::all_cancers_for_state(GA)`

# Data Visualization: Pentaho Report Designer



**Download:** [http://reporting.pentaho.com/report\\_designer.php](http://reporting.pentaho.com/report_designer.php)

# Data Visualization: Mondrian using Saiku UI

Real-time graphical online Analytical Processing (OLAP) on HPC

Users explore business data by drilling into and cross-tabulating information with speed-of-thought response times to complex analytical queries.

The screenshot shows the Saiku UI interface. On the left, there is a sidebar with 'Dimensions' (Store, Time) and 'Measures' (Count, Store Cost, Store Sales, Unit Sales). The main area displays a table with columns for Store Country, Store State, Store City, Unit Sales, Store Sales, Store Cost, and Count. A 'View MDX' dialog box is open, showing the MDX query used to generate the data.

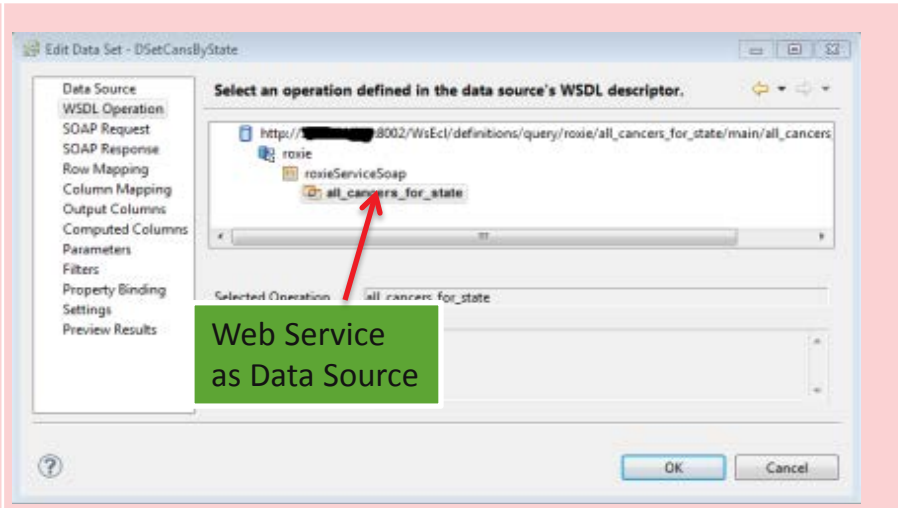
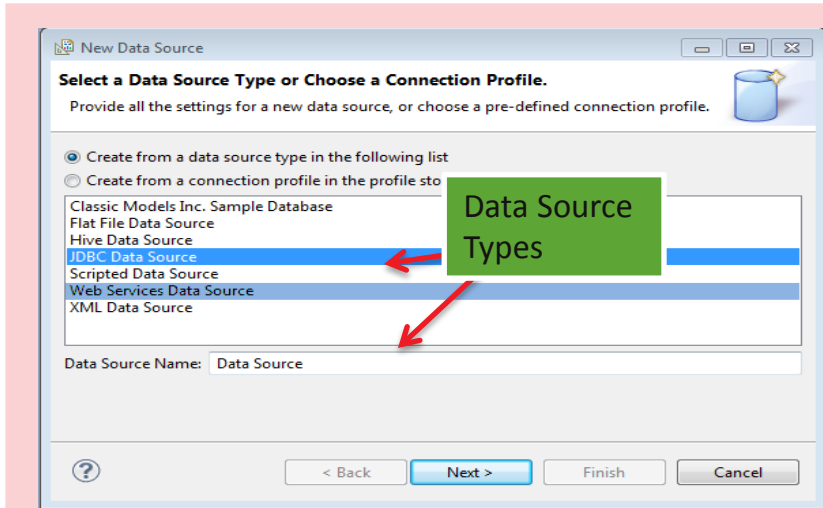
Store Country	Store State	Store City	Unit Sales	Store Sales	Store Cost	Count
USA			266,773.00	\$5,45,238.13	\$2,25,634.95	84,837
	CA		74,748.00	\$1,59,187.84	\$83,532.36	24,442
		Beverly Hills	21,333.00	\$45,758.24	\$18,266.78	6,815
		Los Angeles	25,683.00	\$54,945.28	\$21,772.85	8,287
		San Diego	25,635.00	\$54,431.14	\$21,714.55	8,095
		San Francisco	2,117.00	\$4,441.18	\$1,778.98	1,325
	OR		87,609.00	\$1,42,277.07	\$96,774.44	21,611
		Portland	36,079.00	\$55,358.79	\$21,949.56	8,264
		Salem	41,530.00	\$87,218.28	\$34,824.88	13,347
	WA		124,386.00	\$2,63,793.22	\$1,95,328.15	48,784
		Bellingham	2,237.00	\$4,738.23	\$1,896.84	1,388
		Bremerton	24,576.00	\$52,896.30	\$21,122.56	7,876
		Seattle	25,011.00	\$52,644.87	\$20,957.84	7,956
		Spokane	23,591.00	\$49,834.46	\$18,796.00	7,397
		Tacoma	35,257.00	\$74,843.96	\$29,988.18	11,184
		Walla Walla	2,293.00	\$4,705.97	\$1,888.50	1,338
		Yakima	11,491.00	\$24,329.23	\$9,714.14	3,652

```
SELECT
NON EMPTY {Hierarchize({[Measures].[Count],
[Measures].[Store Cost], [Measures].[Store Sales],
[Measures].[Unit Sales]})} ON COLUMNS,
NON EMPTY {Hierarchize({[Store].[USA], Filter
({[Store].[Store State].Members}, (Exists(Anccestor
([Store].CurrentMember, [Store].[Store Country]),
{[Store].[USA])).Count > 0)), Filter({[Store].[Store
City].Members}, (Exists(Anccestor
([Store].CurrentMember, [Store].[Store Country]),
{[Store].[USA])).Count > 0))}} ON ROWS
FROM [Sales]
```

Download: <http://meteorite.bi/saiku>

# Data Visualization: BIRT

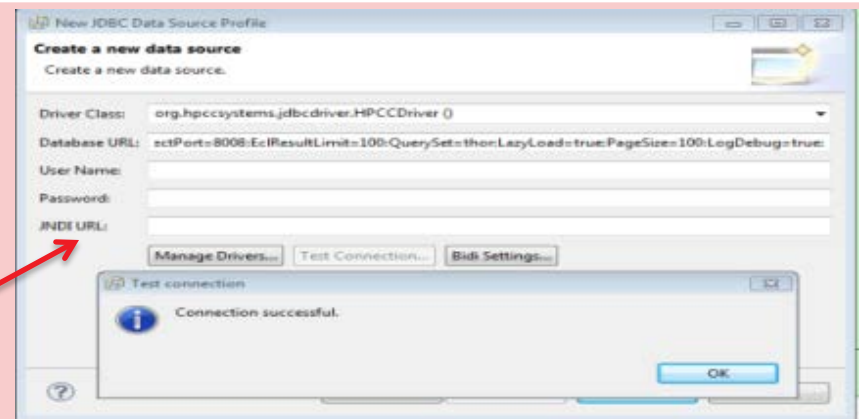
Eclipse-based open source reporting system with HPCC



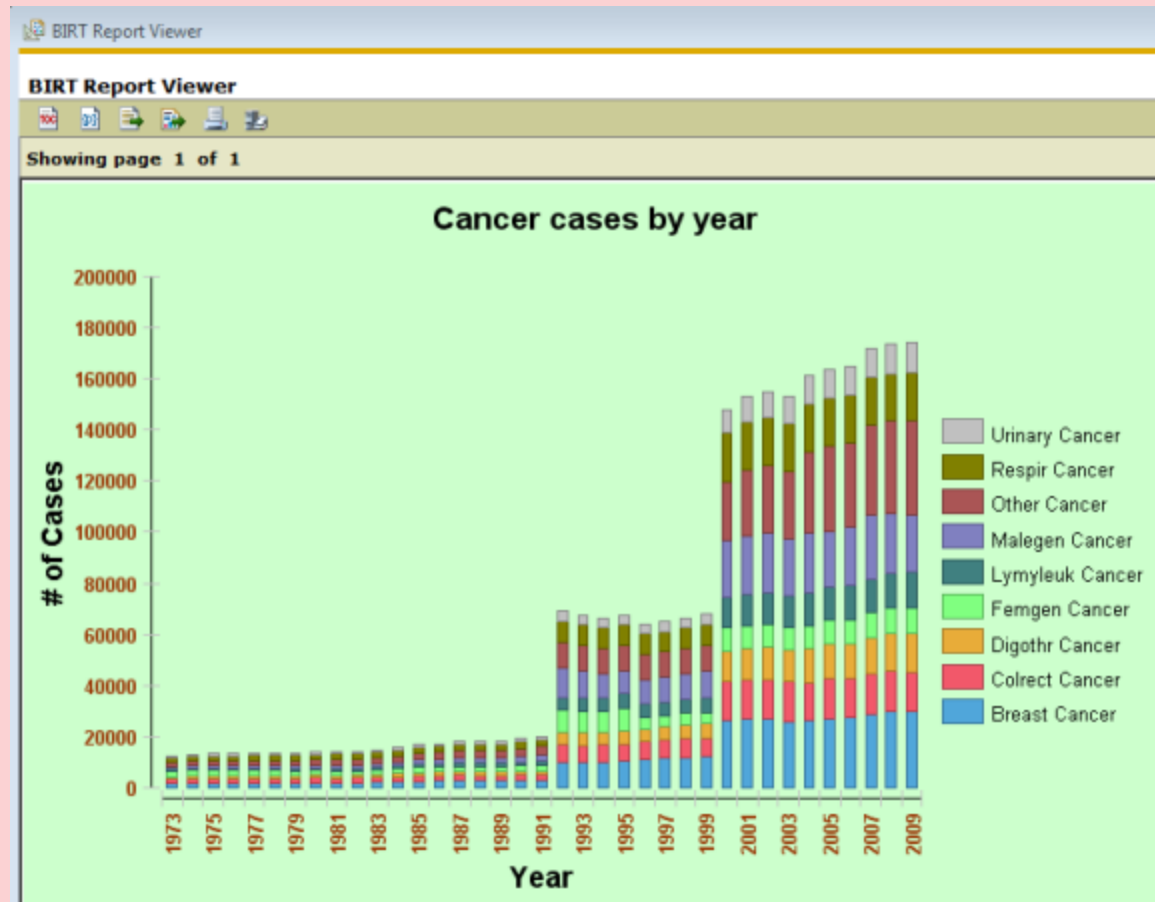
## BIRT Data Source Types:

- JDBC (for AdHoc queries or Roxie queries)
- Web Service (Roxie queries)

JDBC Data Source



# Data Visualization: BIRT Example Output



**Download:** <http://download.eclipse.org/birt/downloads/>



# Data Visualization: Result Cell Formatting

Visualizations can be used to:

- Understand new data
- Demonstrate data/products
- Summarize data.

In the ECL Watch Tech Preview, users can specify specific output columns to contain “HTML” or “JavaScript”, rather than the default plain text. See example ECL code snippet below.

