Relevant Data Analysis: Apache Solr Analytics

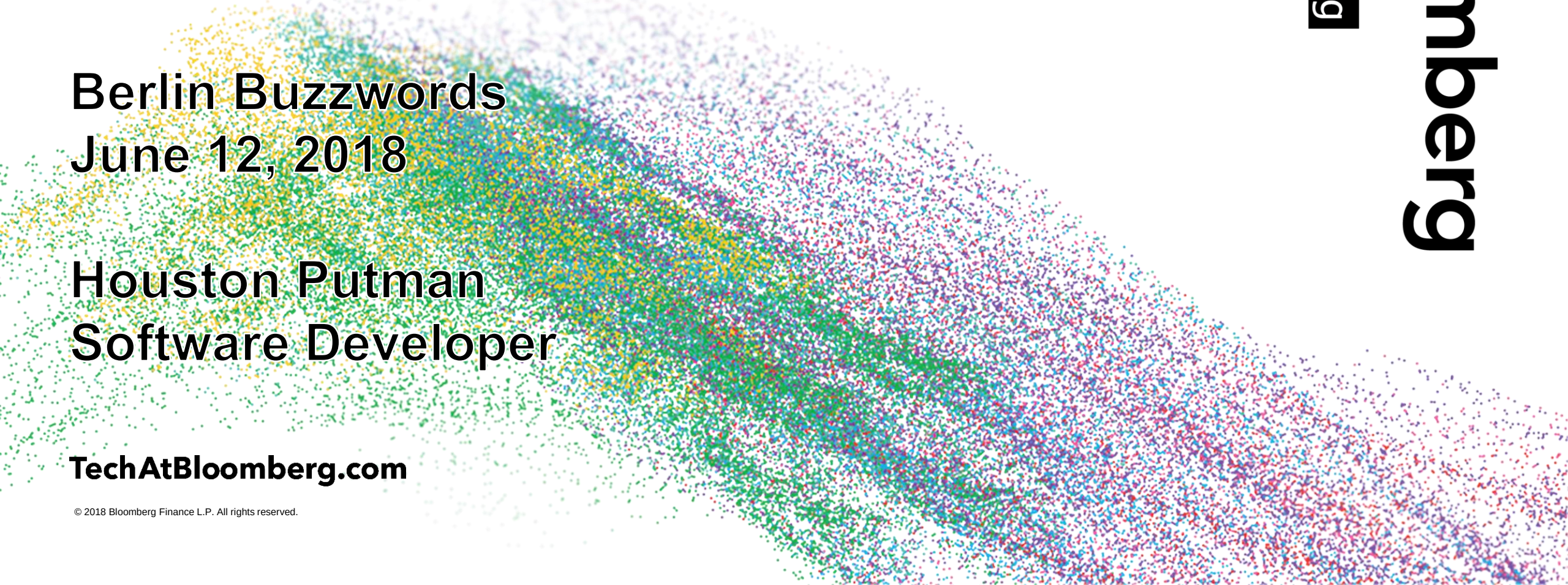
Bloomberg

Engineering

Berlin Buzzwords
June 12, 2018

Houston Putman
Software Developer

TechAtBloomberg.com



Bloomberg at a glance

- Provider of global financial news and information
- Our strength is quickly and accurately delivering data, news and analytics
- Creating high performance and accurate information retrieval systems is core to our strength
  - Stability is key, as downtime can cost clients
- Over 5,000 software engineers
- Many diverse challenges that require different approaches to data analysis

TechAtBloomberg.com

Bloomberg

Engineering

Agenda

- Relevance and Analytics
- Solr Analytics
- Distributed Analytics
- Performance Considerations
- Additional Features
- Use Cases

Combining Relevance and Analytics

- There are many established analytics engines available, such as Spark and Hadoop
- Solutions have been proposed to combine Solr with these projects in order to leverage search capability with analytics
- Using external analytics engines requires exporting the needed data set from Solr
  - The benefits of using external analytics engines come from analyzing large amounts of data; therefore, most problems you need Spark to solve will require long exporting tasks
- Spark and Hadoop have many tools for data scientists to play with data

Solr Analytics

- Using an internal analytics engine, such as the Analytics component allows you to perform complex data introspection without spending the time of exporting data from Solr
  - Solr Analytics was built using map-reduce principles
- Using an internal engine reduces the complexity of the data pipeline
  - Ready to use with any Solr Cloud
- Solr is as live as the data ingested into it

Why do we need analytics?

- I want to analyze the performance of a baseball player over the past season
  - A search engine would return a list of plate appearances and what happened during each

Date	Player	Inning	Plate Appearances (PA)	Walk	1B	2B	3B	HR
17-05-01	Altuve	1	True		1			
17-05-01	Springer	1	True	1				
17-05-01	Correa	1	True					
17-05-01	Altuve	4	True					
17-05-01	Springer	4	True			1		

- However, these individual records don't help me understand how well each player did
  - The result of each plate appearance is as much luck as skill
  - Even the worst hitters will most likely hit a HR sometime in their career

How do we analyze data?

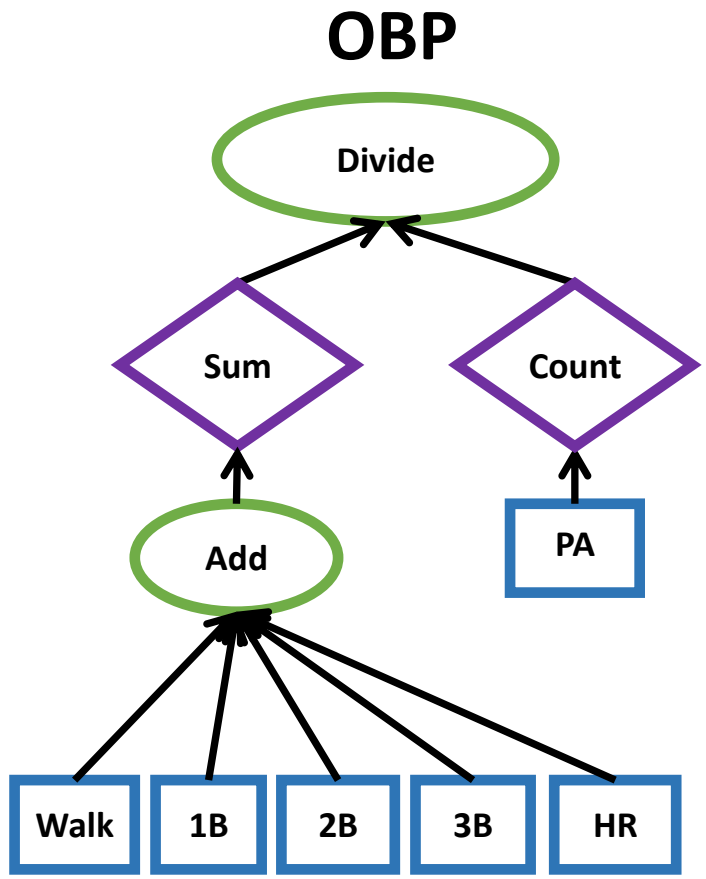
- For large data sets, meaning is found in the aggregate
  - In order to find meaning in the data, we need to combine it
- How do you combine documents?
  - Analytical expressions
- How exactly is baseball performance measured?
  - OBP (On Base Percentage) = (Walk + 1B + 2B + 3B + HR) / PA
  - AVG (Batting Average) = (1B + 2B + 3B + HR) / (PA – Walk)

