TERADATA® ASTER

Efficient Big Data Analytics using SQL and Map-Reduce

Pekka Kostamaa, VP of Engineering and Big Data Lab
ACM Fifteenth International Workshop On Data Warehousing and OLAP
DOLAP 2012 Conference, Maui, Hawaii
November 2, 2012

Agenda

- Aster Data Introduction
- Introduction to nCluster
- SQL-MR Overview
- Word Count Example
- nPath
- SQL-MapReduce® Use Cases

TERADATA®

THE BEST
DECISION
POSSIBLE™

TERADATA® ASTER

aster data

more data. big insights.

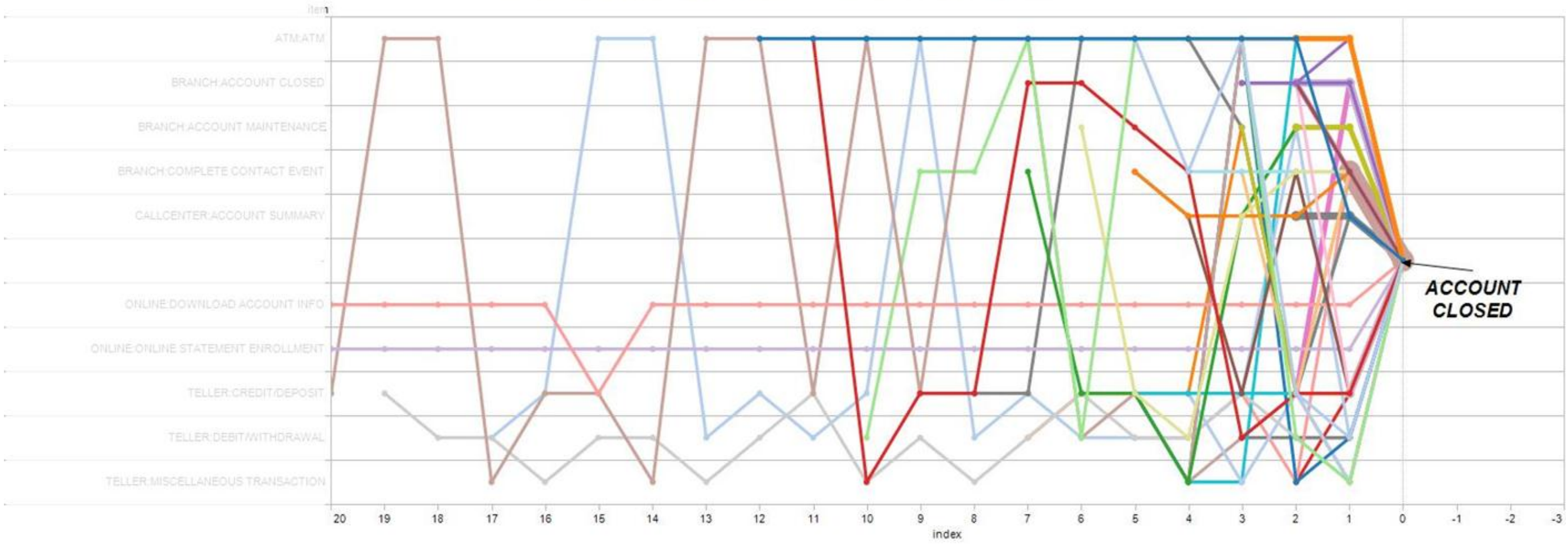
SQL
Database Language
select
RDBMS
delete
views
joines

MapReduce

SQL-MapReduce®

Predictive Analytics with SQL-MR

MULTI-CHANNEL PATHS TO ACCOUNT CLOSURE (REFINED)



```
SELECT * FROM npath (
  ON (
    SELECT …
    WHERE u.event_description IN (
      SELECT aper.event FROM attrition_paths_event_rank aper
      ORDER BY aper.count DESC LIMIT 10)
    )
  …
  PATTERN ('(OTHER|EVENT){1,20}$')
  SYMBOLS (…) RESULT (…)
  )
) n;
```

Interactive Analytics

Agenda

- Aster Data Introduction
- Introduction to nCluster
- SQL-MR Overview
- Word Count Example
- nPath
- SQL-MapReduce® Use Cases

Overview

- Aster Data was founded in 2005, based in SF/Bay Area, focused on the “Big Data” space
- Founders are from Stanford

Tasso Argyros
Co-President, Teradata Aster

Tasso Argyros is Co-President, Teradata Aster, leading the Aster Center of Innovation. Tasso has a background in data management, data mining and large-scale distributed systems. Before founding Aster Data, he was in the Ph.D. program at Stanford University. Tasso was recognized as one of Bloomberg BusinessWeek's Best Young Tech Entrepreneurs for 2009. He holds a Master's Degree in Computer Science from Stanford University and a Diploma in Computer Engineering from Technical University of Athens.

Mayank Bawa
Co-President, Teradata Aster

Mayank Bawa, Co-President, Teradata Aster, leads the Aster Center of Innovation's Research and Development and Customer Support organization devoted to building and supporting Aster Data’s product portfolio. Mayank co-founded Aster Data in 2005 and led it as its CEO from 2005 to 2010, growing the company from three persons to a strong, well-rounded team that shipped products with market-defining features.

Mayank was Aster Data’s Chief Customer Officer from 2010 to 2011, working with customers to do fascinating projects on Big Data.

Mayank has a Bachelor's degree in Computer Science from IIT-Bombay, and a Master's and Ph.D. degree in Computer Science from Stanford University. Mayank was the recipient of the Stanford Graduate Fellowship (Sequoia Capital Fellow) at Stanford University during his doctorate studies. Mayank has published a dozen academic papers in top database conferences and has been affiliated with IBM Research and Microsoft Research.

Overview

- Teradata completed the acquisition of Aster Data in April 2011
  - Expands Teradata analytical capabilities to span emerging analytics on new data types with Aster Data

Customers

Akamai

BARNES&NOBLE

Coremetrics
Relentless Made Easy

GILT GROUPE

LinkedIn

OmnicomGroup

insightexpress
discover tomorrow, today™

intuit.

mzinga

SUPERVALU
AMERICA'S neighborhood grocer

Orchard Supply
EST. 1931
HARDWARE

Gates

myspace.

Chegg.com

U.S. DEPARTMENT OF ENERGY

zazzle

at&t

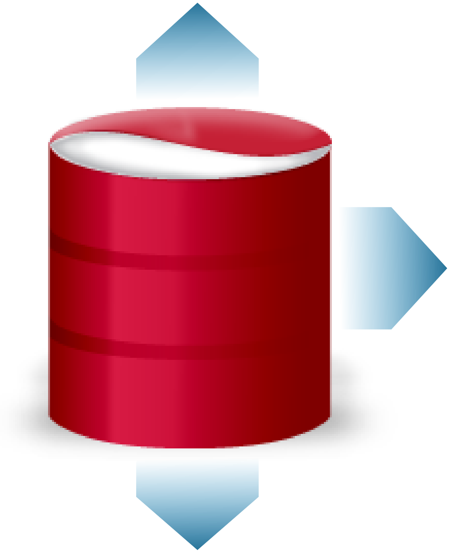
eightfold logic

What is Aster Data’s Solution?