This allows the user to perform some “Cell Formatting” of the results, such as:

- Add textual formatting, e.g.: `<b>Text</b>` (bold text)
- Cell color: Make a cell Red/Yellow to alert the viewer that certain rows are important.
- Embed Hyperlinks: Embed a link to a preformatted Google search, or a link to a Google map with lat. and long. etc.

```
import SampleData.Cars;
import Bundles.d3;

r := record
  dataset(Cars.CarsRecord) cars;
  varstring pc__javascript;
end;

d3Chart := d3.Chart('cars', 'name', 'unused');
d := dataset([Cars.CarsDataset, d3Chart.ParallelCoordinates], r);
d;
```

Name the column with a trailing “\_\_html” or “\_\_javascript”

# Data Visualization: Result Cell Formatting Output

HPCC Platform

192.168.1.201:8010/esp/files/stub.htm?Widget=HPCCPlatformWidget#

HPCC Platform

Workunits | Playground

W20130813-052410 | W20130813-052421 | W20130813-052429 | W20130813-052434

W20130813-052434 | Variables | **Outputs (1)** | Inputs | Timers (38) | Graphs (1) | ECL | Helpers (3) | Playground | XML

Outputs | Result 1

Download: Zip | GZip | XLS

```
## cars
1 | name | economy_mpg | cylinders | displacement_cc | power_hp | weight_lb | _0_60_mph_s | year
  |-----|-----|-----|-----|-----|-----|-----|-----|-----|
  | AMC Ambassador Brougham | 13 | 8 | 360 | 175 | 3821 | 11 | 73
  | AMC Ambassador DFL | 15 | 8 | 390 | 190 | 3850 | 8 | 70
  | AMC Ambassador SST | 17 | 8 | 304 | 150 | 3672 | 11 | 72
  | AMC Concord DL 6 | 20 | 6 | 232 | 90 | 3265 | 18 | 79
  | AMC Concord DL | 18 | 6 | 258 | 120 | 3410 | 15 | 78
  | AMC Concord DL | 23 | 4 | 151 | 0 | 3035 | 20 | 82
  | AMC Concord | 19 | 6 | 232 | 90 | 3210 | 17 | 78
  | AMC Concord | 24 | 4 | 151 | 90 | 3003 | 20 | 80
  | AMC Gremlin | 18 | 6 | 232 | 100 | 2789 | 15 | 73
  | AMC Gremlin | 19 | 6 | 232 | 100 | 2634 | 13 | 71
  | AMC Gremlin | 20 | 6 | 232 | 100 | 2914 | 16 | 75
  | AMC Gremlin | 21 | 6 | 199 | 90 | 2648 | 15 | 70
  | AMC Hornet Sportabout (Wagon) | 18 | 6 | 258 | 110 | 2962 | 13 | 71
  | AMC Hornet | 18 | 6 | 199 | 97 | 2774 | 15 | 70
  | AMC Hornet | 18 | 6 | 232 | 100 | 2945 | 16 | 73
  | AMC Hornet | 19 | 6 | 232 | 100 | 2901 | 16 | 74
  | AMC Hornet | 22 | 6 | 232 | 90 | 3085 | 17 | 76
  | AMC Matador (Wagon) | 14 | 8 | 304 | 150 | 4257 | 15 | 74
  | AMC Matador (Wagon) | 15 | 8 | 304 | 150 | 3892 | 12 | 72
  | AMC Matador | 14 | 8 | 304 | 150 | 3672 | 11 | 73
  | AMC Matador | 15 | 6 | 258 | 110 | 3730 | 19 | 75
  | AMC Matador | 15 | 8 | 304 | 120 | 3962 | 13 | 76
  | AMC Matador | 16 | 6 | 258 | 110 | 3632 | 18 | 74
  | AMC Matador | 18 | 6 | 232 | 100 | 3288 | 15 | 71
  | AMC Pacer D/L | 17 | 6 | 258 | 95 | 3193 | 17 | 76
  | AMC Pacer | 19 | 6 | 232 | 90 | 3211 | 17 | 75
  | AMC Rebel SST (Wagon) | 0 | 8 | 360 | 175 | 3850 | 11 | 70
  | AMC Rebel SST | 16 | 8 | 304 | 150 | 3433 | 12 | 70
  | AMC Spirit DL | 27 | 4 | 121 | 80 | 2670 | 15 | 79
  | Audi 100 LS | 20 | 4 | 114 | 91 | 2582 | 14 | 73
  | Audi 100 LS | 23 | 4 | 115 | 95 | 2694 | 15 | 75
  | Audi 100 LS | 24 | 4 | 107 | 90 | 2430 | 14 | 70
  | Audi 4000 | 34 | 4 | 97 | 78 | 2188 | 15 | 80
  | Audi 5000 | 20 | 5 | 131 | 103 | 2830 | 15 | 78
  | Audi 5000S (Diesel) | 36 | 5 | 121 | 67 | 2950 | 19 | 80
  | Audi Fox | 29 | 4 | 98 | 83 | 2219 | 16 | 74
  | BMW 2002 | 26 | 4 | 121 | 113 | 2234 | 12 | 70
  | BMW 320i | 21 | 4 | 121 | 110 | 2600 | 12 | 77
  | Buick Century 350 | 13 | 8 | 350 | 175 | 4100 | 13 | 73
```

pc\_javascript

# Data Visualization: Result Cell Formatting – Basic Chart

HPCC Platform

192.168.1.201:8010/esp/files/stub.htm?Widget=HPCCPlatformWidget#/stub/ECL/Workunits/W20130813-052410

HPCC Platform Wuid, User, More... LOGGED IN AS: ?

Workunits Playground

Workunits W20130813-052410 x W20130813-052421 x W20130813-052429 x W20130813-052434 x

W20130813-052410 Variables Outputs (1) Inputs (1) Timers (38) Graphs (1) ECL Helpers (3) Playground XML

Outputs Result 1 x

Download: Zip GZip XLS

state	numrows	data
AK	142	V
AK	66	P
AK	102	Q
AK	17	R
AK	231	S
AK	97	T
AK	67	U
AK	25	V
AK	72	W
AK	88	X
AK	151	Y
AK	69	Z

bar\_javascript

pie\_javascript

bubble\_javascript

AL	26	state	middleinitial	stat
AL	26	A		501
AL	26	B		610
AL	26	C		365
AL	26	D		334
AL	26	E		348
AL	26	F		325
AL	26	G		306
AL	26	H		339
AL	26	I		315
AL	26	J		579
AL	26	K		404
AL	26	L		426
AL	26	M		533
AL	26	N		377
AL	26	O		316
AL	26	P		346
AL	26	Q		264
AL	26	R		451
AL	26	S		252
AL	26	T		402
AL	26	U		493
AL	26	V		592
AL	26	W		357
AL	26	X		357
AL	26	Y		288
AL	26	Z		308

bar\_javascript

pie\_javascript

bubble\_javascript

1 - 50 of 56 results

1 2 >> 50

# Data Visualization: Result Cell Formatting – Tree Structures

The screenshot displays the HPCC Platform interface with a data table and four visualizations. The data table is as follows:

state	label	kids
AK	99504	AK 99504 L 28 AK 99504 T 18
AK	99505	AK 99505 E 20 AK 99505 Q 22 AK 99505 T 18
AK	99506	AK 99506 F 21 AK 99506 D 24 AK 99506 W 19
AK	99546	AK 99546 D 22 AK 99546 L 20 AK 99546 R 23
AK	99586	AK 99586 K 17 AK 99586 S 25
AL	85004	AL 85004 F 21 AL 85004 B 22 AL 85004 S 24
AL	85007	AL 85007 K 19 AL 85007 S 19
AL	85010	AL 85010 D 24 AL 85010 P 18 AL 85010 R 14
AL	85016	AL 85016 A 24 AL 85016 F 24 AL 85016 D 18
AL	85019	AL 85019 A 24 AL 85019 M 19 AL 85019 T 22
AR	71601	AR 71601 H 20 AR 71601 T 20
AR	71612	AR 71612 K 22 AR 71612 D 22
AR	71635	AR 71635 T 21 AR 71635 Q 19
AR	71642	AR 71642 D 16 AR 71642 P 20

The four visualizations are:

- ed\_javascript:** A flat tree structure where the root node 'USA' branches into state nodes (AK, AL, AR, AS), which then branch into individual record nodes.
- cp\_javascript:** A hierarchical bubble chart where nodes are represented as circles. The root 'USA' is a large blue circle, and child nodes are smaller circles, some containing letters (e.g., 'A', 'D', 'Y', 'E' for AK).
- rtt\_javascript:** A radial tree structure where nodes are arranged in concentric circles around a central point, with lines connecting parent and child nodes.
- sp\_javascript:** A sunburst chart where data is represented as segments in concentric rings, with the innermost ring representing the root and outer rings representing child nodes.

# Data Visualization: Result Cell Formatting – Graphs

HPCC Platform

192.168.1.201:8010/esp/files/stub.htm?Widget=HPCCPlatformWidget#/stub/ECL/Workunits/W20130813-052421

HPCC Platform

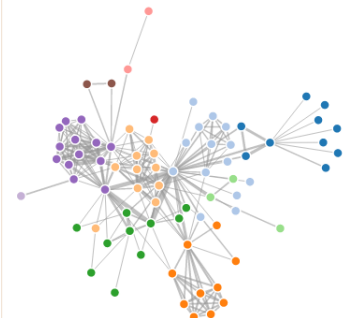
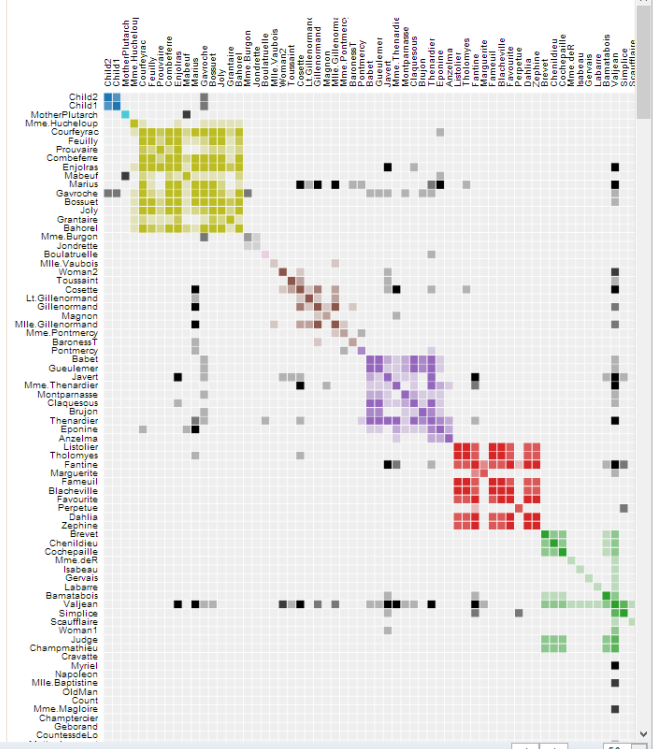
Workunits Playground

Workunits W20130813-052410 x W20130813-052421 x W20130813-052429 x W20130813-052434 x

W20130813-052421 Variables Outputs (1) Inputs Timers (38) Graphs (1) ECL Helpers (3) Playground XML