Mapping vs. Reducing

- First, let's query some Astros results
- Mapping functions combine values within documents
- Reduction functions combine data across documents
- Mapping functions also combine the results of reductions

Date	Player	Inning	PA	Walk	1B	2B	3B	HR	Add(…)
17-05-01	Altuve	1	True		1				1
17-05-01	Springer	1	True	1					1
17-05-01	Correa	1	True						0
17-05-01	Altuve	4	True						0
17-05-01	Springer	4	True			1			1

			Count						Sum	Divide(…)
			5						3	.600



OBP

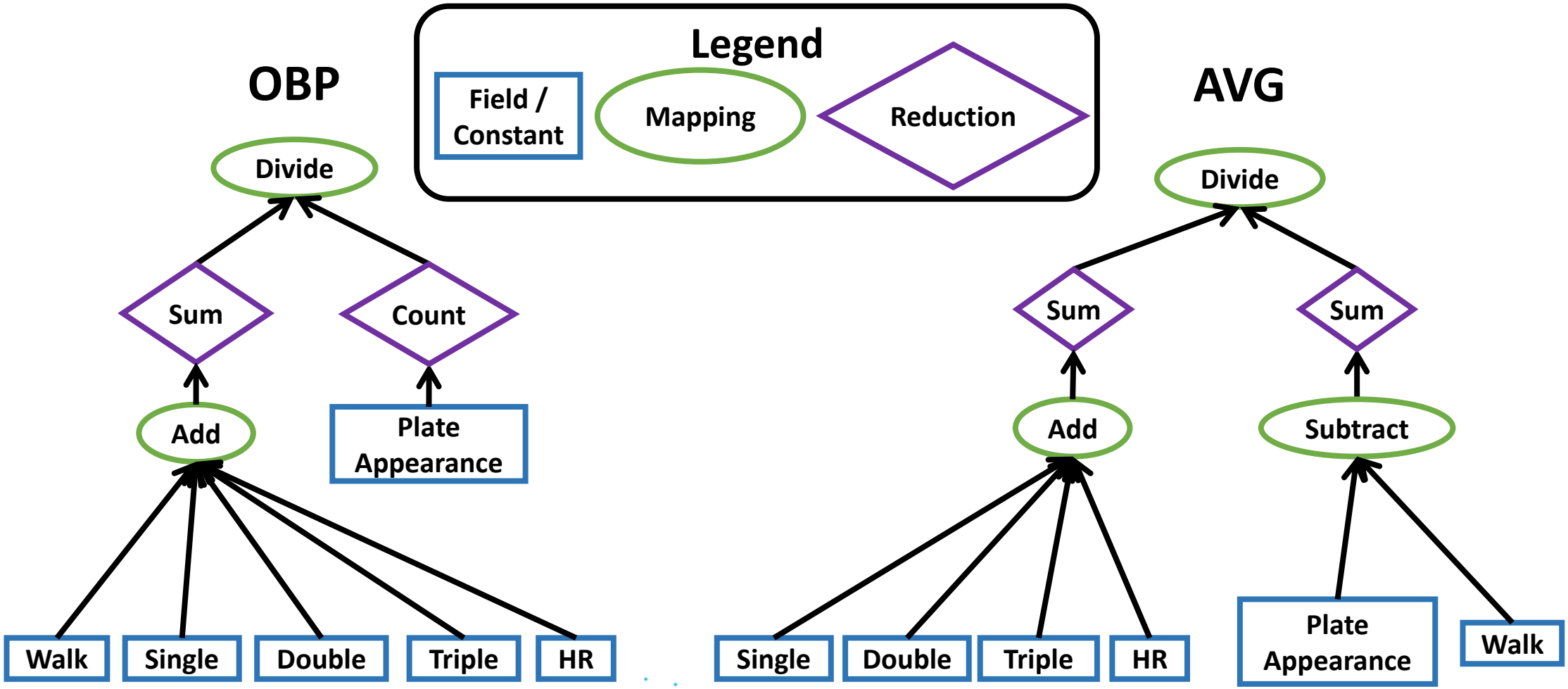
TechAtBloomberg.com

Bloomberg
Engineering

Mapping vs. Reducing

- Built using the same principals as Hadoop and Spark
- Allows for parallelization

Analytics expressions



TechAtBloomberg.com

Bloomberg

Engineering

Facets

- With a diverse set of data, we need to be able to break up the data in order to analyze it
  - These analytics are only interesting when comparing results across different:
    - Players
    - Years
    - Teams
- Facets allow us to group the data and calculate analytics on each group separately
  - Group Jose Altuve's results by year
  - Group 1998 statistics by team

Types of Facets

- Value Facet – Group the data by a field or mapping
  - Player
- Range Facet – Group the data by a defined set of ranges
  - Date Ranges (Date)
    - **May**
    - **June**
  - Value Ranges (Innings)
    - **1-3**
    - **4-6**
- Query Facet – Group the data by extra queries
  - **Cold games (< 15° AND NOT Indoor)**
  - **Hot games (> 35° AND NOT Indoor)**

Date	Player	Inning	Temp	Indoor
2017-05-01	Altuve	1	10°	F
2017-05-01	Springer	1	10°	F
2017-05-01	Correa	1	10°	F
2017-05-01	Altuve	4	10°	T
2017-05-01	Springer	4	10°	T
2017-06-02	Bregman	7	38°	F
2017-06-02	Reddick	7	38°	F
2017-06-02	Gattis	7	38°	F
2017-06-05	Altuve	3	25°	F