A highly scalable Discovery Platform for Big Data

Scale and speed of analytics

- Accelerated SQL querying
- Rich analytics on large data sets (MapReduce, SQL-MapReduce)
- Out-of-the-box analytic functions (Time-series, Sessionization …)



Ultra-fast, efficient access to data

- Analytics applications run *inside* the DB eliminating data movement
- Push down any analytics application, packaged or custom

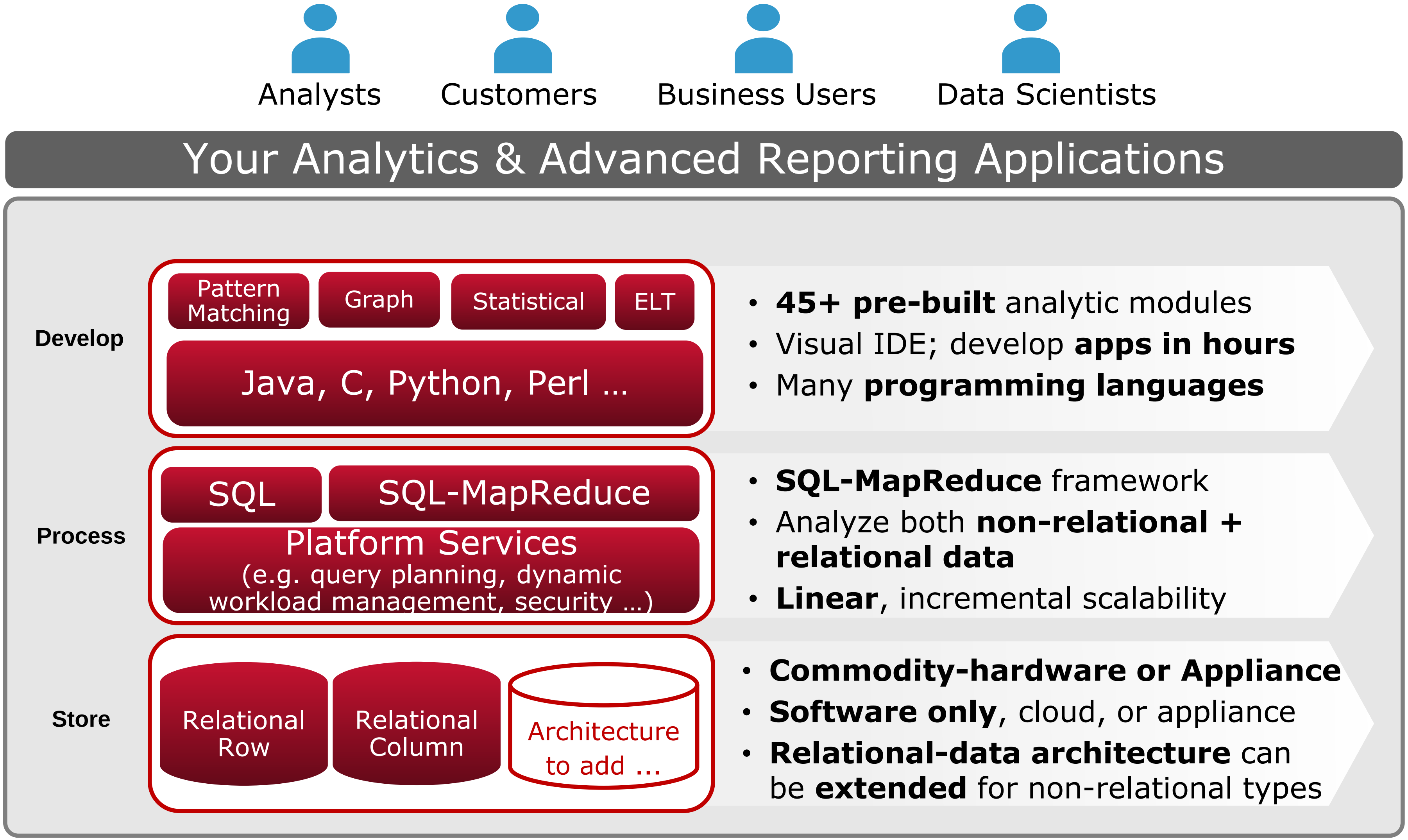
Highly scalable, high performance

- Always On
- Always Parallel
- High Concurrency

Exceptional price/performance

- MPP, shared-nothing architecture
- Teradata Appliance
- Commodity hardware

A Look Inside the Aster Data Analytic Platform



Analysts

Customers

Business Users

Data Scientists

Your Analytics & Advanced Reporting Applications

Develop

Pattern Matching | Graph | Statistical | ELT

Java, C, Python, Perl …

- **45+ pre-built** analytic modules
- Visual IDE; develop **apps in hours**
- Many **programming languages**

Process

SQL | SQL-MapReduce

Platform Services (e.g. query planning, dynamic workload management, security …)

- **SQL-MapReduce** framework
- Analyze both **non-relational + relational data**
- **Linear**, incremental scalability

Store

Relational Row | Relational Column | Architecture to add …

- **Commodity-hardware or Appliance**
- **Software only**, cloud, or appliance
- **Relational-data architecture** can be **extended** for non-relational types

TERADATA® ASTER

Introduction to SQL/MR

SQL/MR: Goals

- **Motto**: “a UDF for the 21st century”
- **Scalable**
  - It should be easy to leverage hardware resources of 100’s of servers
  - Fault-tolerance should be handled by the system
- **Analyst-friendly**
  - Want flexible, declarative language for analysts
  - Enable developers to create widely-reusable tools for analysts
  - Semantics of queries should not get mixed up with implementation details
- **Developer-friendly**
  - Want straightforward programming model
  - Provide useful platform services to developer to maximize freedom

What is SQL/MR?

SQL/MapReduce (SQL/MR) is Aster's framework for enabling the execution of user code within *n*Cluster:

- User code is installed in the cluster, and then it's invoked on database data from SQL.
- Execution is automatically parallelized across the cluster.
- Supports Java and C as primary languages, but other languages are supported through the Streaming interface.

Basic Primitive: SQL/MR Function

- Self-describing, parallelized, relation-to-relation transform
- Operates on an input relation
  - Input relation (table, sub-query, etc.) specified by query
  - By default, allow any input schema; function can choose to reject
  - Input rows accessed row-wise or partition-wise (group-wise)
- Emits an output relation
  - Function can be used in query like any relation; joined, aggregated, etc.
- Free-form transformation
  - Output schema chosen by function; based on input schema, argument clauses, external considerations, etc.
  - No enforced relationship between rows in input and output

Example: Word Count

What is Map/Reduce?

- Map/Reduce is a parallel programming paradigm.
- A parallel computation is specified in Map/Reduce by defining two functions: **map** and **reduce**.
- Independent of any particular implementation.
- The first implementation of a Map/Reduce system for dealing with large data was built at Google in 2001.