Outputs Result 1 x

Download: Zip GZip XLS

##	vertices	edges	fd_javascript	cooccurrence_javascript																																																																																																																																																																																																																																																
1	<table border="1"><thead><tr><th>id</th><th>name</th><th>category</th></tr></thead><tbody><tr><td>0</td><td>Myriel</td><td>1</td></tr><tr><td>1</td><td>Napoleon</td><td>1</td></tr><tr><td>2</td><td>Mlle.Baptistine</td><td>1</td></tr><tr><td>3</td><td>Mme.Magloire</td><td>1</td></tr><tr><td>4</td><td>CountessdeLo</td><td>1</td></tr><tr><td>5</td><td>Geborand</td><td>1</td></tr><tr><td>6</td><td>Champcercier</td><td>1</td></tr><tr><td>7</td><td>Cravatte</td><td>1</td></tr><tr><td>8</td><td>Count</td><td>1</td></tr><tr><td>9</td><td>OldMan</td><td>1</td></tr><tr><td>10</td><td>Labarre</td><td>2</td></tr><tr><td>11</td><td>Valjean</td><td>2</td></tr><tr><td>12</td><td>Marguerite</td><td>3</td></tr><tr><td>13</td><td>Mme.deR</td><td>2</td></tr><tr><td>14</td><td>Isabeau</td><td>2</td></tr><tr><td>15</td><td>Gervais</td><td>2</td></tr><tr><td>16</td><td>Tholomyes</td><td>3</td></tr><tr><td>17</td><td>Listoller</td><td>3</td></tr><tr><td>18</td><td>Fameuil</td><td>3</td></tr><tr><td>19</td><td>Blacheville</td><td>3</td></tr><tr><td>20</td><td>Favourite</td><td>3</td></tr><tr><td>21</td><td>Dahlia</td><td>3</td></tr><tr><td>22</td><td>Zephine</td><td>3</td></tr><tr><td>23</td><td>Fantine</td><td>3</td></tr><tr><td>24</td><td>Mme.Thenardier</td><td>4</td></tr><tr><td>25</td><td>Thenardier</td><td>4</td></tr><tr><td>26</td><td>Cosette</td><td>5</td></tr><tr><td>27</td><td>Javert</td><td>4</td></tr><tr><td>28</td><td>Fauchelevant</td><td>0</td></tr><tr><td>29</td><td>Bamatabois</td><td>2</td></tr><tr><td>30</td><td>Perpetue</td><td>3</td></tr><tr><td>31</td><td>Simplice</td><td>2</td></tr><tr><td>32</td><td>Scaufflaire</td><td>2</td></tr><tr><td>33</td><td>Woman1</td><td>2</td></tr><tr><td>34</td><td>Judge</td><td>2</td></tr><tr><td>35</td><td>Champmathieu</td><td>2</td></tr><tr><td>36</td><td>Brevet</td><td>2</td></tr><tr><td>37</td><td>Chenildieu</td><td>2</td></tr><tr><td>38</td><td>Cochebailla</td><td>2</td></tr></tbody></table>	id	name	category	0	Myriel	1	1	Napoleon	1	2	Mlle.Baptistine	1	3	Mme.Magloire	1	4	CountessdeLo	1	5	Geborand	1	6	Champcercier	1	7	Cravatte	1	8	Count	1	9	OldMan	1	10	Labarre	2	11	Valjean	2	12	Marguerite	3	13	Mme.deR	2	14	Isabeau	2	15	Gervais	2	16	Tholomyes	3	17	Listoller	3	18	Fameuil	3	19	Blacheville	3	20	Favourite	3	21	Dahlia	3	22	Zephine	3	23	Fantine	3	24	Mme.Thenardier	4	25	Thenardier	4	26	Cosette	5	27	Javert	4	28	Fauchelevant	0	29	Bamatabois	2	30	Perpetue	3	31	Simplice	2	32	Scaufflaire	2	33	Woman1	2	34	Judge	2	35	Champmathieu	2	36	Brevet	2	37	Chenildieu	2	38	Cochebailla	2	<table border="1"><thead><tr><th>source</th><th>target</th><th>weight</th></tr></thead><tbody><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>2</td><td>0</td><td>8</td></tr><tr><td>3</td><td>0</td><td>10</td></tr><tr><td>3</td><td>2</td><td>6</td></tr><tr><td>4</td><td>0</td><td>1</td></tr><tr><td>5</td><td>0</td><td>1</td></tr><tr><td>6</td><td>0</td><td>1</td></tr><tr><td>7</td><td>0</td><td>1</td></tr><tr><td>8</td><td>0</td><td>2</td></tr><tr><td>9</td><td>0</td><td>1</td></tr><tr><td>11</td><td>10</td><td>1</td></tr><tr><td>11</td><td>3</td><td>3</td></tr><tr><td>11</td><td>2</td><td>3</td></tr><tr><td>11</td><td>0</td><td>5</td></tr><tr><td>12</td><td>11</td><td>1</td></tr><tr><td>13</td><td>11</td><td>1</td></tr><tr><td>14</td><td>11</td><td>1</td></tr><tr><td>15</td><td>11</td><td>1</td></tr><tr><td>17</td><td>16</td><td>4</td></tr><tr><td>18</td><td>16</td><td>4</td></tr><tr><td>18</td><td>17</td><td>4</td></tr><tr><td>19</td><td>16</td><td>4</td></tr><tr><td>19</td><td>17</td><td>4</td></tr><tr><td>19</td><td>18</td><td>4</td></tr><tr><td>20</td><td>16</td><td>3</td></tr><tr><td>20</td><td>17</td><td>3</td></tr><tr><td>20</td><td>18</td><td>3</td></tr><tr><td>20</td><td>19</td><td>4</td></tr><tr><td>21</td><td>16</td><td>3</td></tr><tr><td>21</td><td>17</td><td>3</td></tr><tr><td>21</td><td>18</td><td>3</td></tr><tr><td>21</td><td>19</td><td>3</td></tr><tr><td>21</td><td>20</td><td>5</td></tr><tr><td>22</td><td>16</td><td>3</td></tr><tr><td>22</td><td>17</td><td>3</td></tr><tr><td>22</td><td>18</td><td>3</td></tr><tr><td>22</td><td>19</td><td>3</td></tr><tr><td>22</td><td>20</td><td>4</td></tr><tr><td>22</td><td>21</td><td>4</td></tr></tbody></table>	source	target	weight	1	0	1	2	0	8	3	0	10	3	2	6	4	0	1	5	0	1	6	0	1	7	0	1	8	0	2	9	0	1	11	10	1	11	3	3	11	2	3	11	0	5	12	11	1	13	11	1	14	11	1	15	11	1	17	16	4	18	16	4	18	17	4	19	16	4	19	17	4	19	18	4	20	16	3	20	17	3	20	18	3	20	19	4	21	16	3	21	17	3	21	18	3	21	19	3	21	20	5	22	16	3	22	17	3	22	18	3	22	19	3	22	20	4	22	21	4		
id	name	category																																																																																																																																																																																																																																																		
0	Myriel	1																																																																																																																																																																																																																																																		
1	Napoleon	1																																																																																																																																																																																																																																																		
2	Mlle.Baptistine	1																																																																																																																																																																																																																																																		
3	Mme.Magloire	1																																																																																																																																																																																																																																																		
4	CountessdeLo	1																																																																																																																																																																																																																																																		
5	Geborand	1																																																																																																																																																																																																																																																		
6	Champcercier	1																																																																																																																																																																																																																																																		
7	Cravatte	1																																																																																																																																																																																																																																																		
8	Count	1																																																																																																																																																																																																																																																		
9	OldMan	1																																																																																																																																																																																																																																																		
10	Labarre	2																																																																																																																																																																																																																																																		
11	Valjean	2																																																																																																																																																																																																																																																		
12	Marguerite	3																																																																																																																																																																																																																																																		
13	Mme.deR	2																																																																																																																																																																																																																																																		
14	Isabeau	2																																																																																																																																																																																																																																																		
15	Gervais	2																																																																																																																																																																																																																																																		
16	Tholomyes	3																																																																																																																																																																																																																																																		
17	Listoller	3																																																																																																																																																																																																																																																		
18	Fameuil	3																																																																																																																																																																																																																																																		
19	Blacheville	3																																																																																																																																																																																																																																																		
20	Favourite	3																																																																																																																																																																																																																																																		
21	Dahlia	3																																																																																																																																																																																																																																																		
22	Zephine	3																																																																																																																																																																																																																																																		
23	Fantine	3																																																																																																																																																																																																																																																		
24	Mme.Thenardier	4																																																																																																																																																																																																																																																		
25	Thenardier	4																																																																																																																																																																																																																																																		
26	Cosette	5																																																																																																																																																																																																																																																		
27	Javert	4																																																																																																																																																																																																																																																		
28	Fauchelevant	0																																																																																																																																																																																																																																																		
29	Bamatabois	2																																																																																																																																																																																																																																																		
30	Perpetue	3																																																																																																																																																																																																																																																		
31	Simplice	2																																																																																																																																																																																																																																																		
32	Scaufflaire	2																																																																																																																																																																																																																																																		
33	Woman1	2																																																																																																																																																																																																																																																		
34	Judge	2																																																																																																																																																																																																																																																		
35	Champmathieu	2																																																																																																																																																																																																																																																		
36	Brevet	2																																																																																																																																																																																																																																																		
37	Chenildieu	2																																																																																																																																																																																																																																																		
38	Cochebailla	2																																																																																																																																																																																																																																																		
source	target	weight																																																																																																																																																																																																																																																		
1	0	1																																																																																																																																																																																																																																																		
2	0	8																																																																																																																																																																																																																																																		
3	0	10																																																																																																																																																																																																																																																		
3	2	6																																																																																																																																																																																																																																																		
4	0	1																																																																																																																																																																																																																																																		
5	0	1																																																																																																																																																																																																																																																		
6	0	1																																																																																																																																																																																																																																																		
7	0	1																																																																																																																																																																																																																																																		
8	0	2																																																																																																																																																																																																																																																		
9	0	1																																																																																																																																																																																																																																																		
11	10	1																																																																																																																																																																																																																																																		
11	3	3																																																																																																																																																																																																																																																		
11	2	3																																																																																																																																																																																																																																																		
11	0	5																																																																																																																																																																																																																																																		
12	11	1																																																																																																																																																																																																																																																		
13	11	1																																																																																																																																																																																																																																																		
14	11	1																																																																																																																																																																																																																																																		
15	11	1																																																																																																																																																																																																																																																		
17	16	4																																																																																																																																																																																																																																																		
18	16	4																																																																																																																																																																																																																																																		
18	17	4																																																																																																																																																																																																																																																		
19	16	4																																																																																																																																																																																																																																																		
19	17	4																																																																																																																																																																																																																																																		
19	18	4																																																																																																																																																																																																																																																		
20	16	3																																																																																																																																																																																																																																																		
20	17	3																																																																																																																																																																																																																																																		
20	18	3																																																																																																																																																																																																																																																		
20	19	4																																																																																																																																																																																																																																																		
21	16	3																																																																																																																																																																																																																																																		
21	17	3																																																																																																																																																																																																																																																		
21	18	3																																																																																																																																																																																																																																																		
21	19	3																																																																																																																																																																																																																																																		
21	20	5																																																																																																																																																																																																																																																		
22	16	3																																																																																																																																																																																																																																																		
22	17	3																																																																																																																																																																																																																																																		
22	18	3																																																																																																																																																																																																																																																		
22	19	3																																																																																																																																																																																																																																																		
22	20	4																																																																																																																																																																																																																																																		
22	21	4																																																																																																																																																																																																																																																		

1 - 1 of 1 results

---

## Exploratory Data Analysis

---

# Exploratory Data Analysis

The EDA tool set will be tools added to HPC Flow that allow analytics on big data. The EDA tools will not provide all the functionality of existing tools so an additional benefit is the ability to reduce a “Big Data” problem down to a statistically significant subset of random records to be used in existing analytic tools such as R and SAS.