Value Facets

- Value Facets replicate the functionality of Solr Field Facets
- Adds the ability to facet over any mapping expression
  - The expression cannot contain a reduction function
  - **`expression: if(atHome, team, opposingTeam)`**
- Allows complex sorting
  - Multiple sorting criteria accepted, including sorting by facetValue or by the result of an expression
  - Setting a limit and offset after the sorting has been done

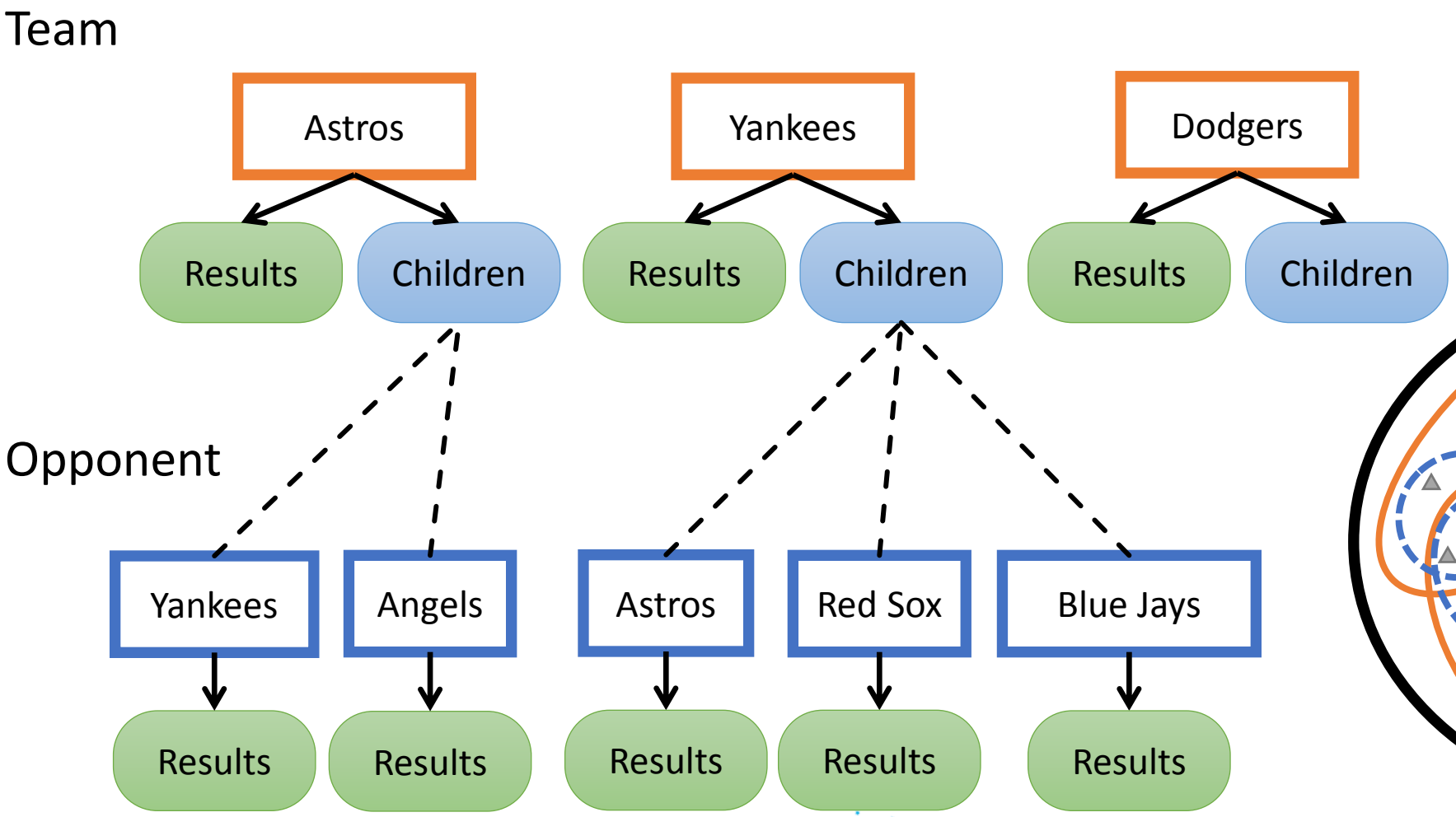
Value Facet Example Stadium HRs

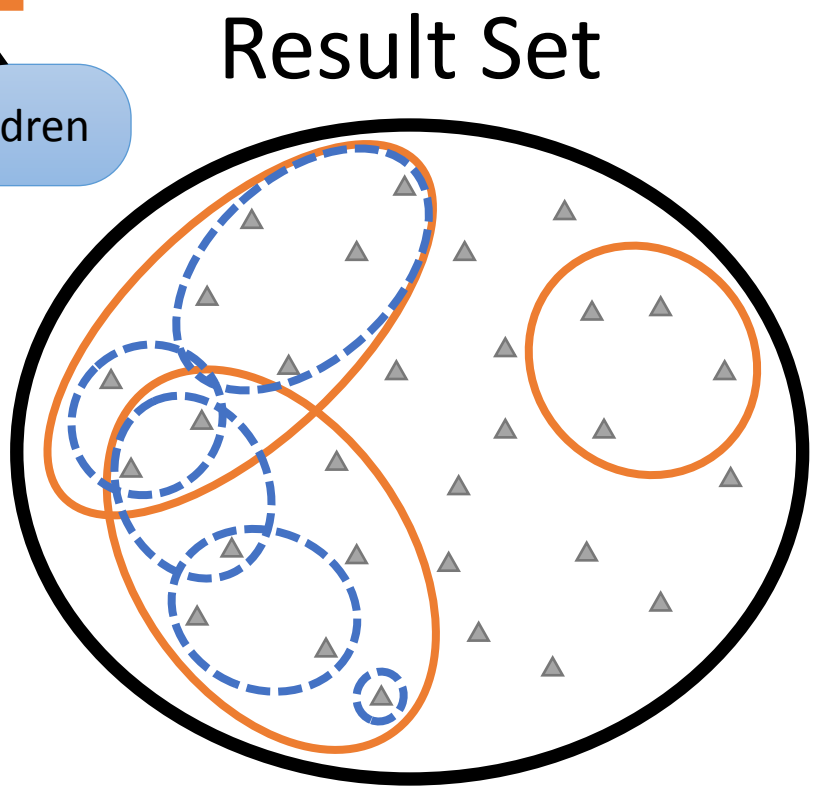
```
{
    type : "value",
    expression : "if(atHome, team, opposingTeam)",
    sort : {
        criteria : [
            {
                type : "expression",
                expression : "homeRunCount",
                direction : "descending"
            },
            {
                type : "facetValue"
            }
        ],
        limit : 15,
        offset : 0
    }
}
```