Word Count in Map/Reduce

- Goal: find frequencies of words in a set of documents
  - Canonical “teach MapReduce” example since Dean and Ghemawat’s paper
- Input data set:
  - Documents (docid int, body text)
- Visually:

docid	body
1	‘Jack and Jill went up the hill to fetch a pail of water. Jack fell down and broke his crown, and Jill came tumbling after.’
2	‘Little Miss Muffet sat on a tuffet, eating her curds and whey. Along came a spider, who sat down beside her, and frightened Miss Muffet away!’
3	‘The itsy bitsy spider went up the water spout. Down came the ran and washed the spider out. Out came the sun, and dried up all the rain. And the itsy bitsy spider went up the spout again.’

word	count
went	3
hill	1
up	4
down	3
spider	4
tuffet	1
…	

Word Count in Map/Reduce

- MapReduce performs (logically) in two steps
  - Map: for each document, tokenize the document body into <*word*, *c*>, where *c* is just the count in that one document
  - Reduce: for each word, sum up all of the counts, and emit <*word*, *count*>
- MapReduce performs (physically) in three steps:
  1. Map, on each document independently
  2. Shuffle, to bring the partial counts to one place, for each word
  3. Reduce, on each word (and its partial counts) independently
- Easy to parallelize
  - Map task can run on each document independently
  - Reduce task can run on each word independently

Input: The Documents Table

```
BEGIN;

CREATE FACT TABLE documents (docid int, body text,
   PARTITION KEY(docid));

INSERT INTO documents VALUES (0, ‘this is a single test
   document. it is simple to count the words in this single document
   by hand. do we need a cluster?’);

END;

SELECT body FROM documents;
```

Map Function: tokenize

```
public class tokenize implements RowFunction {
   ...

  public void operateOnSomeRows(RowIterator inputIterator,
     RowEmitter outputEmitter)
  {
    while ( inputIterator.advanceToNextRow() ) {
      String[] parts =
        splitPattern_.split(inputIterator.getStringAt(1));

      for (String part : parts) {
        outputEmitter.addString(part);
        outputEmitter.addInt(1);
        outputEmitter.emitRow();
      }
    }
  }
}
```

Reduce Function: count_tokens

```
public class count_tokens implements PartitionFunction {
   ...

  public void operateOnPartition(
    PartitionDefinition partitionDefinition,
    RowIterator inputIterator, RowEmitter outputEmitter)
  {
    int count = 0;
    String word = inputIterator.getStringAt(0);

    while ( inputIterator.advanceToNextRow() )
      count++;

    outputEmitter.addString(word);
    outputEmitter.addInt(count);
    outputEmitter.emitRow();
  }
}
```

Invoking the Functions

```
\install tokenize.jar
\install count_tokens.jar

SELECT word, count FROM count_tokens (
   ON ( SELECT word, count
        FROM tokenize(ON documents))
   PARTITION BY word
) ORDER BY word DESC;
```

Even Better: Forget the Reduce

```
\install tokenize.jar

SELECT word, sum(count)
   FROM tokenize(ON documents)
GROUP BY word
ORDER BY word;
```

Types of SQL/MR Functions

- *RowFunction*
  - Corresponds to a *map* function.
  - Must implement the *operateOnSomeRows* method.
  - Must be invoked without a PARTITION BY.
  - “Sees” all the appropriate rows on a particular worker.

- *PartitionFunction*
  - Corresponds to a *reduce* function.
  - Must implement the *operateOnPartition* method.
  - Must be invoked with a PARTITION BY, which specifies how rows are reshuffled.
  - “Sees” all the appropriate rows in a partition.

TERADATA® ASTER

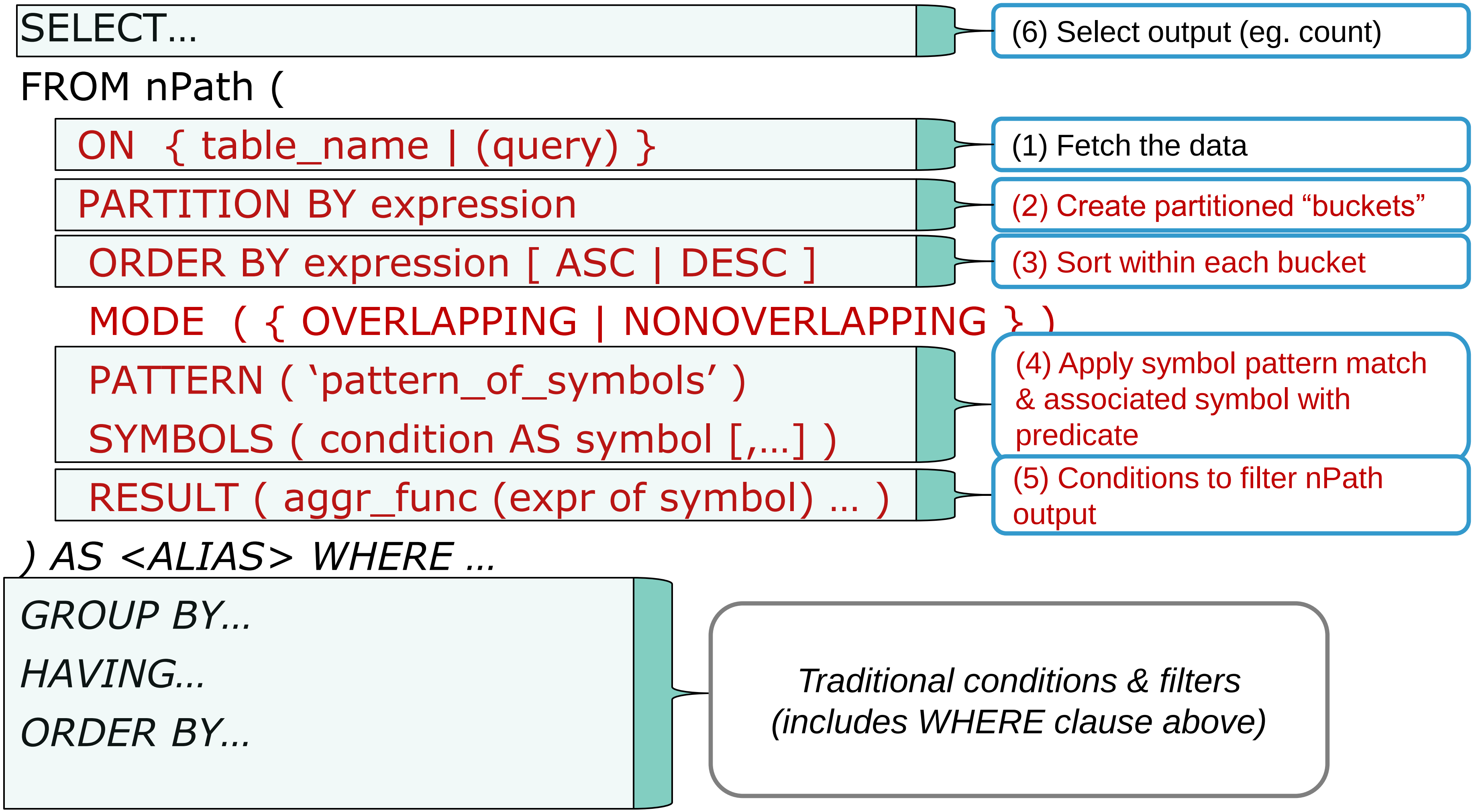
Introduction to nPath

What is *n*Path?

- *n*Path is a SQL/MR function included with *n*Cluster.
- *n*Path enables analysis of *ordered* data:
  - Clickstream data
  - Financial transaction data
  - User interaction data
  - Anything of a time series nature

Leverages the power of the SQL/MR framework to transcend SQL's limitations with respect to ordered data.