## EDA Procedures

- Frequency
- Mean, Median, Mode, Standard Deviation, Percentiles, Min, Max (Essentially Proc Univariate)
- Rank descending and ascending
- Tabulate procedure
- Random number generators
- Simple math ops such as  $x/y$  for each observation
- Options to easily add columns of data created by doing math on one or more existing columns of data
- Correlation statistics between two variables

## EDA Visualization

- Scatter Plots
- Line Plots
- Histograms
- Pie charts
- GUI will allow user to define which data is plotted against the X and Y axis
- Graph options will include Line, Bar, Pie and Histogram

---

## Sentiment Analysis

---



# Sentiment Analysis: ECL-ML

HPCC Systems and its ECL-ML library includes an extensible set of fully parallel Machine Learning (ML) and Matrix processing algorithms to assist with business intelligence; covering supervised and unsupervised learning, document and text analysis, statistics and probabilities, and general inductive inference related problems.

1. Download the ML Library  
<http://hpccsystems.com/ml>
2. Extract the contents of the zip file to the ECL IDE source folder.
3. Reference the library in your ECL source using a import statement as shown in the example:

```
IMPORT * FROM ML;
IMPORT * FROM ML.Cluster;
IMPORT * FROM ML.Types;

x2 := DATASET([ {1, 1, 1}, {1, 2, 5}, {2, 1, 5}, {2, 2, 7},
{3, 1, 8}, {3, 2, 1}, {4, 1, 0}, {4, 2, 0}, {5, 1, 9}, {5, 2,
3}, {6, 1, 1}, {6, 2, 4}, {7, 1, 9}, {7, 2,
4}],NumericField);

c := DATASET([ {1, 1, 1}, {1, 2, 5}, {2, 1, 5}, {2, 2, 7},
{3, 1, 9}, {3, 2, 4}],NumericField);

x3 := Kmeans(x2,c);

OUTPUT(x3);
```

# ECL-ML Classification: Naïve Bayes Classification

ML.Classify tackles the problem: “given I know these facts about an object; can I predict some other value or attribute of that object.” For example, can I predict whether sentiment of a tweet is positive or negative?

Using a classifier in ML involves three logical steps:

1. Learning the model from a training set of data that has been classified externally.
2. Testing. Getting measures of how well the classifier fits.
3. Classifying. Apply the classifier to new data in order to give it a classification

```
IMPORT ML;

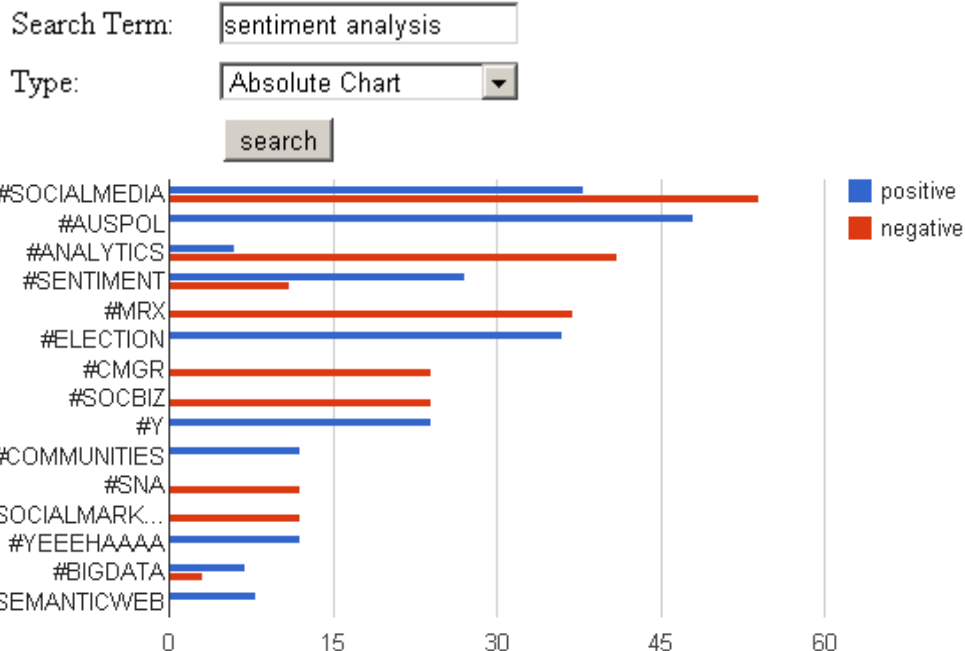
/* Use ML.Docs module to pre-process tweet text and generate
classification training set: IndependentDS, ClassDS

dRaw - collection of positive and negative sentiment tweets
dTokens := ML.Docs.Tokenize.Split(ML.Docs.Tokenize.Clean(dRaw));
dLexicon := ML.Docs.Tokenize.Lexicon(dTokens);
ML.Docs.Trans(ML.Docs.Tokenize.ToO(dTokens,dLexicon)).WordsCounted;
*/
BayesModule := ML.Classify.NaiveBayes;
// Learning the model
Model := BayesModule.LearnD(IndependentDS, ClassDS);
// Testing
TestModule := BayesModule.TestD(IndependentDS, ClassDS);
OUTPUT(TestModule);
//Classifying
Results := BayesModule.ClassifyD(IndependentDS, Model);
OUTPUT(Results);
```

# Sentiment Analysis: “Sentlyze” Twitter Sentiment Classification

Sentlyze uses HPCC Systems and its ML-Library to classify tweets with positive or negative sentiment.

Enter a search query and receive real-time tweets via the Twitter API in real-time.



**Try the demo:**

<http://hpccsystems.com/demos/twitter-sentiment>

**Get the source code:**

<https://github.com/hpcc-systems/ecl-ml>

Example code of a dataset of tweets that are classified using both Keyword Count and Naïve Bayes sentiment classifiers:

```
IMPORT Sentlyze;  
  
Tweets :=  
DATASET('~SENTILYZE::TWEETS',Sentlyze.Types.TweetType.CSV);  
  
rawTweets := Sentlyze.PreProcess.ConvertToRaw(Tweets);  
  
processTweets := Sentlyze.PreProcess.RemoveAnalysis(rawTweets)  
  
kcSentiment := Sentlyze.KeywordCount.Classify(processTweets);  
  
nbSentiment := Sentlyze.NaiveBayes.Classify(processTweets);  
  
OUTPUT(kcSentiment,NAMED('TwitterSentiment_KeywordCount'));  
  
OUTPUT(nbSentiment,NAMED('TwitterSentiment_NaiveBayes'));
```

---

## Graph Traversal Problem Handling

---

# Graph Traversal Problem Handling

Data analysis techniques using HPC Systems ECL can be applied to social graph traversal problems for finding customer insight and detection of important data patterns and trends.

## Get the example code:

<http://hpccsystems.com/download/docs/six-degrees>

```
/* *****  
ATTRIBUTE PURPOSE:  
Produce a series of sets for Actors and Movies that are : distance-0  
away (KBacons Direct movies ), distance-2 Away KBacon's Costars Movies ,  
distance-3 away - Movies of Costars of Costars etc all the way upto level 7  
  
The nested attributes below are shown here together for the benefit of the reader.  
  
Notes on variable naming convention used for costars and movies  
KBMovies      : Movies Kevin Bacon Worked in (distance 0)  
KBCoStars     : Stars who worked in KBMovies (distance 1)  
KBCoStarMovies : Movies worked in by KBCoStars  
                except KBMovies (distance 1)  
KBCo2Stars    : Stars(Actors) who worked in KBCoStarMovies (distance 2)  
KBCo2StarMovies : Movies worked in by KBCo2Stars  
                except KBCoStarMovies (distance 2)  
KBCo3Stars    : Stars(Actors) who worked in KBCo2StarMovies (distance 3)  
KBCo3StarMovies : Movies worked in by KBCo3Stars  
                except KBCo2StarMovies (distance 3)  
etc..  
***** */  
  
IMPORT Std;  
IMPORT IMDB;  
  
EXPORT KevinBaconNumberSets := MODULE  
  // Constructing a proper name match function is an art within itself  
  // For simplicity we will define a name as matching if both first and last name  
  //are found within the string  
  
  NameMatch(string full_name, string fname,string lname) :=  
    Std.Str.Find(full_name,fname,1) > 0 AND  
    Std.Str.Find(full_name,lname,1) > 0;  
  
  //----- Get KBacon Movies  
  AllKBEntries := IMDB.ActorsInMovies(NameMatch(actor,'Kevin','Bacon'));  
  EXPORT KBMovies := DEDUP(AllKBEntries, movie, ALL); // Each movie should ONLY occur once  
  
  //----- Get KBacon CoStars  
  CoStars := IMDB.ActorsInMovies(Movie IN SET(KBMovies,Movie));  
  EXPORT KBCoStars := DEDUP( CoStars(actor<>'Kevin Bacon'), actor, ALL);
```

## Getting Useful Information from Data using Links and Degrees of Separation aka “Kevin Bacon example”

```
IMPORT IMDB;  
IMDB.KevinBaconNumberSets.doCounts;
```

KB Movies	71
KB Co Stars	3520
KB Co Star Movies	33504
KB Co 2 Stars	430145
KB Co 2 Star Movies	251867
KB Co 3 Stars	896009
KB Co 3 Star Movies	51650
KB Co 4 Stars	102729
KB Co 4 Star Movies	2634
KB Co 5 Stars	6080
KB Co 5 Star Movies	190
KB Co 6 Stars	450
KB Co 6 Star Movies	14
KB Co 7 Stars	22