Pivot Facets

- Pivot Facets allow drill-down faceting through multiple mapping expressions
- Much like Solr pivot facets, with a few differences
  - Like value facets, analytics pivot facets allow for faceting on expressions instead of fields
  - Complex sorting is enabled for each pivot independently
- Results are calculated at each pivot level and for each pivot value

Pivot Facet Example Head-To-Head Stats





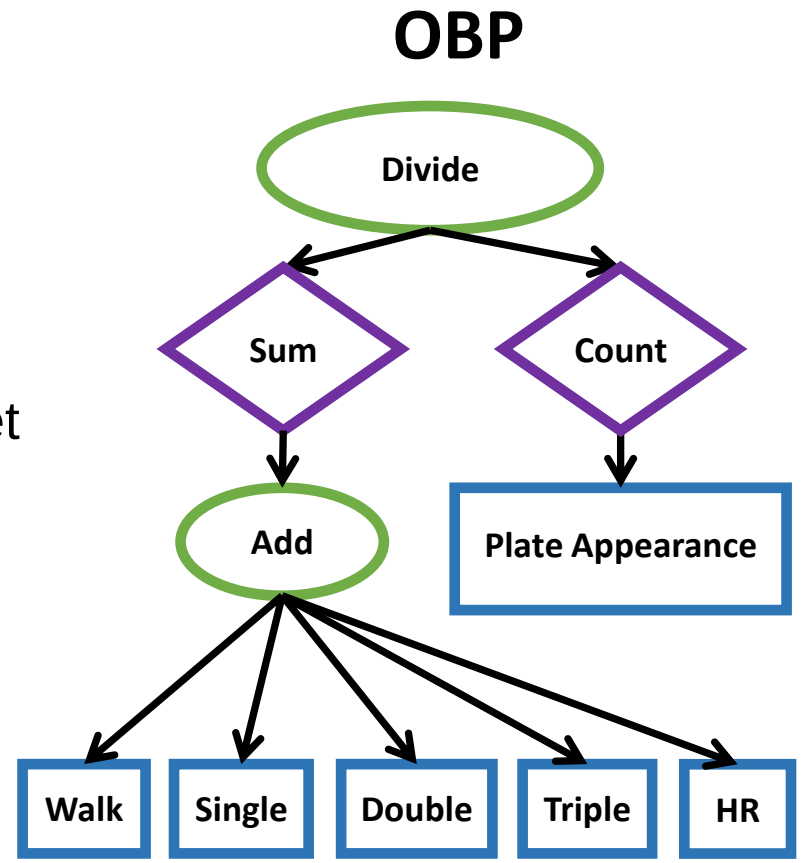
Bloomberg
Engineering

Pivot Facet Example Head-To-Head Stats

```
{
  type : "pivot",
  pivots : [
    {
      name : "Team",
      expression : "fill_missing(team,'No Team')",
      sort : {
        criteria : [{
          type : "expression",
          expression : "homerunCount",
          direction : "ascending"
        }],
        limit : 10
      }
    },
    {
      name : "Opposing Team",
      expression : "opposingTeam",
      sort : {
        criteria : [{
          type : "facetValue"
        }]
      }
    }
  ]
}
```

Mapping vs. Reducing with Facets

- Mapping functions combine values within documents
- Reduction function combine data across documents
  - Breaking up the data for each player
- Mapping functions also combine the results of reductions, per facet



Date	Player	Inning	PA	Walk	1B	2B	3B	HR	ADD(…)
17-05-01	Altuve	1	True		1				1
17-05-01	Springer	1	True	1					1
17-05-01	Correa	1	True						0
17-05-01	Altuve	4	True						0
17-05-01	Springer	4	True			1			1

	Player		Count						Sum	Divide(…)
	Altuve		2						1	.500
	Springer		2						2	1.000
	Correa		1						0	.000