*n*PATH Syntax



```
SELECT...
FROM nPath (
    ON  { table_name | (query) }
    PARTITION BY expression
     ORDER BY expression [ ASC | DESC ]
     MODE  ( { OVERLAPPING | NONOVERLAPPING } )
     PATTERN ( 'pattern_of_symbols' )
     SYMBOLS ( condition AS symbol [,...] )
     RESULT ( aggr_func (expr of symbol) ... )
) AS <ALIAS> WHERE ...
GROUP BY...
HAVING...
ORDER BY...
```

(6) Select output (eg. count)

(1) Fetch the data

(2) Create partitioned “buckets”

(3) Sort within each bucket

(4) Apply symbol pattern match & associated symbol with predicate

(5) Conditions to filter nPath output

Traditional conditions & filters (includes WHERE clause above)

Review: The Four *n*Path Steps

1. **Partition** the source data into groups.

2. **Order** each group to form a sequence.

3. **Match** subsequences of interest.
   a. Define a set of symbols via predicates.
   b. Define the subsequences of interest via a regular expression of symbols.

4. **Compute aggregates** over each matching subsequence.

TERADATA® ASTER

Aster Data Customer Use Cases

TERADATA® ASTER

Retail Banking: “Last Mile” Marketing

Aster in Retail Banking: “Last Mile” Marketing

Challenge

- Know the “last mile” of a decision
- Data Mining tools predict probability but do not ID the “last mile”

With Aster

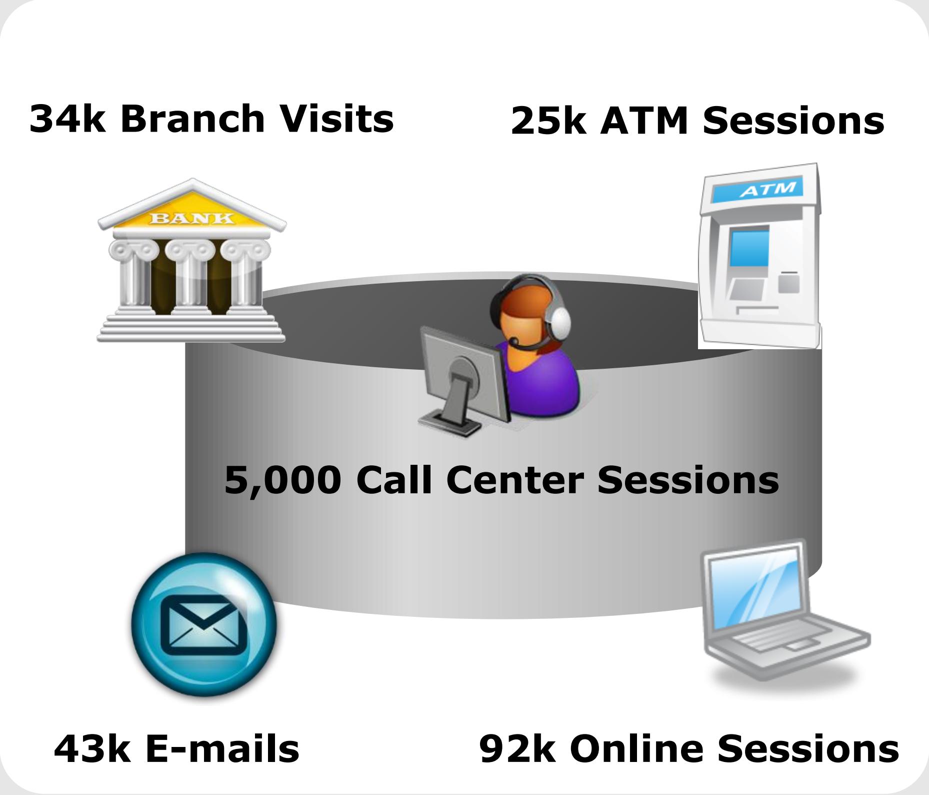
- SQL-MapReduce listens and predicts the “last mile”
  - Identifies all interaction patterns prior to acquisition or attrition

Business Impact

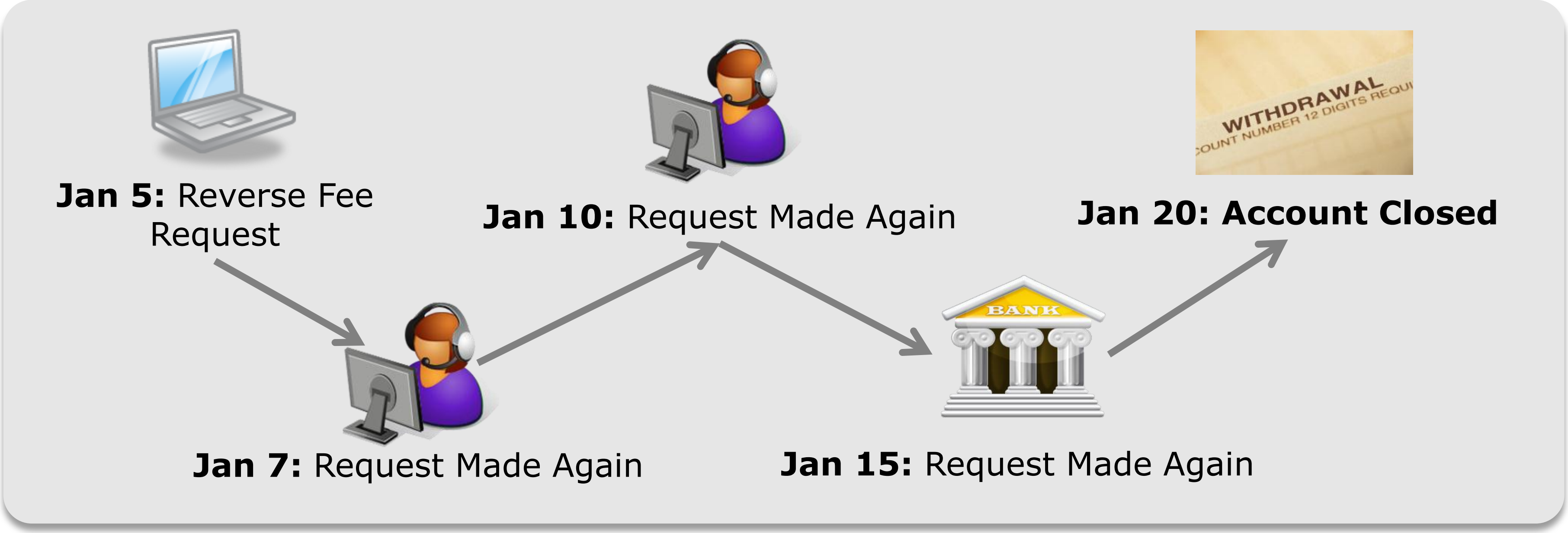
- **10-300x** less effort to pinpoint a customer in the “last mile”

Cross-Channel Customer Interactions

17,000 Customers, 1 Month



Aster MapReduce: Understanding the "Last Mile"