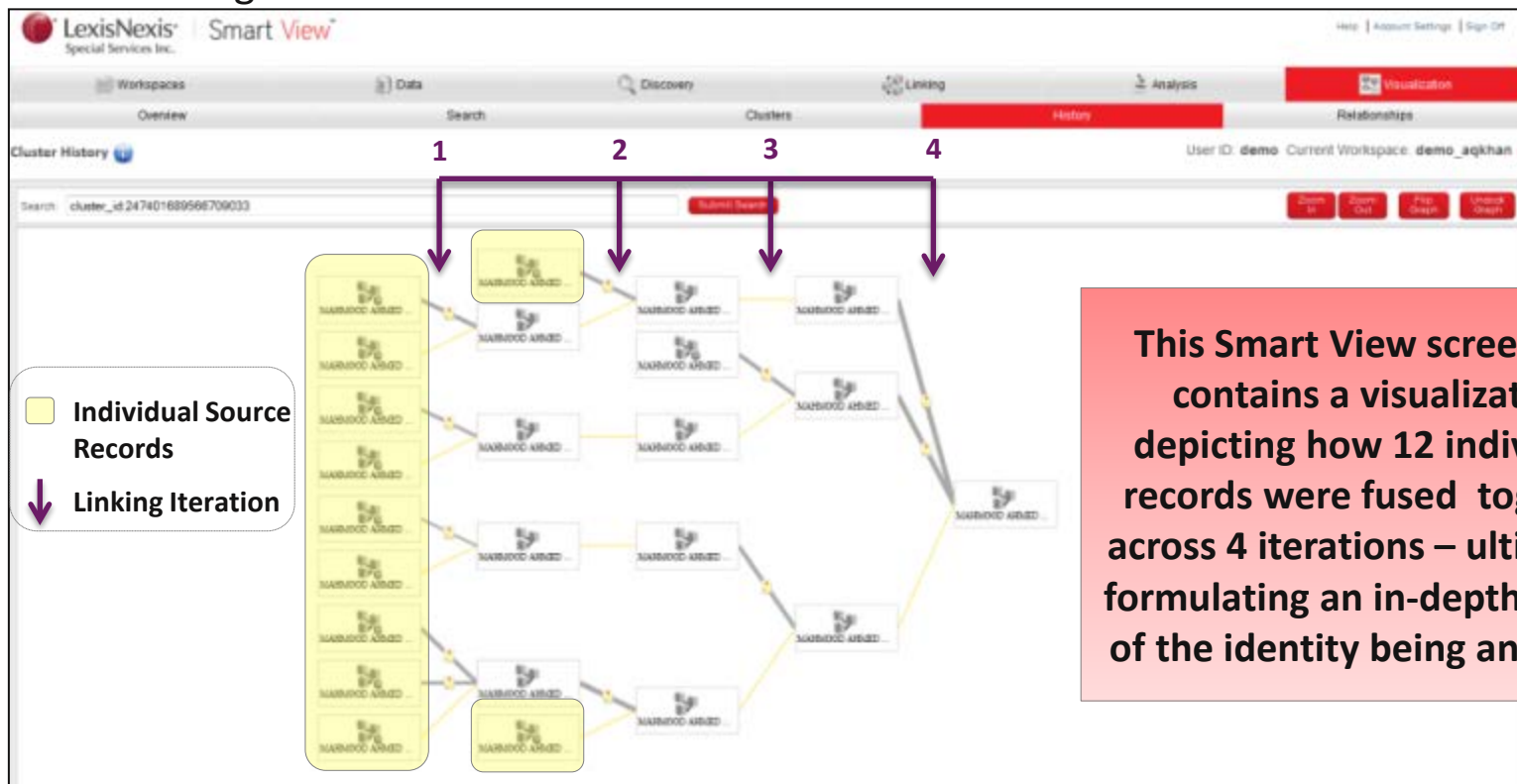
---

Smart View

---

# Smart View™ Entity Resolution & Visualization

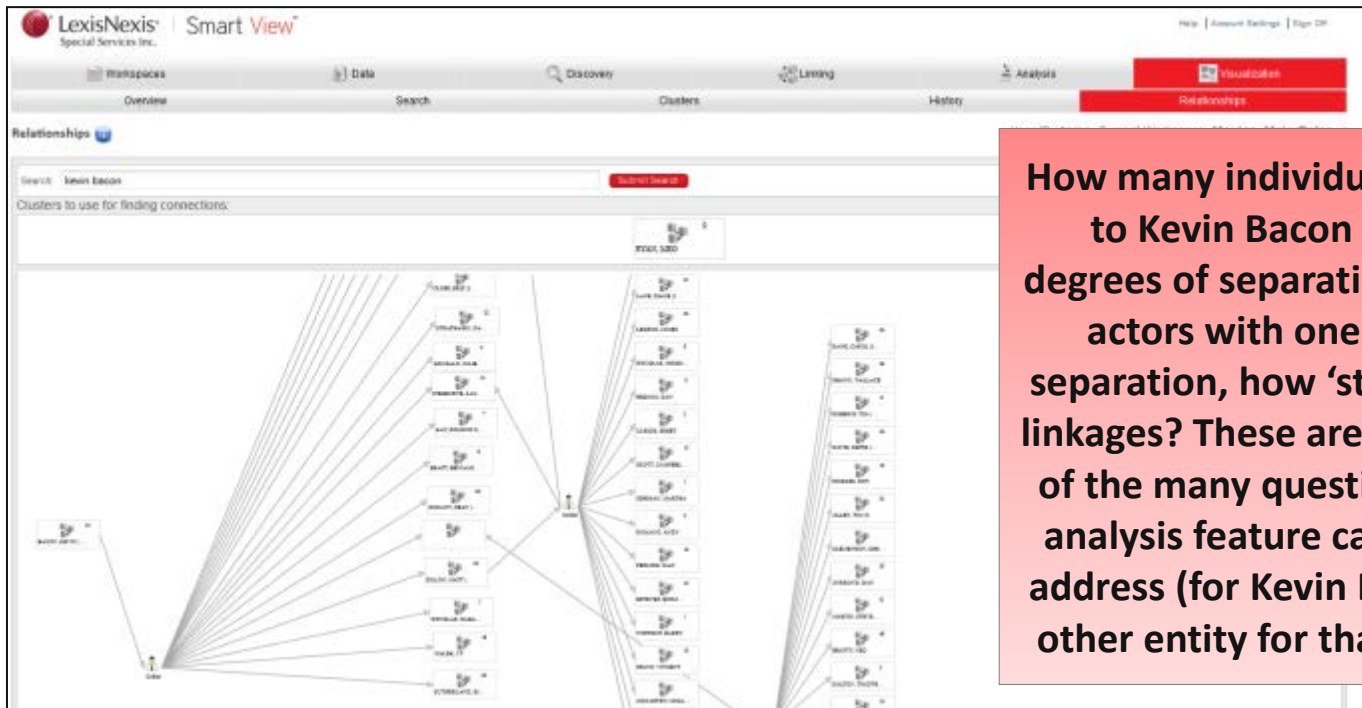
Smart View™ (SV) generates comprehensive profiles of ‘entities’ – people, places, and organizations – from a ‘perfect storm’ of big data (ie, disparate, complex and massive datasets requiring fusion for time and mission-critical applications). The combination of SV’s intuitive user interface, along with its leveraging of a patented, statistically-based clustering technique, make it a compelling option for the data scientist or business analyst faced with complex entity resolution challenges.



**This Smart View screenshot contains a visualization depicting how 12 individual records were fused together across 4 iterations – ultimately formulating an in-depth profile of the identity being analyzed.**

# Smart View™ Link Analysis & Visualization

The more comprehensive your entity profiles are, the greater the probability that non-obvious linkages between them will be uncovered. Smart View's link analysis feature allows the end-user to define a series of relationships, then uncover such relationships through an iterative, statistical process (similar to how SV's entity resolution problem is solved). To aide the data scientist in interpreting what their link analysis results signify, Smart View includes a set of relationship visualizations that display dynamically according to 'point' and 'click' commands.



**How many individuals are linked to Kevin Bacon through 3 degrees of separation? For those actors with one degree of separation, how 'strong' are the linkages? These are just a sample of the many questions SV's link analysis feature can help users address (for Kevin Bacon, or any other entity for that matter 😊)**



---

## SQL/ JDBC Integration

---

# SQL/JDBC Integration

The HPCC JDBC Driver provides SQL-based access to HPCC Systems data files and published queries.

This allows you to connect to the HPCC Systems platform through your favorite JDBC client and access your data without the need to write a single line of ECL!

- Analyze HPCC data using many JDBC Client User Interfaces
- Supports simple SQL SELECT or CALL syntax
  - Access HPCC data files as DB Tables
  - Access published queries as DB Stored Procedures
- Harnesses the full power of HPCC under the covers
  - Submitted SQL request generates ECL code which is submitted, compiled, and executed on your target cluster
  - Automatic Index fetching capabilities for quicker data fetches
- Creates entry-point for programmatic data access
- Leverage HPCC data and JDBC client functionality without need to learn and write ECL!
  - Opens the door for non ECL programmers to access HPCC data.

# SQL/JDBC Integration: Example Code

```
public class HpccConnect {
    public static void main(String[] args)
    {
        Connection con=null;;
        try {

            //1. Initialize HPCC Connection
            Class.forName("org.hpccsystems.jdbcdriver.HPCCDriver");
            con=DriverManager.getConnection("jdbc:hpcc:ServerAddress=99.99.99.99;
            Cluster=default:WsECLDirectPort=8010:EclResultLimit=100:QuerySet=thor:LazyLoad=true:PageSize=100:LogDebug=true:" , "" , "");

            //2. Write the Query to select foodMart::currency table
            PreparedStatement pst=con.prepareStatement("select conversion_ratio,currency_id,currency from foodmart::currency");
            pst.clearParameters();
            ResultSet rs=pst.executeQuery();

            //3. Print the headers to the console
            System.out.println("conversion_ratio\tcurrency_id\tcurrency");

            //4. Loop through all the records and write the information to the console
            while(rs.next()) {
                System.out.println(rs.getObject(1)+"\t\t"+rs.getString(2)+"\t\t"+rs.getString(3));
            }

            //5. Close the connection
            con.close();

        }
        catch(Exception e){
            e.printStackTrace();
        }
    }
}
```

Download JDBC Driver:

<http://hpccsystems.com/products-and-services/products/plugins/JDBC-Driver>

---

## Data Streaming

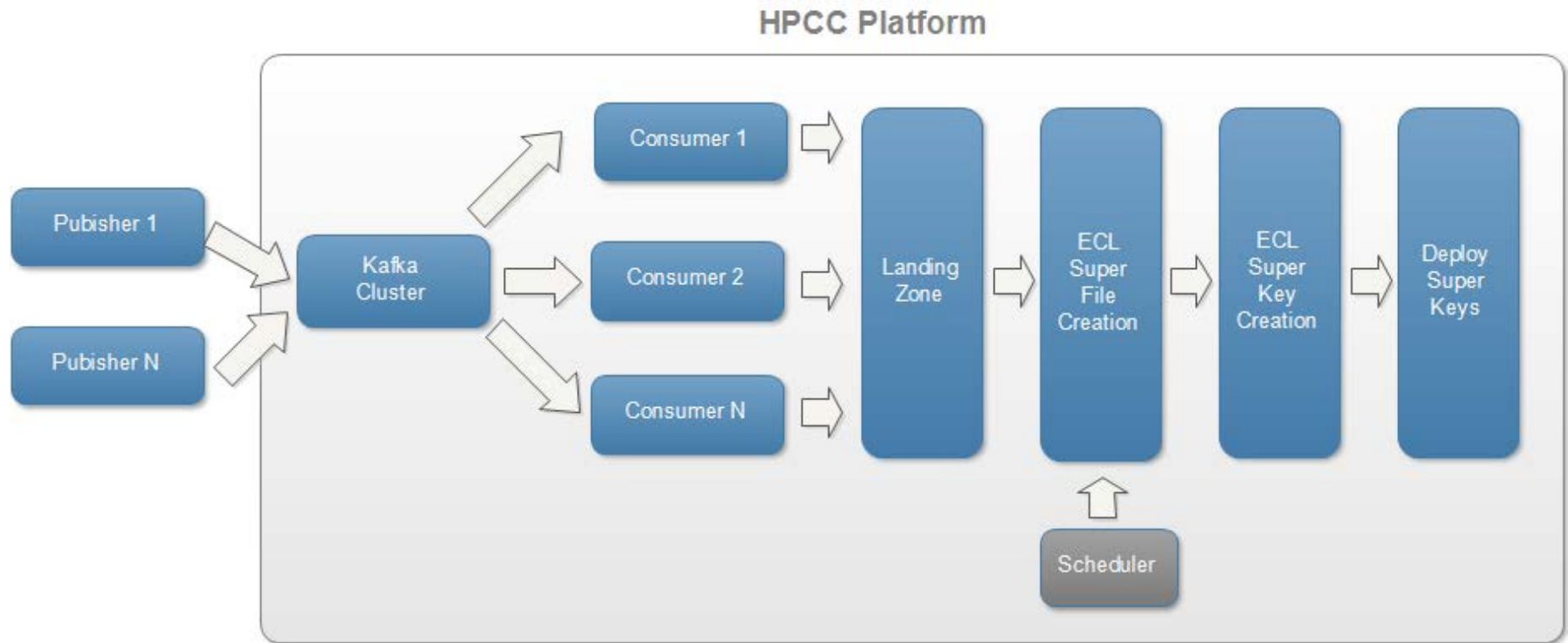
---

# Data Streaming: Apache Kafka

Apache Kafka is a distributed publish-subscribe messaging system. It is designed to support the following:

- Persistent messaging with  $O(1)$  disk structures that provide constant time performance even with many TB of stored messages.
- High-throughput: even with very modest hardware Kafka can support hundreds of thousands of messages per second.
- Explicit support for partitioning messages over Kafka servers and distributing consumption over a cluster of consumer machines while maintaining per-partition ordering semantics.
- Horizontally Scalable

# Data Streaming: How we are using Apache Kafka



---

## R Integration

---

# R Integration: rHPCC Plugin

rHPCC is an R package providing an interface between R and HPCC Systems.

```
# Creates an ECL "PROJECT" definition.
```

```
# The PROJECT function processes through all records in the recordset performing
```

```
# the transform function on each record in turn.
```

```
# Example ECL CODE: PROJECT(recordset,TRANSFORM(record,SELF := LEFT));
```

```
ECLProject <- setRefClass("ECLProject", contains="ECLDataset", fields = list(name = "character", inDataset="ECLDataset",  
  outECLRecord="ECLRecord", def = "character"),  
  methods = list(  
    getName = function() {  
      result <- name  
    },  
  
    addField = function(id, value) {  
      def <- paste(def, " ", id, " := ", value, ";")  
    },  
  
    print = function() {  
      result <- paste (name, " := PROJECT( ", inDataset$getName(), ", TRANSFORM(", outECLRecord$getName() ,  
        ",",def, " ));")  
    }  
  )  
)
```



# R Integration: Code Example

## ECL Code Snippet using EMBED(R)

```
IMPORT R;
```

### **EXAMPLE 1:**

```
SET OF INTEGER2 sortArray ( SET OF INTEGER2 val) := EMBED(R)  
    sort(val); // R code here  
ENDEMBED;
```

```
arr := sortArray ([-111, 0, 113, 12, 45, 5]);  
arr[3]; // Returns 5
```

### **EXAMPLE 2:**

```
STRING cat(VARSTRING what, VARSTRING who) := EMBED(R)  
    paste(what,who) // R code here  
ENDEMBED;
```

```
result := cat('Hello', 'World');  
result; // Hello World
```

---

## Java Integration

---

# Java Integration: Java ECL API

Java ECL API was developed along side the HPC Flow tool building the necessary JAVA code base to handle generating the ECL code, calling the Cluster via the SOAP interface, and retrieving results and server status updates. A secondary goal to this project is to provide examples on using the HPC Systems Platform from another programming language in general.

**Compile and  
submit to cluster**

Java ECL API Example with no client side compile check

```
String eclCode = "output('Hello World');";
//getECLSoap() is a class used that sets up the server connection and returns a ECLSoap class
ECLSoap es = getECLSoap();
boolean isValid = false;
ArrayList dsList = null;
String outStr = "";

isValid = es.executeECL(eclCode);
String wuid = es.getWuid();
if(!this.isValid){
    this.error += "\r\nFailed to execute code on the cluster, please verify your settings\r\n";
}
}
```

To check the ECL code with a local compiler call:

```
boolean isError = false;
boolean isWarning = false;
String errorMessage = (es.syntaxCheck(eclCode)).trim();
if(es.getErrorCount() > 0){
    isError = true;
}
if(es.getWarningCount() > 0){
    isWarning = true;
}
}
```

# Java Integration: Java ECL API – Code Generators

## ECL Primitive

- Count
- Dataset
- Dedup
- DeSpray
- Distribute
- FromField
- Group
- Index
- Iterate
- Join
- Limit
- Loop
- Merge
- Output
- Project
- Rollup
- Sort
- Spray
- Table
- ToField

## SALT

- Data Profile Report
- Concepts
- Specificities
- Hygiene Report
- Internal Linking

## ML Library

- Associate
- Build Classify
- Classification
- Classify
- Discretize
- KMeans
- Naïve Bayes

# Java Integration: Java ECL API – ECL Primitive Code Generator

## Render ECL Example

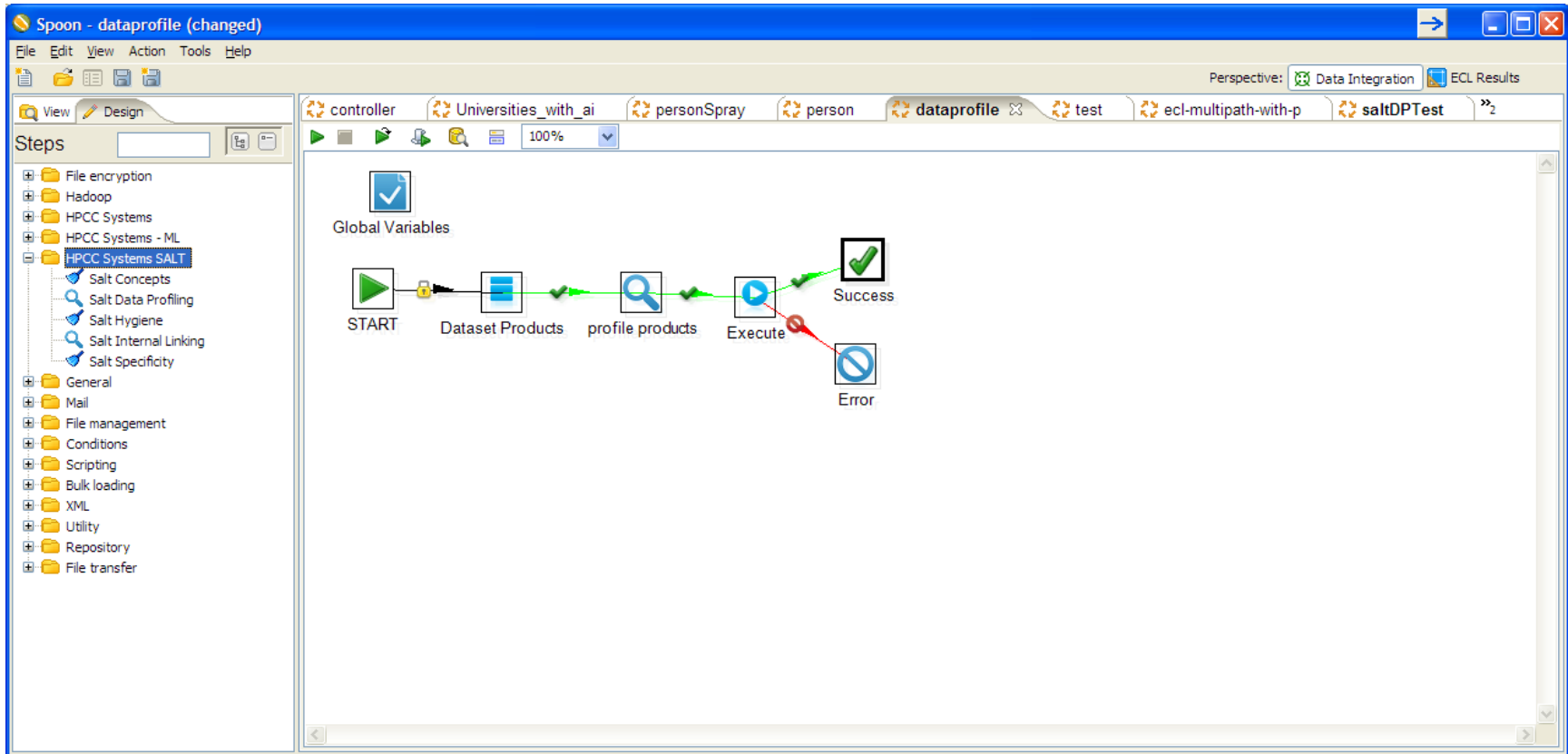
```
Join join = new Join();
join.setName("myRecSet");
join.setJoinCondition("left.theID" = "+"right.theID");
join.setJoinType("INNER");
join.setLeftRecordSet("MyLeftRecordSet");
join.setRightRecordSet("MyRightRecordSet");
String joinResults = join.ecl();
```