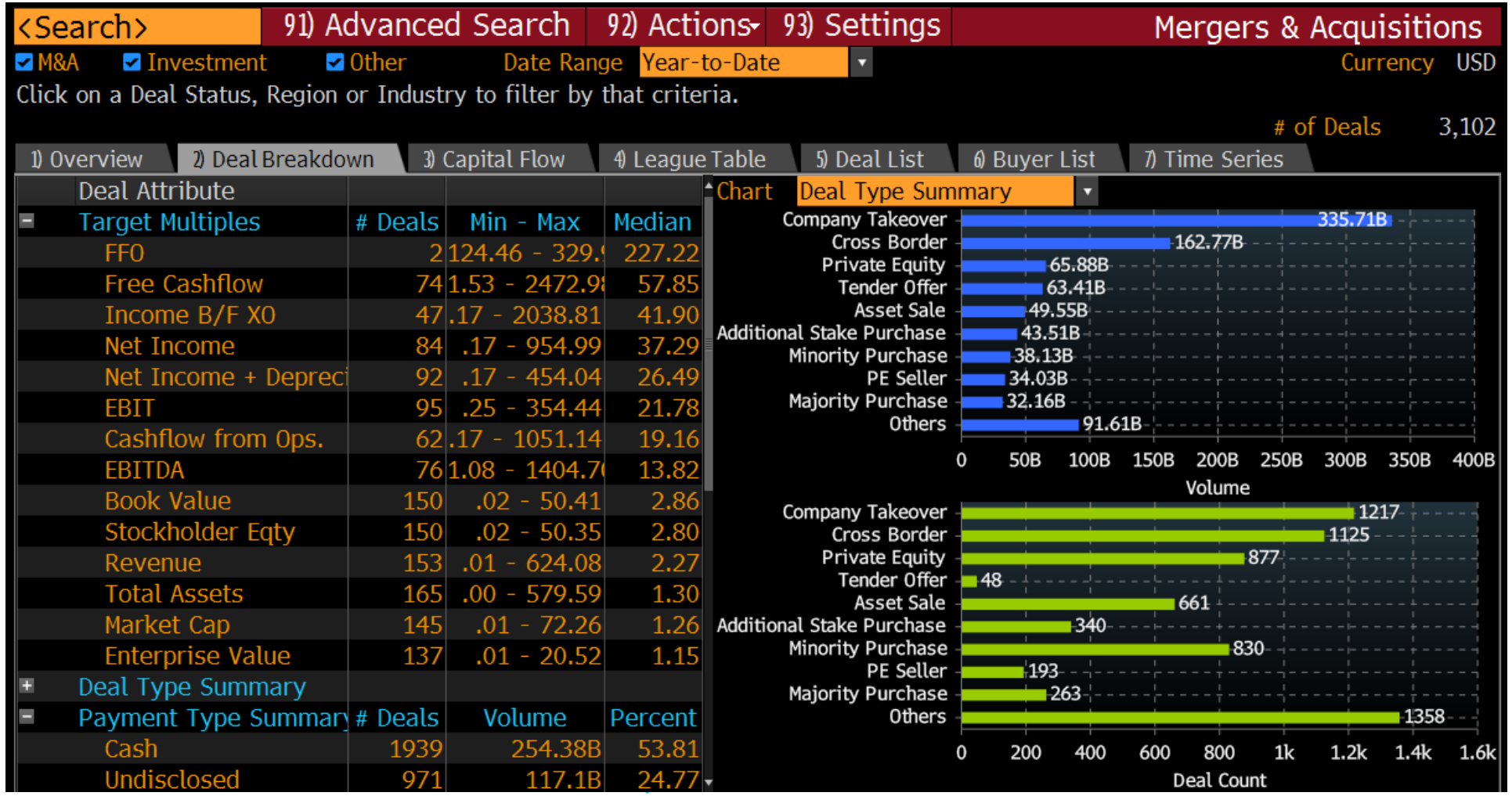
OBP

TechAtBloomberg.com

Bloomberg

Engineering

Analytics at Bloomberg

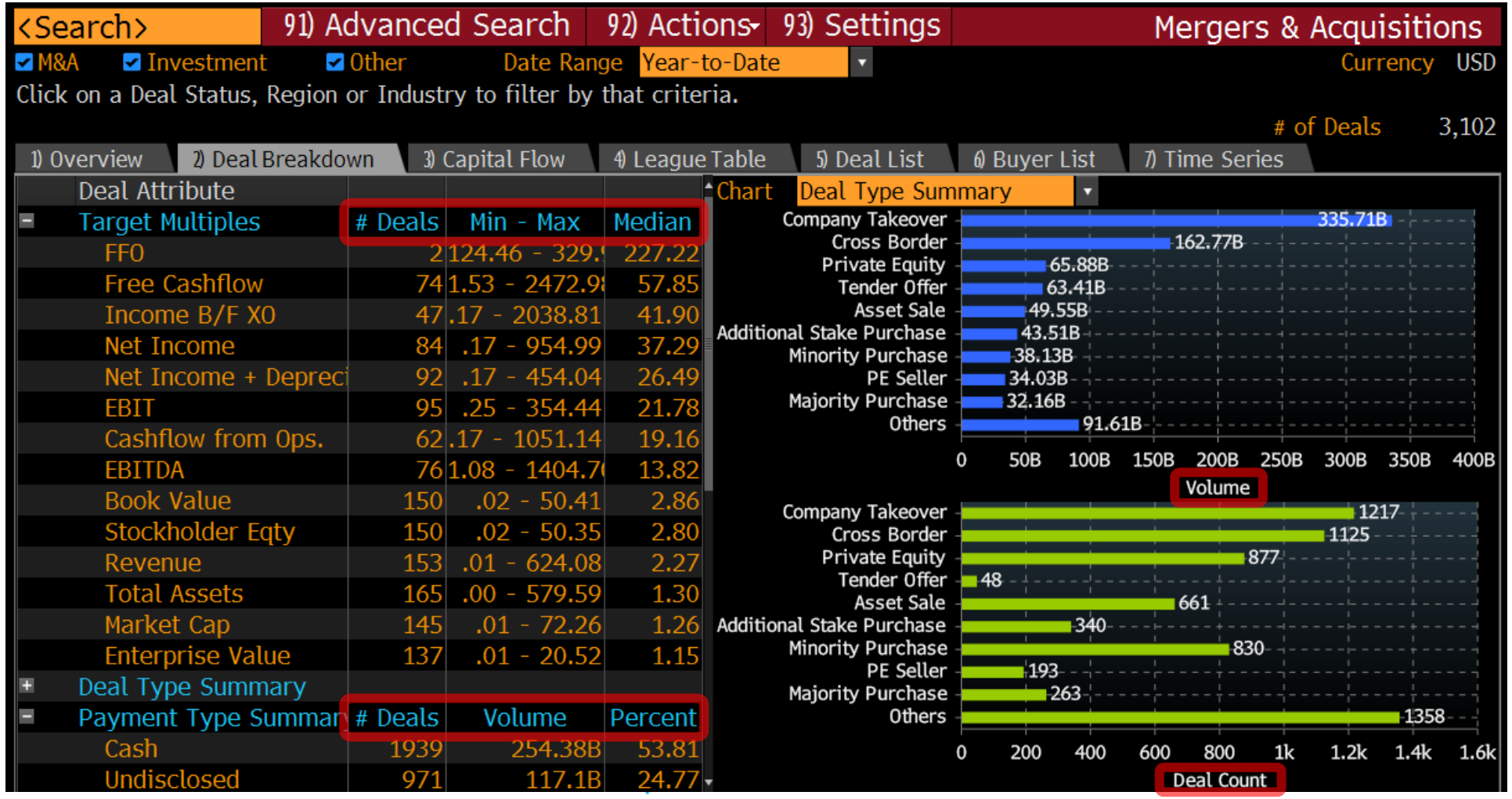


TechAtBloomberg.com

Bloomberg

Engineering

Analytics Expressions

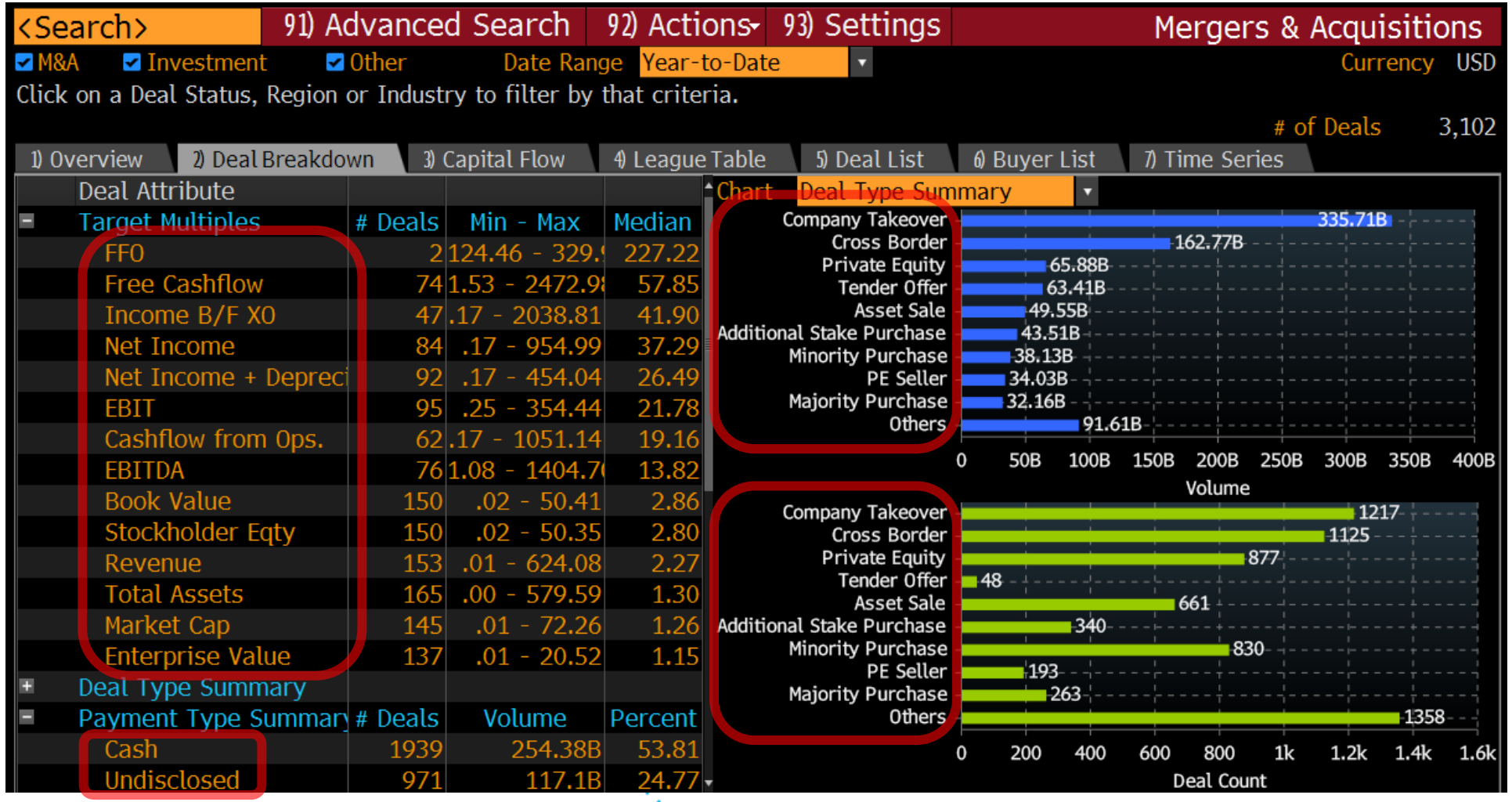


TechAtBloomberg.com

Bloomberg

Engineering

Facets



Why is it needed?

- Collections can have billions of documents in them
  - It is very costly to iterate over billions of documents to calculate analytics
  - Solr allows at most 2 billion documents per shard
- Solution
  - Spread your data across many machines
    - Each machine will only have to iterate over a subset of the data

Sounds good. What's the issue?

- Solr supports partitioned collections