Jan 5: Reverse Fee Request

Jan 7: Request Made Again

Jan 10: Request Made Again

Jan 15: Request Made Again

Jan 20: Account Closed

What if I knew that this customer was likely to leave? I could…

- Apologize
- Offer an explanation
- Reverse the $5 fee

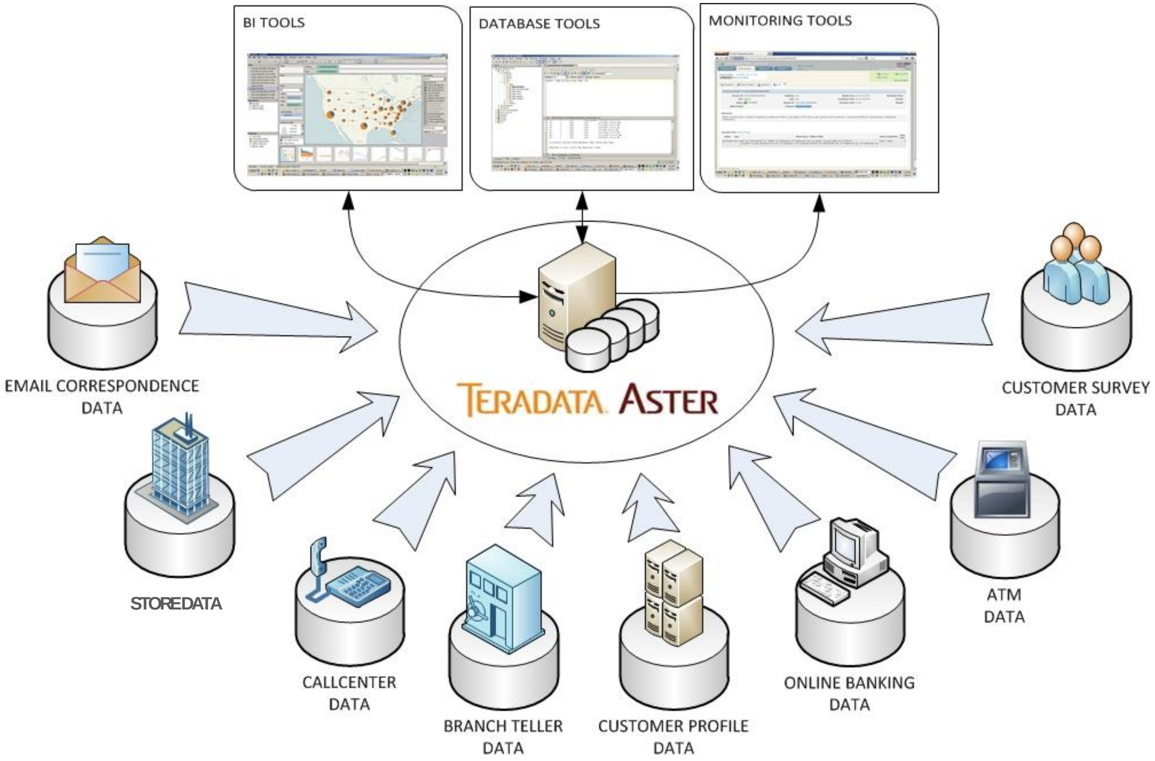
"It takes 3x more to acquire a customer than to retain one"

Multi-Channel Customer Analysis

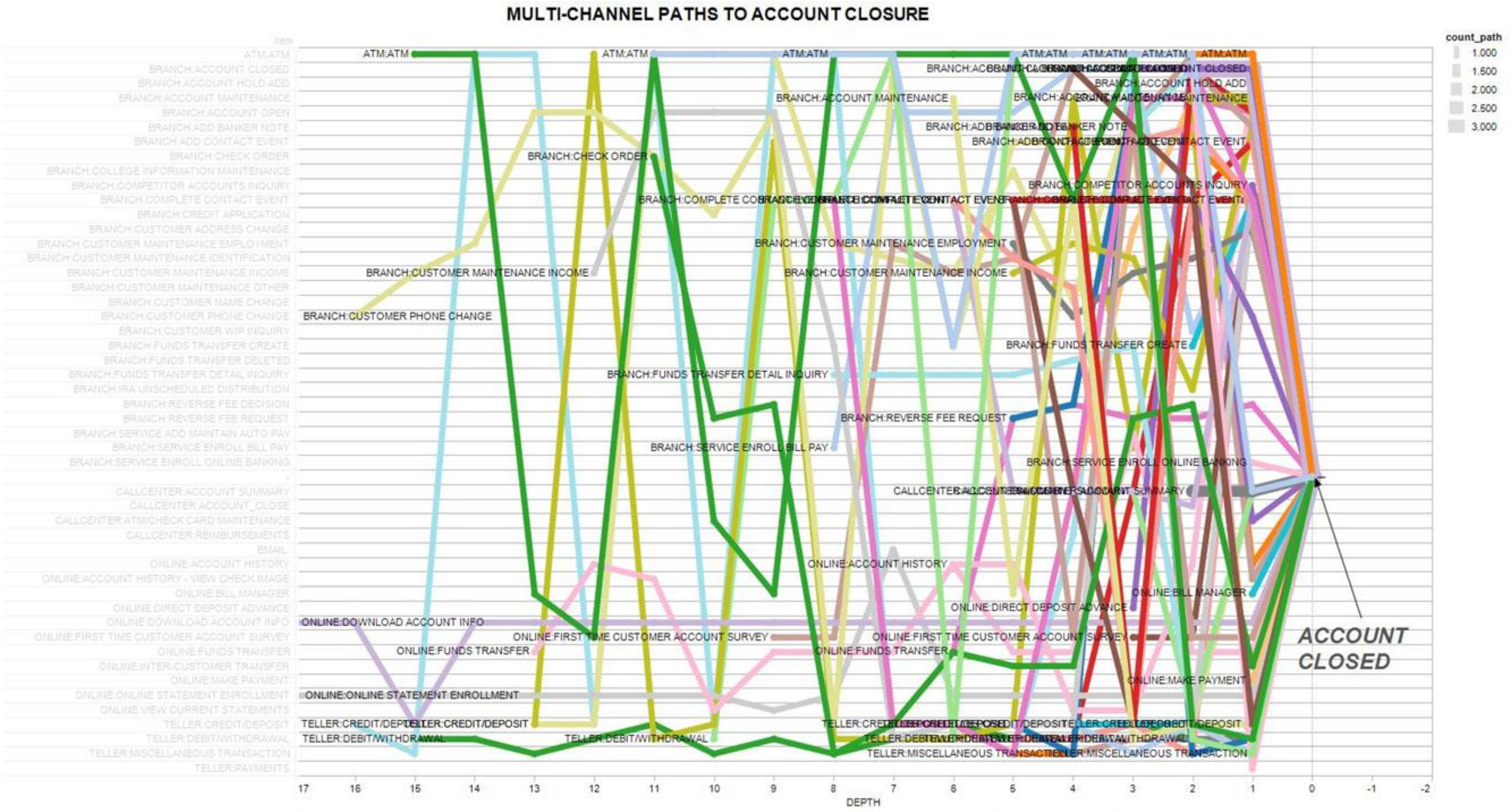
Iterative Discovery Analytics

Business Question(s):

- Is there any identifiable pattern of behavior prior to account closure?
- Prior to new product additions?
- If so, what does this pattern look like?