## Resulting ECL Code

```
myRecSet := join(MyLeftRecordSet,MyRightRecordSet,left.theID = right.theID, INNER);
```

# Java Integration: Java ECL API – SALT Code Generator

## Data Profile Example



# Java Integration: Java ECL API – SALT Code Generator

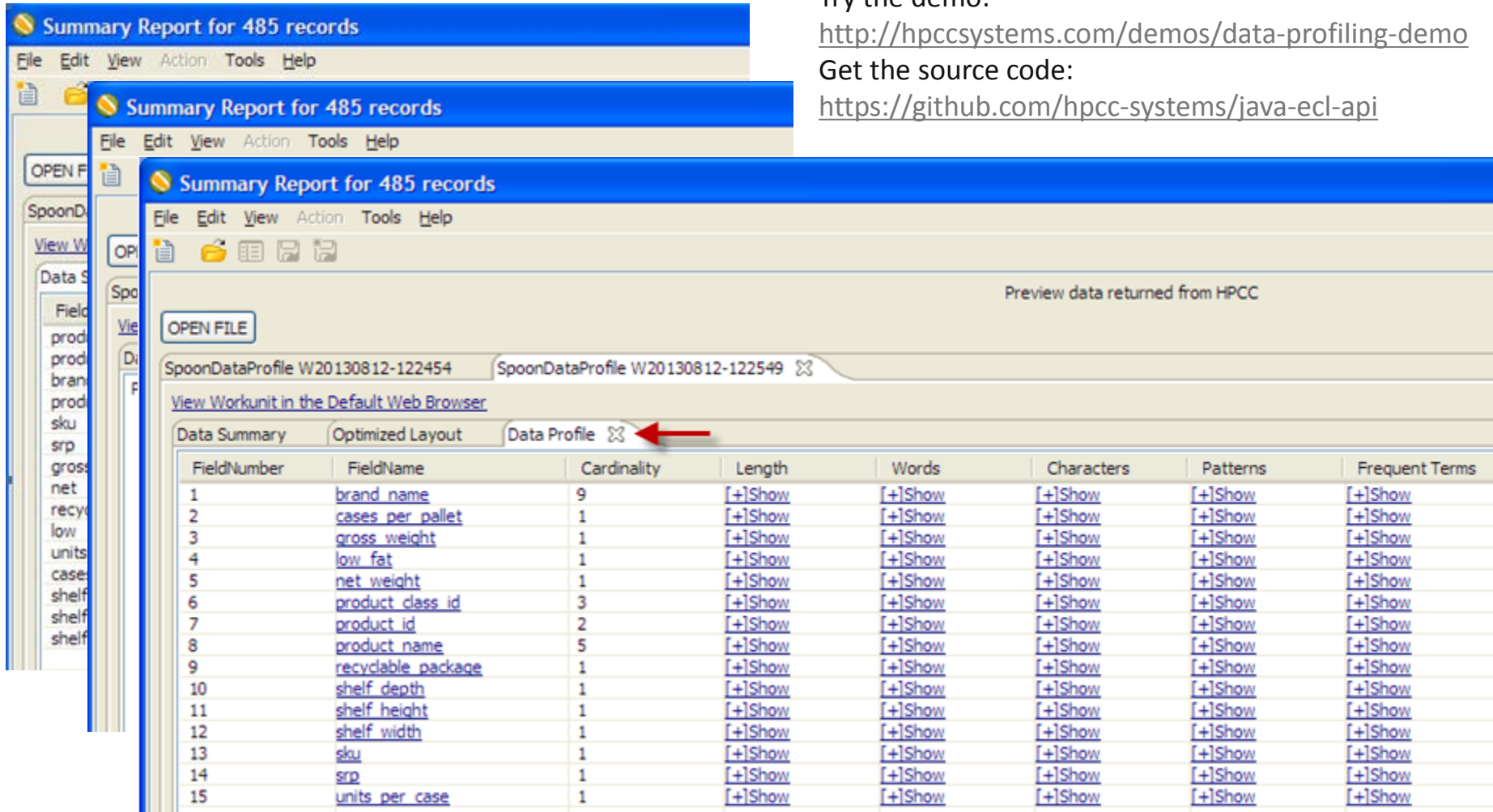
## Data Profile Example Result Set

Try the demo:

<http://hpcsystems.com/demos/data-profiling-demo>

Get the source code:

<https://github.com/hpcc-systems/java-ecl-api>



Summary Report for 485 records

Preview data returned from HPCC

View Workunit in the Default Web Browser

FieldNumber	FieldName	Cardinality	Length	Words	Characters	Patterns	Frequent Terms
1	brand name	9	[+]Show	[+]Show	[+]Show	[+]Show	[+]Show
2	cases per pallet	1	[+]Show	[+]Show	[+]Show	[+]Show	[+]Show
3	gross weight	1	[+]Show	[+]Show	[+]Show	[+]Show	[+]Show
4	low fat	1	[+]Show	[+]Show	[+]Show	[+]Show	[+]Show
5	net weight	1	[+]Show	[+]Show	[+]Show	[+]Show	[+]Show
6	product class id	3	[+]Show	[+]Show	[+]Show	[+]Show	[+]Show
7	product id	2	[+]Show	[+]Show	[+]Show	[+]Show	[+]Show
8	product name	5	[+]Show	[+]Show	[+]Show	[+]Show	[+]Show
9	recyclable package	1	[+]Show	[+]Show	[+]Show	[+]Show	[+]Show
10	shelf depth	1	[+]Show	[+]Show	[+]Show	[+]Show	[+]Show
11	shelf height	1	[+]Show	[+]Show	[+]Show	[+]Show	[+]Show
12	shelf width	1	[+]Show	[+]Show	[+]Show	[+]Show	[+]Show
13	sku	1	[+]Show	[+]Show	[+]Show	[+]Show	[+]Show
14	srp	1	[+]Show	[+]Show	[+]Show	[+]Show	[+]Show
15	units per case	1	[+]Show	[+]Show	[+]Show	[+]Show	[+]Show

---

## Python Integration

---



# Python Integration : Code Example

## ECL Code Snippet using EMBED(Python)

```
IMPORT Python;
```

### EXAMPLE 1:

```
SET OF INTEGER sortArrayOfIntegers (SET OF INTEGER val) := EMBED(Python)  
  return sorted (val) // Python code here  
ENDEMBED;
```

```
arr := sortArrayOfIntegers ([-111, 0, 113, 12, 45, 5]);  
arr[3]; // Returns 5
```

### EXAMPLE 2:

```
SET OF STRING sortArrayOfString (SET OF STRING val) := EMBED(Python)  
  return sorted (val) // Python code here  
ENDEMBED;
```

```
result := sortArrayOfString (['red','green','yellow']);  
result[1]; // Returns 'green'
```

- Version 5.2 should be out EOY.
- Ongoing R&D (5.2 and beyond):
  - Level 3 BLAS support
  - More Machine Learning related algorithms
  - Heterogeneous/hybrid computing (FPGA, memory computing, GPU)
  - Knowledge Engineering Language driving ML
  - General usability enhancements (GUI to SALT data profiling and linking, etc.)
  - Embedding other languages (Python, JavaScript, Java and R, for starters)
  - Integration with other tools and systems (Pentaho Mondrian, R, Kafka, etc.)

# Trivia

**Answer the question and win a prize!**



- What is the name of the data refinery engine that provides batch oriented data manipulation?
- What is the name of the data delivery engine that provides real-time analytics?
- What is the query processing language supporting the HPCC Systems platform?
- What can be used to generate ECL code for automating data profiling, parsing and cleansing?
- What does KEL stand for?
- Name three examples where HPCC can be integrated and applied.

# Useful Links

- LexisNexis Open Source HPCC Systems Platform: <http://hpccsystems.com>
  - Machine Learning portal: <http://hpccsystems.com/ml>
  - Online Training: <http://learn.lexisnexis.com/hpcc>
  - The HPCC Systems blog: <http://hpccsystems.com/blog>
  - Our GitHub portal: <https://github.com/hpcc-systems>
  - Community Forums: <http://hpccsystems.com/bb>
- 

## **Contact information:**

Phone: 678 694 3320

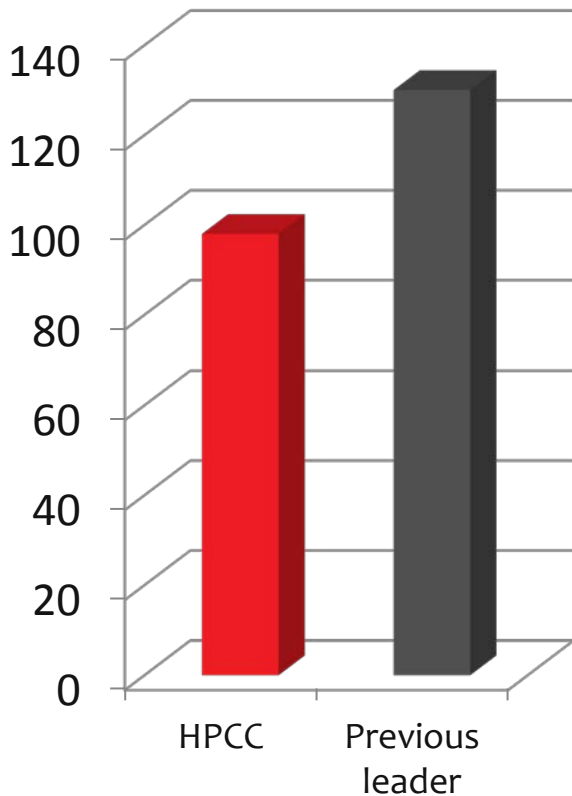
Email: [John.D.Holt@lexisnexis.com](mailto:John.D.Holt@lexisnexis.com)

# Questions???





## Execution Time (seconds)



## Productivity

```
// Perform global terasort
rec := record
  string10 key;
  string10 seq;
  string80 fill;
end;
in := STATETEST('terasort1',rec,FLAT);
OUTPUT SORT (key,UNSTABLE),,'nhctest::terasort1out',overwrite);
//End
```

**3 Lines of ECL**

```
}
abstract int findPartition(Text key);
abstract void print(PrintStream stm) throws IOException;
int getLevel() {
  return level;
}
}

/**
 * An inner trie node that contains 256 children based on the next
 * character.
 */
static class InnerTrieNode extends TrieNode {
  private TrieNode[] child;
  InnerTrieNode(int level) {
    super(level);
  }
  int findPartition(Text key) {
    int level = getLevel();
    if (key.getLength() <= level) {
      return child[0].findPartition(key);
    }
    return child[key.getBytes()[level] & 0xff].findPartition(key);
  }
}
```

**700+ Lines of Java MapReduce Code**

## Space/Cost



**HPCC**

**Previous leader**

# Medicaid Case Study

## Scenario

Proof of concept for Office of the Medicaid Inspector Generation (OMIG) of large Northeastern state. Social groups game the Medicaid system which results in fraud and improper payments.

## Task

Given a large list of names and addresses, identify social clusters of Medicaid recipients living in expensive houses, driving expensive houses.

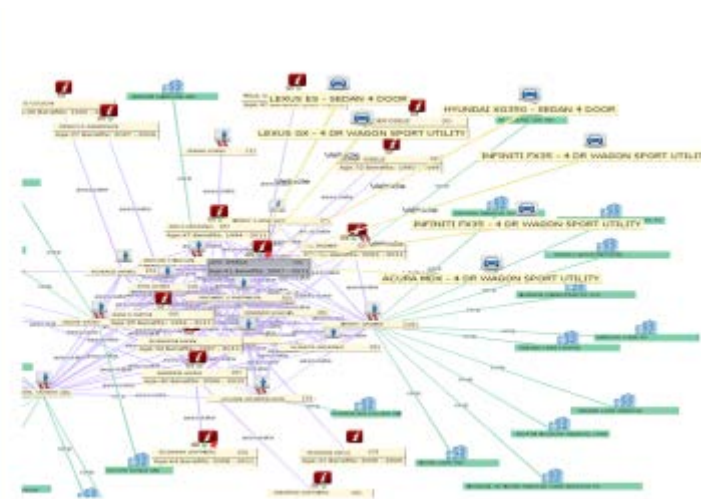
## Result

Interesting recipients were identified using asset variables, revealing hundreds of high-end automobiles and properties.

Leveraging the Public Data Social Graph, large social groups of interesting recipients were identified along with links to provider networks.