Shard 1

Date	Player	Inning	HR	Map()
17-05-01	Altuve	1	1	…
17-05-01	Springer	1	0	…
17-06-02	Bregman	7	0	…
17-06-02	Reddick	7	1	…
Reductions				

Shard 2

Date	Player	Inning	HR	Map()
17-05-01	Correa	1	0	…
17-05-01	Altuve	4	0	…
17-06-02	Springer	4	1	…
Reductions				

Shard 3

Date	Player	Inning	HR	Map()
17-06-02	Gattis	7	0	…
17-06-05	Altuve	3	0	…
17-06-05	Springer	3	0	…
Reductions				

- Mapping functions aren't affected since operations are done per-document
- Reduction functions need to consolidate all data; however, this data spread across shards
- With three shards, we will end up with 3 sets of reductions

Is distribution reduction hard?

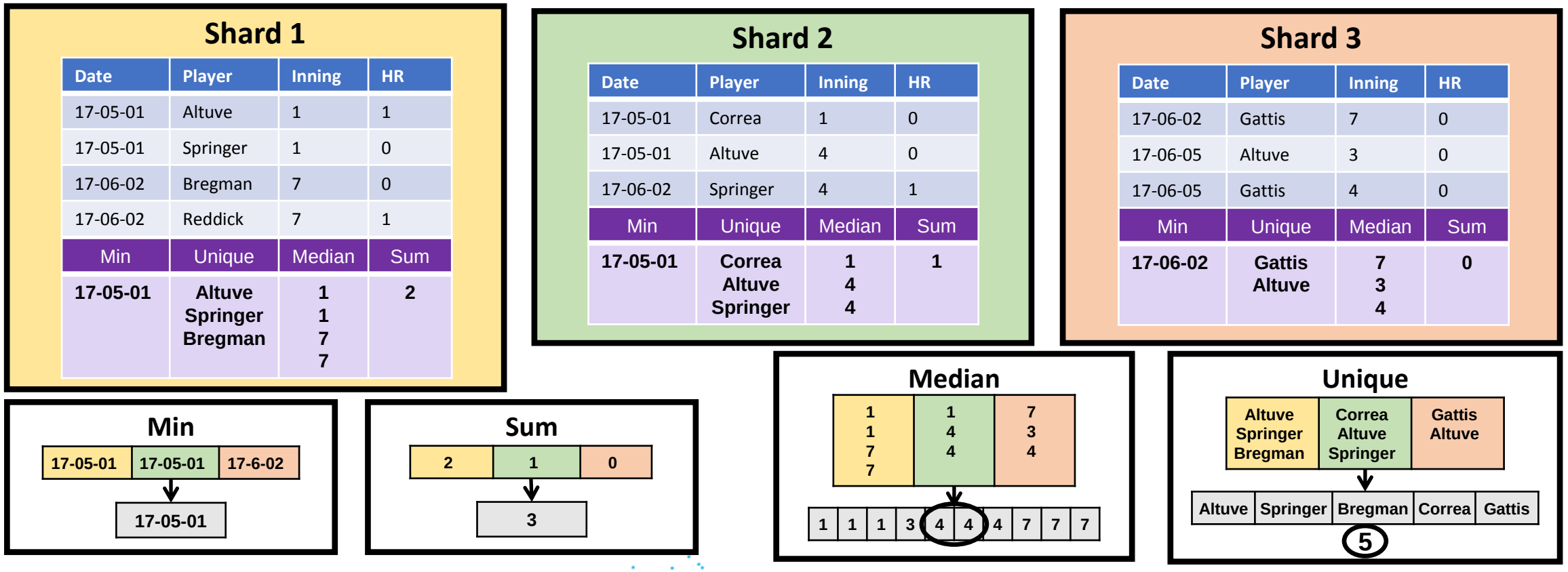
- No, not for associative reduction functions