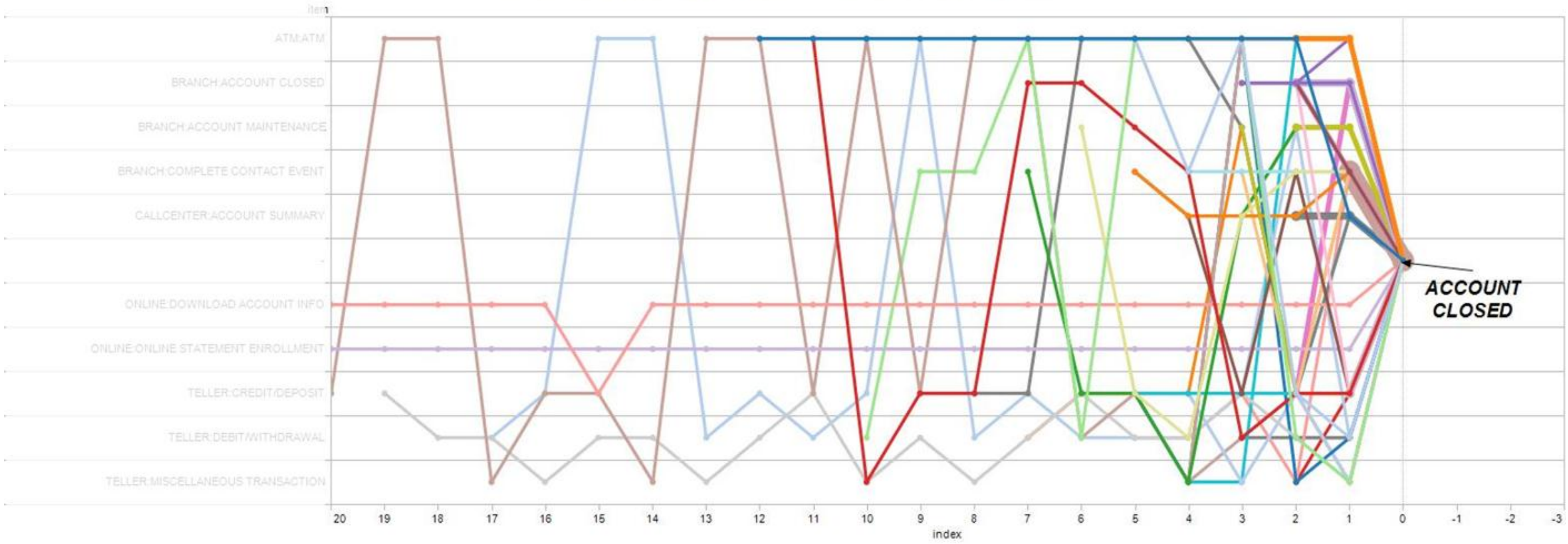


Events Preceding Account Closure



The trend of depth for event. Color shows details about the specific event path. Size shows details about the frequency of the specific event path. The marks are labeled by event. Details are shown for the frequency of the specific path and event.

Events Preceding Account Closure

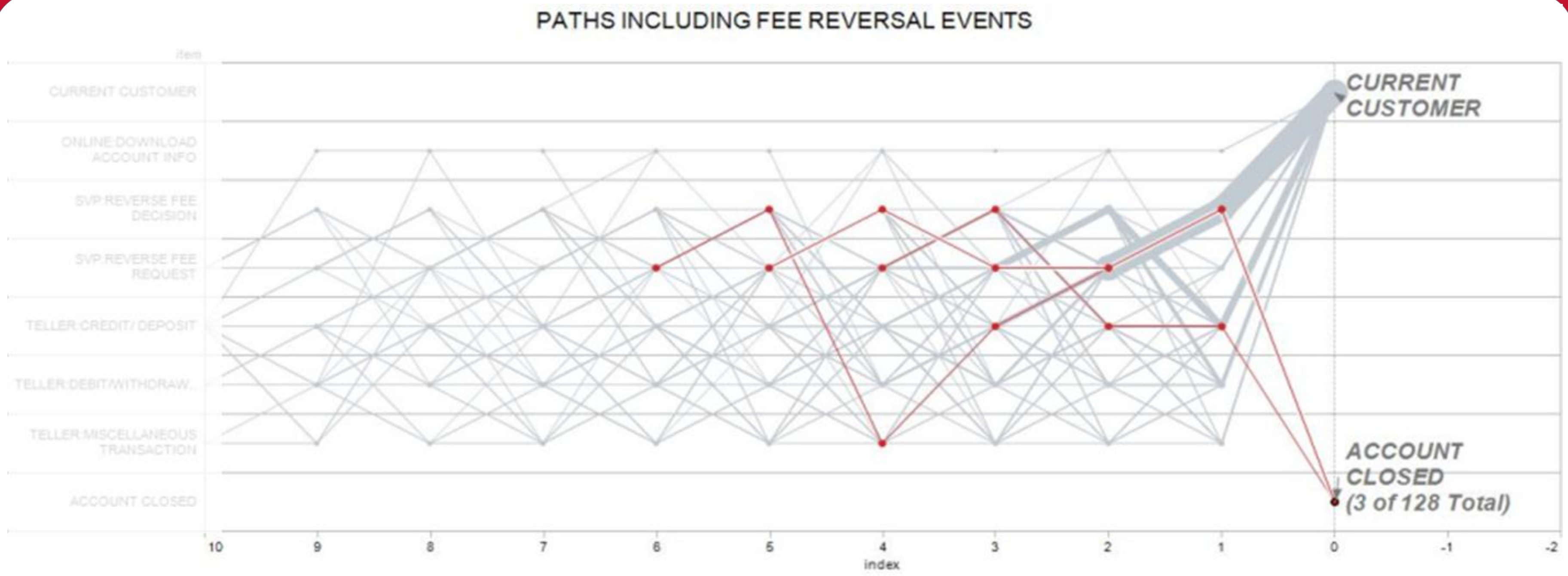

MULTI-CHANNEL PATHS TO ACCOUNT CLOSURE (REFINED)

```
SELECT * FROM npath (
  ON (
    SELECT …
    WHERE u.event_description IN (
      SELECT aper.event FROM attrition_paths_event_rank aper
      ORDER BY aper.count DESC LIMIT 10)
    )
  …
  PATTERN ('(OTHER|EVENT){1,20}$')
  SYMBOLS (…) RESULT (…)
  )
) n;
```

Interactive Analytics

Reducing the "Noise" to find the "Signal"

Events Preceding Account Closure



PATHS INCLUDING FEE REVERSAL EVENTS

```
SELECT *
FROM nPath (
  ON (…)
  PARTITION BY sba_id
  ORDER BY datestamp
  MODE (NONOVERLAPPING)
  PATTERN ('(OTHER_EVENT|FEE_EVENT)+')
  SYMBOLS (
    event LIKE '%REVERSE FEE%' AS FEE_EVENT,
    event NOT LIKE '%REVERSE FEE%' AS  OTHER_EVENT)
  RESULT (…)
) n;
```

Closed Accounts

Fee reversal seems to be a "Signal"

How did Big Data Help?

1. Collect & analyze not only customer transactions, but also customer interactions

2. Use SQL-MapReduce pre-built operators to identify behavioral patterns and uncover business insight

3. Utilize standard BI tools to ensure insights are consumed by the right business analysts and acted upon

Big Data for Business = ***use more data and more analytics to achieve a competitive edge***

TERADATA® ASTER

Multi-Touch Attribution

Optimize Marketing Spend to Drive Higher ROI

Strategy for Success

FIRST CLICK
LAST CLICK
MULTI-TOUCH

Where should I increase my Marketing Spend to drive higher ROI?