The analysis identified key individuals not in the data supplied along with connections to suspicious volumes of “property flipping” potentially indicative of mortgage fraud and money laundering

Label	LED_C00LE
Full Name	LED
Level Name	OSELE
SIRE	
DOB	
Date Paid Sa	1/18/2011
Medicaid	1
MedicaidCm	13
MedicaidDv	0
recip_id	
HighEndHous	Yes
autoModel_c	22000
occupant_civ	0
yearhouse_sac	200300119
yearhouse_sac	58992
rempage_sac	0
rempage_sac	0
highend_prop	Yes
value_pers	100000
total_debt_e	2
total_highend	1
vehicle_model	0
medicaid	0
id_clear_lead	20070201
id_clear_lead	20111130
highend_rfr	Yes
lower_pers	31505
make_desc	Lexus
model_desc	Lexus LS460 L Series 3.5L V6
yearmake_val	0-132009
	L2009 Infiniti FX35
	0-132009
	L2008 Infiniti FX35
	0-132009
	L2004 Lexus RTO 400
	0-132009
	L2003 Acura SPORT
	MDX 0-132009



Make Description	#	Make Description	#
Mercedes-Benz	46	Chevrolet	2
Lexus	41	Hummer	2
BMW	27	Jeep	2
Infiniti	13	Nissan	2
Acura	9	Toyota	2
Lincoln	8	Aston Martin	1
Audi	7	Bentley	1
Land Rover	7	Cadillac	1
Porsche	6	GMC	1
Jaguar	5	Honda	1
Mercedes Benz	3	Volkswagen	1
Saab	3	Volvo	1



# Network Traffic Analysis in Seconds

## Scenario

Conventional network sensor and monitoring solutions are constrained by inability to quickly ingest massive data volumes for analysis

- 15 minutes of network traffic can generate 4 Terabytes of data, which can take 6 hours to process
- 90 days of network traffic can add up to 300+ Terabytes

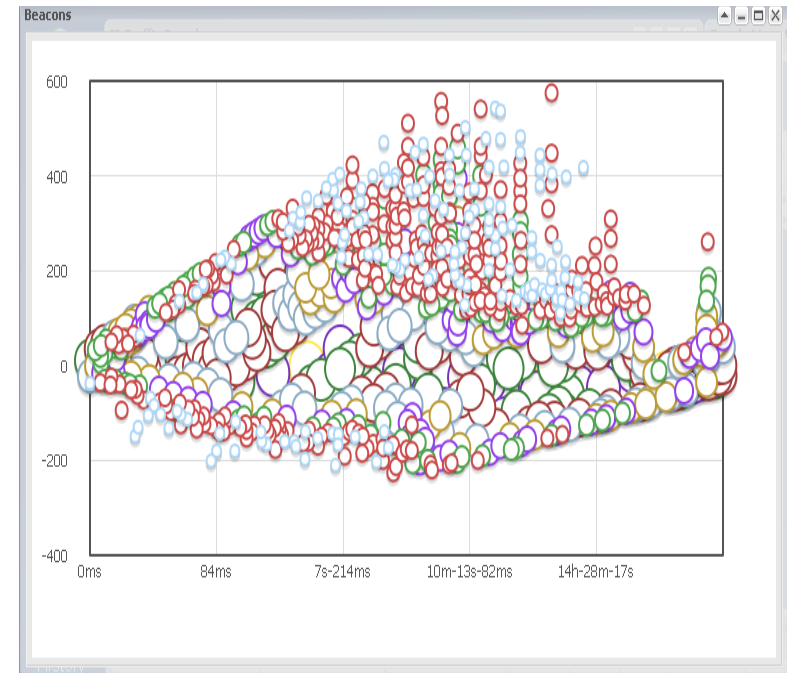
## Task

Drill into all the data to see if any US government systems have communicated with any suspect systems of foreign organizations in the last 6 months

- In this scenario, we look specifically for traffic occurring at unusual hours of the day

## Result

In seconds, the HPCC sorted through months of network traffic to identify patterns and suspicious behavior



Horizontal axis: time on a logarithmic scale  
Vertical axis: standard deviation (in hundredths)  
Bubble size: number of observed transmissions



# Wikipedia Pageview Demo

## Scenario

21 Billion Rows of Wikipedia Hourly Page view Statistics for a year.

## Task

Generate meaningful statistics to understand aggregated global interest in all Wikipedia pages across the 24 hours of the day built off all English Wikipedia page view logs for 12 months.

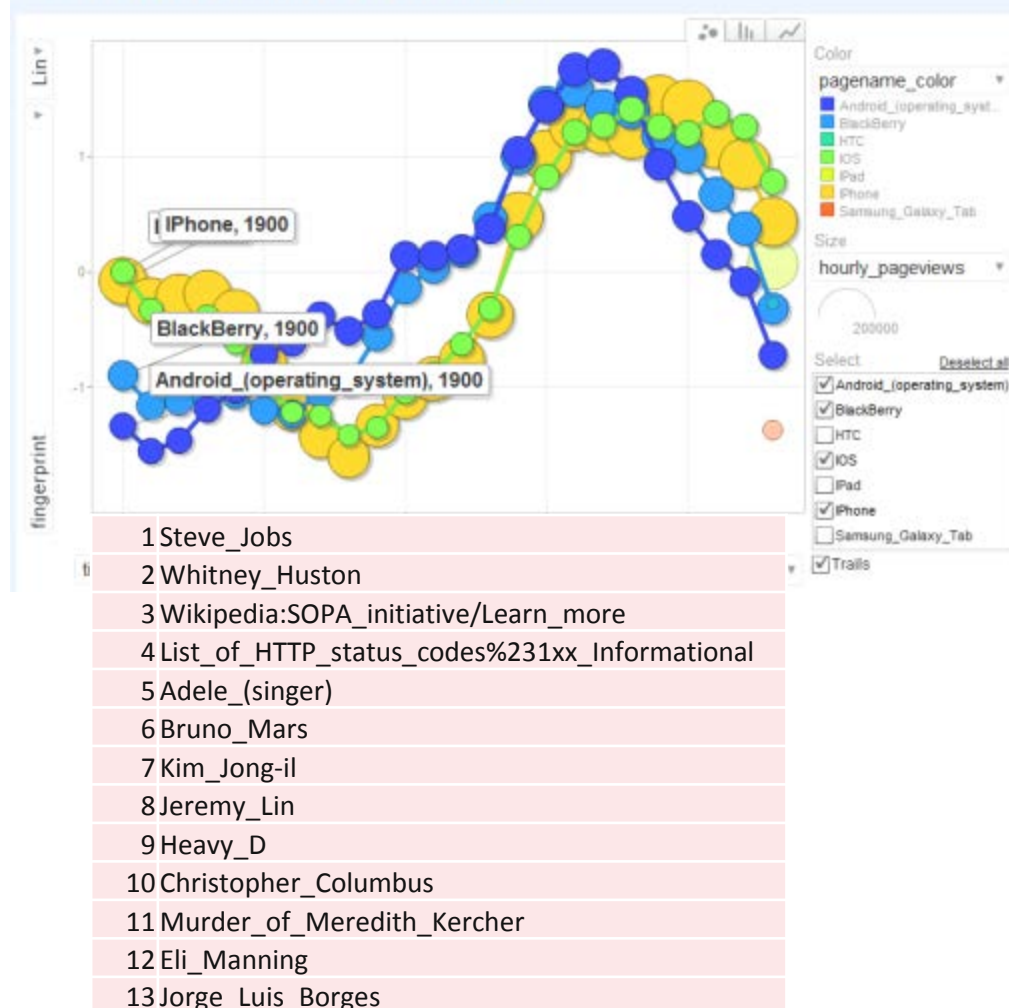
## Result

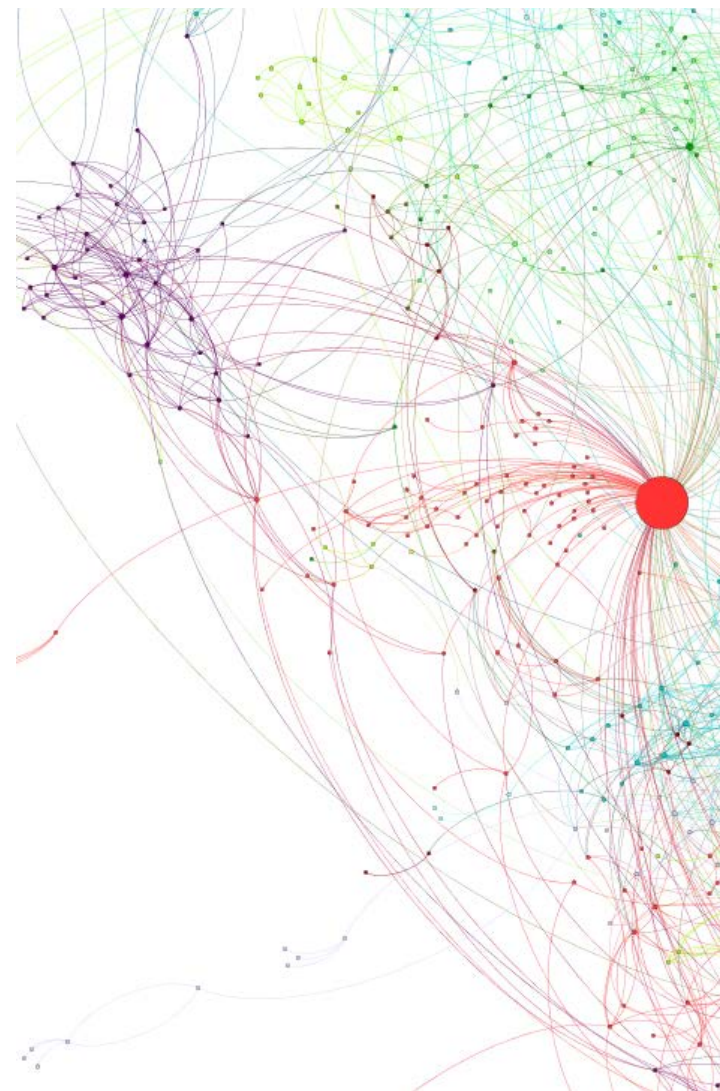
Produces page statistics that can be queried in seconds to visualize which key times of day each Wikipedia page is more actively viewed.

Helps gain insight into both regional and time of day key interest periods in certain topics.

This result can be leveraged with Machine Learning to cluster pages with similar 24hr Fingerprints.

## Wikipedia Hourly Fingerprint





- LexisNexis Public Data Social Graph (PDSG)
  - Public Data relationships.
  - High Value relationships for Mapping trusted networks.
- Large Scale Data Fabrication and Analytics.
  - Thousands of data sources to ingest, clean, aggregate and link.
  - 300 million people, 4 billion relationships, 700 million deeds.
  - 140 billion intermediate data points when running analysis.
- HPCC Systems from LexisNexis Risk Solutions
  - Open Source Data Intensive high performance supercomputer. (<http://hpccsystems.com>)
- Innovative Examples leveraging the LexisNexis PDSG
  - Healthcare.
    - Medicaid\Medicare Fraud.
    - Drug Seeking Behavior
  - Financial Services.
    - Mortgage Fraud.
    - Anti Money Laundering.
    - “Bust out” Fraud.
- Potential Collusion (The value in detecting non arms length transactions)