  $f(a,b,c,d) \equiv f(f(a,b),f(c,d))$

  - Sum
  - Count
  - Min
  - Max
- Yes, for non-associative reduction functions. These require all data to be in one place
  - Percentile
  - Median
  - Unique

Solution

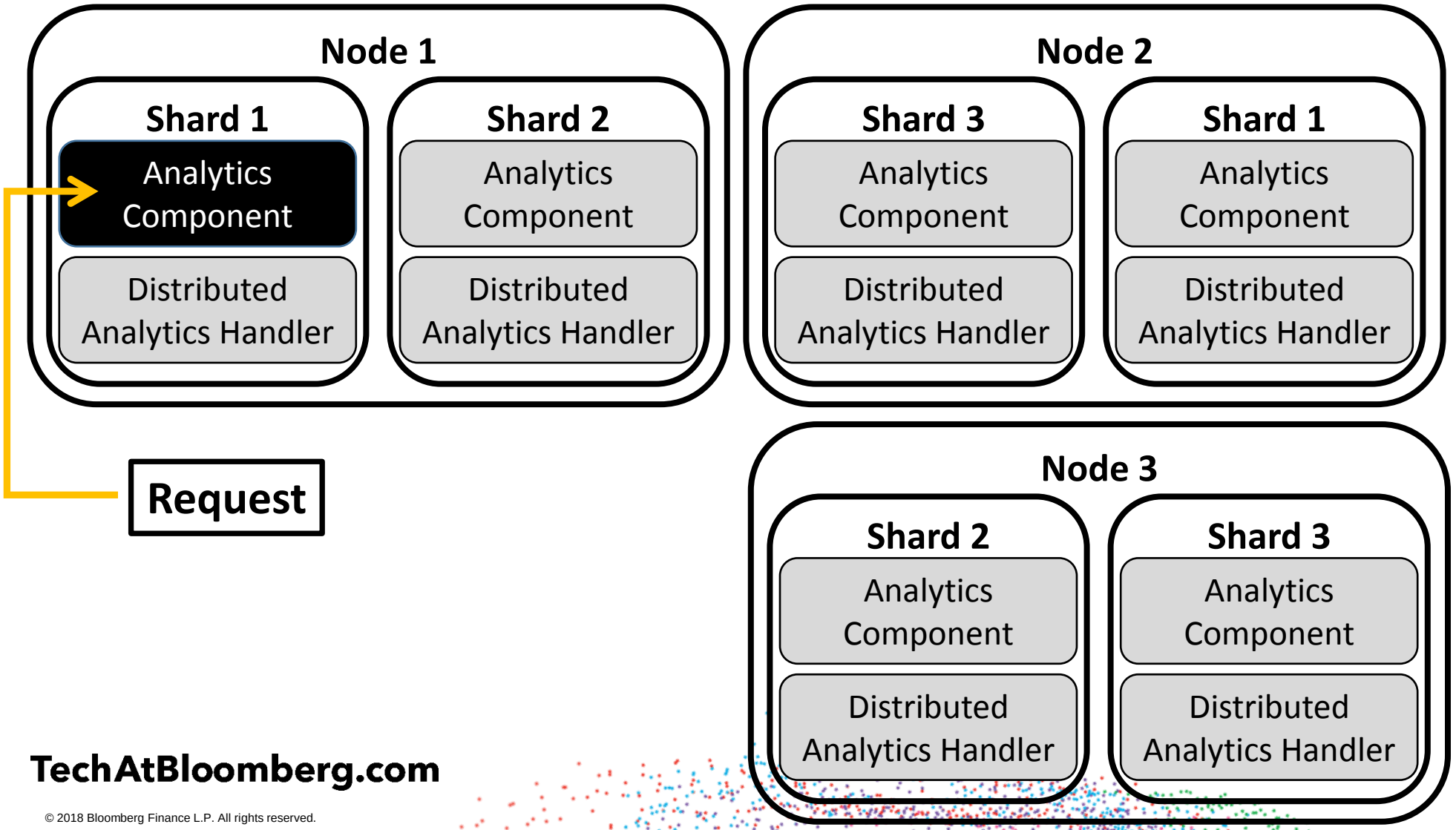
- Each reduction function requires different data to be sent from shards
- Therefore, each reduction function is in charge of exporting its shard data and merging the results