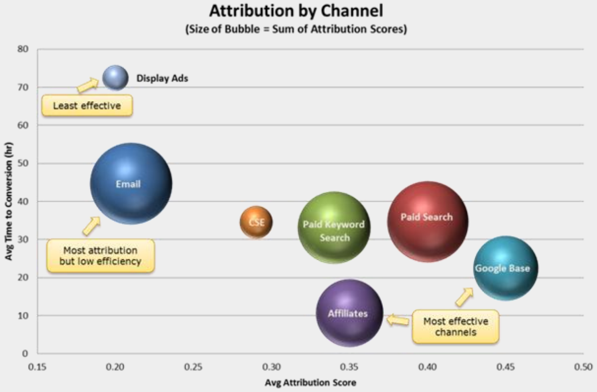
Multi-Touch Attribution

Go beyond “last click” and identify paths customers take to conversion

Quantify channels effectiveness (**attribute**) to drive revenue

Identify which channels perform the best

Time sensitive - campaigns and channels to drive rev in the immediate short term


Attribution by Channel

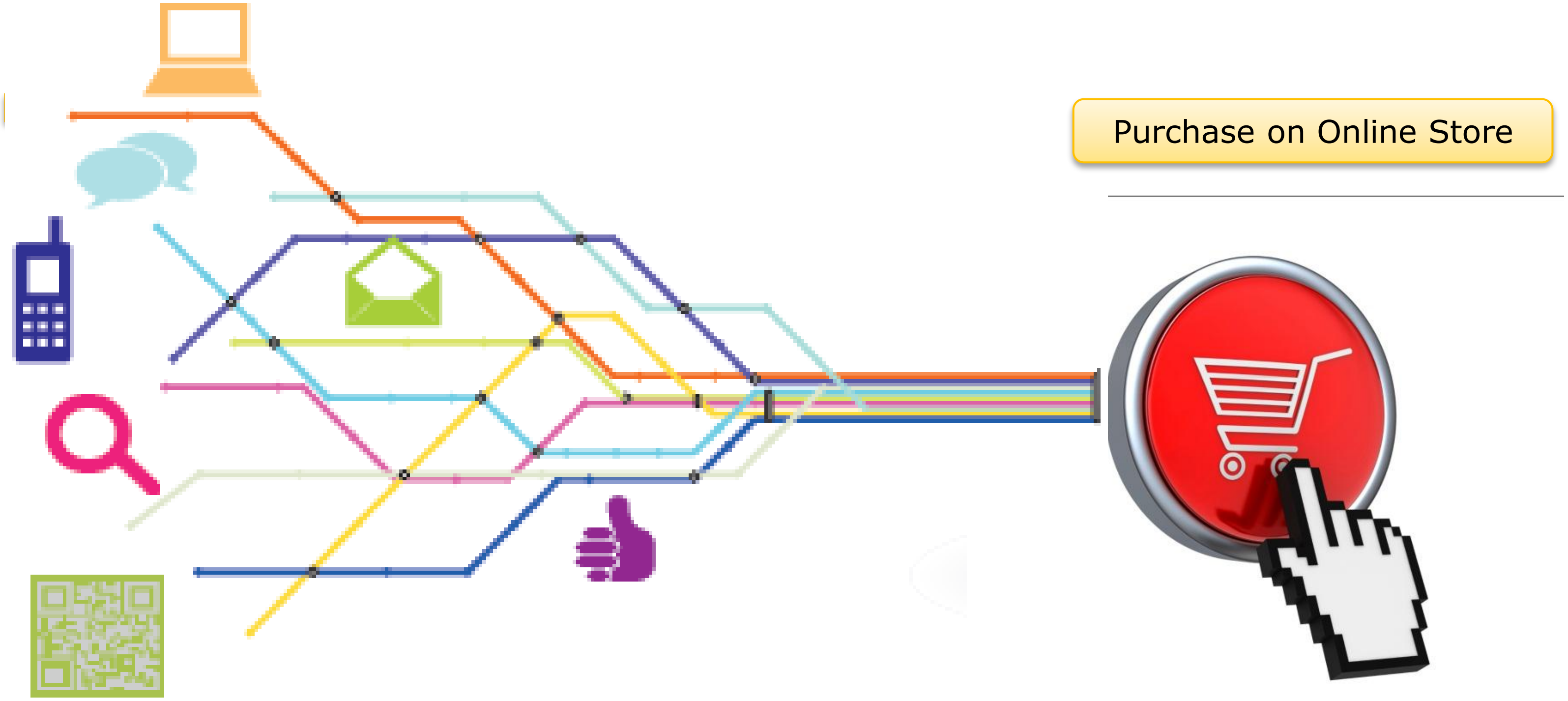
(Size of Bubble = Sum of Attribution Scores)

Least effective

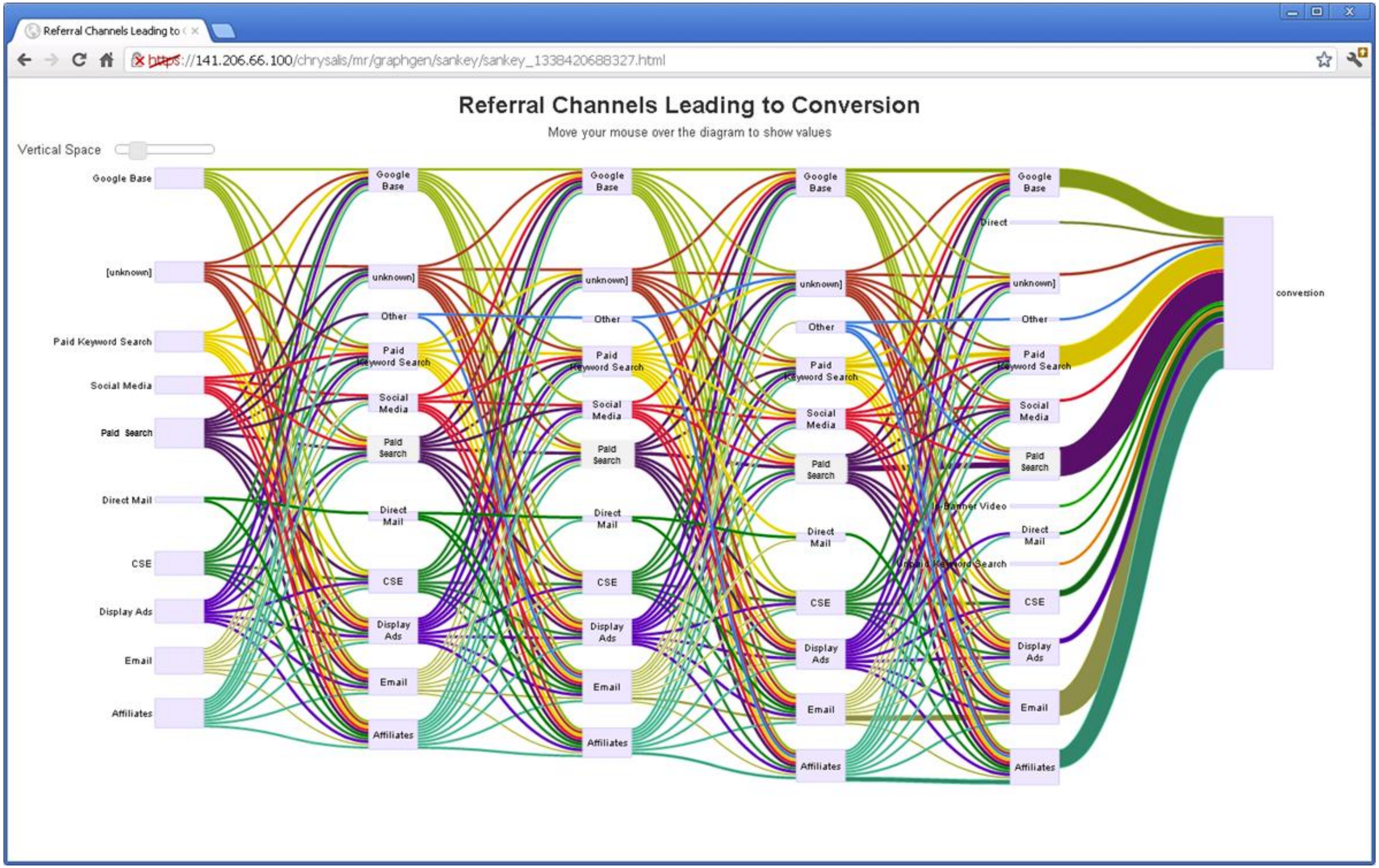
Most attribution but low efficiency

Most effective channels

Customer Journey Leading to Purchase on Online Store



Customer Path Analysis and Referral Channels



https://141.206.66.100/chrysalis/mr/graphgen/sankey/sankey_1338420688327.html

Referral Channels Leading to Conversion

Move your mouse over the diagram to show values

Attributing Revenue to Various Channels

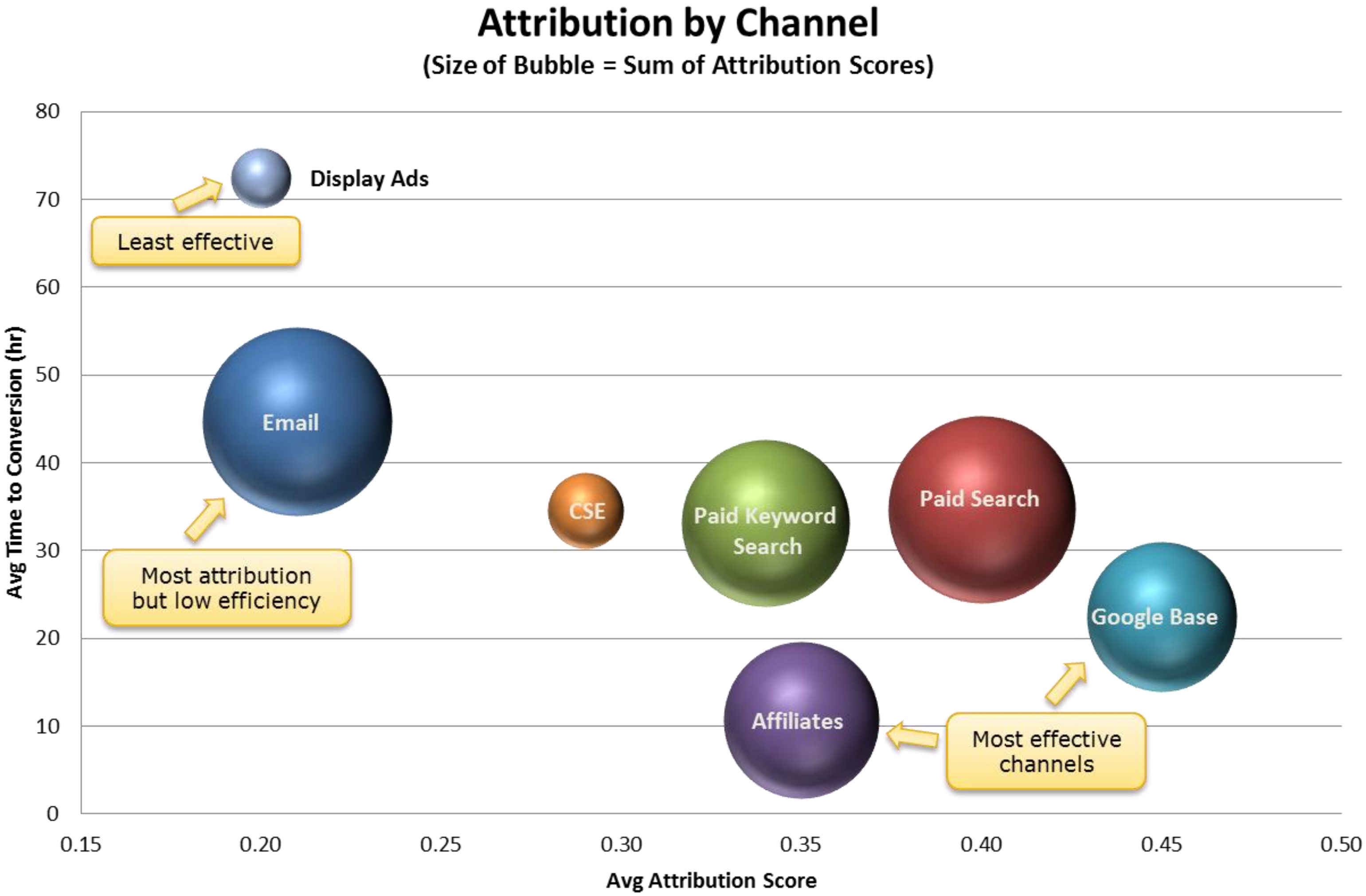
Attribution Function

```
EVENT_COLUMN_NAME('event')

CONVERSION_EVENT_TYPE_VALU
E('purchase')

EXCLUDING_EVENT_TYPE_VALUE
('website')

TIMESTAMP_COLUMN_NAME('seq
uence')
    WINDOW('rows:10')

MODEL1('SIMPLE','EXPONENTI
AL:.5')
```

Attributed Revenue

Sequence	Event	Attribution	Rev
10	Purchase		
9	Email	0.533	$133.33
8	Registration	0.267	$66.67
2	Ad Click	0.133	$33.33
1	Google Impression	0.067	$16.67

Attribution by Channel



Least effective

Most attribution but low efficiency

Most effective channels

Strategy for Success

Multi-Touch Attribution

Go beyond “last click” and identify paths customers take to conversion

Quantify channels effectiveness (**attribute**) to drive revenue

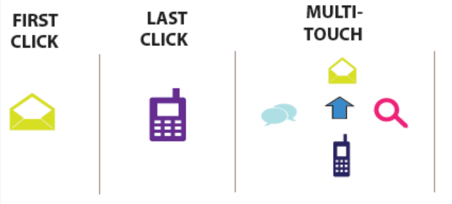
Identify which channels perform the best

Time sensitive - campaigns and channels to drive rev in the immediate short term

Business Impact / ROI

Calculate true ROMI on a per channel basis

Run time-sensitive promotions by knowing which channels convert the fastest.



Summary

- Teradata Aster Platform
- Power of SQL and MapReduce in a single, easy to use platform
  - SQL/MR
  - nPath
- Support for multiple types of relational and non-relational data
- Discovery Analytics to drill down