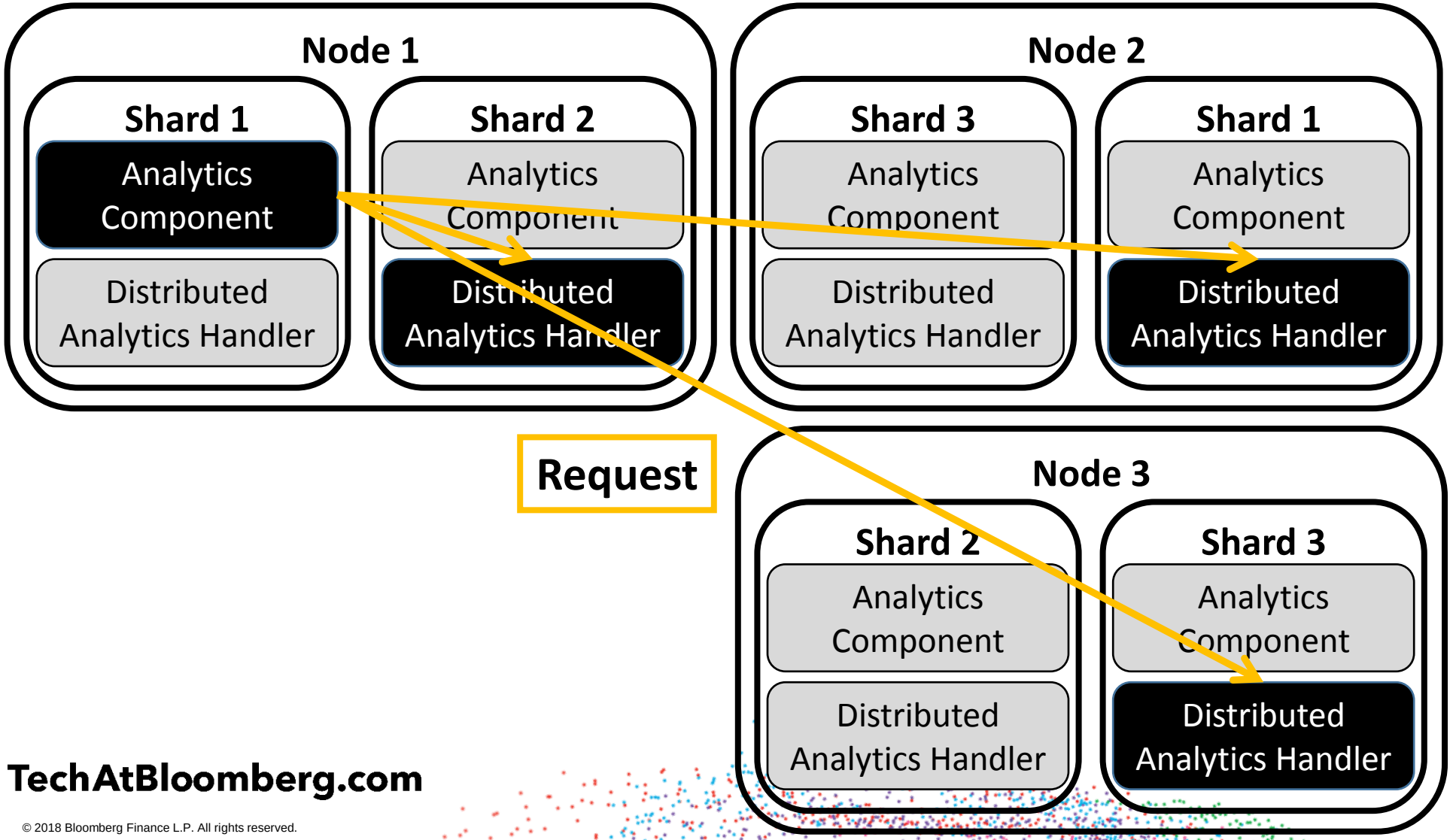
TechAtBloomberg.com

Bloomberg
Engineering

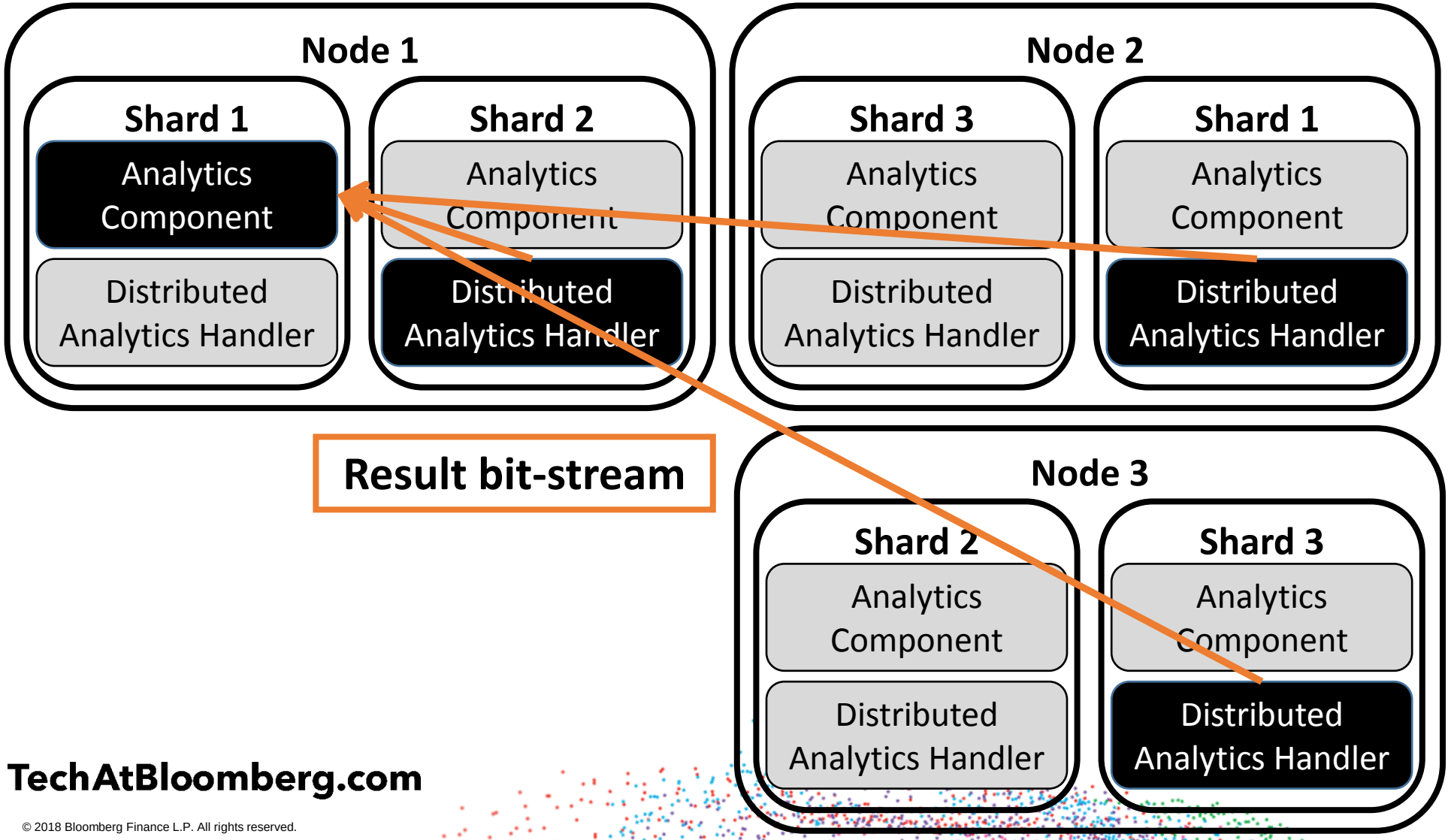
Distributed Request



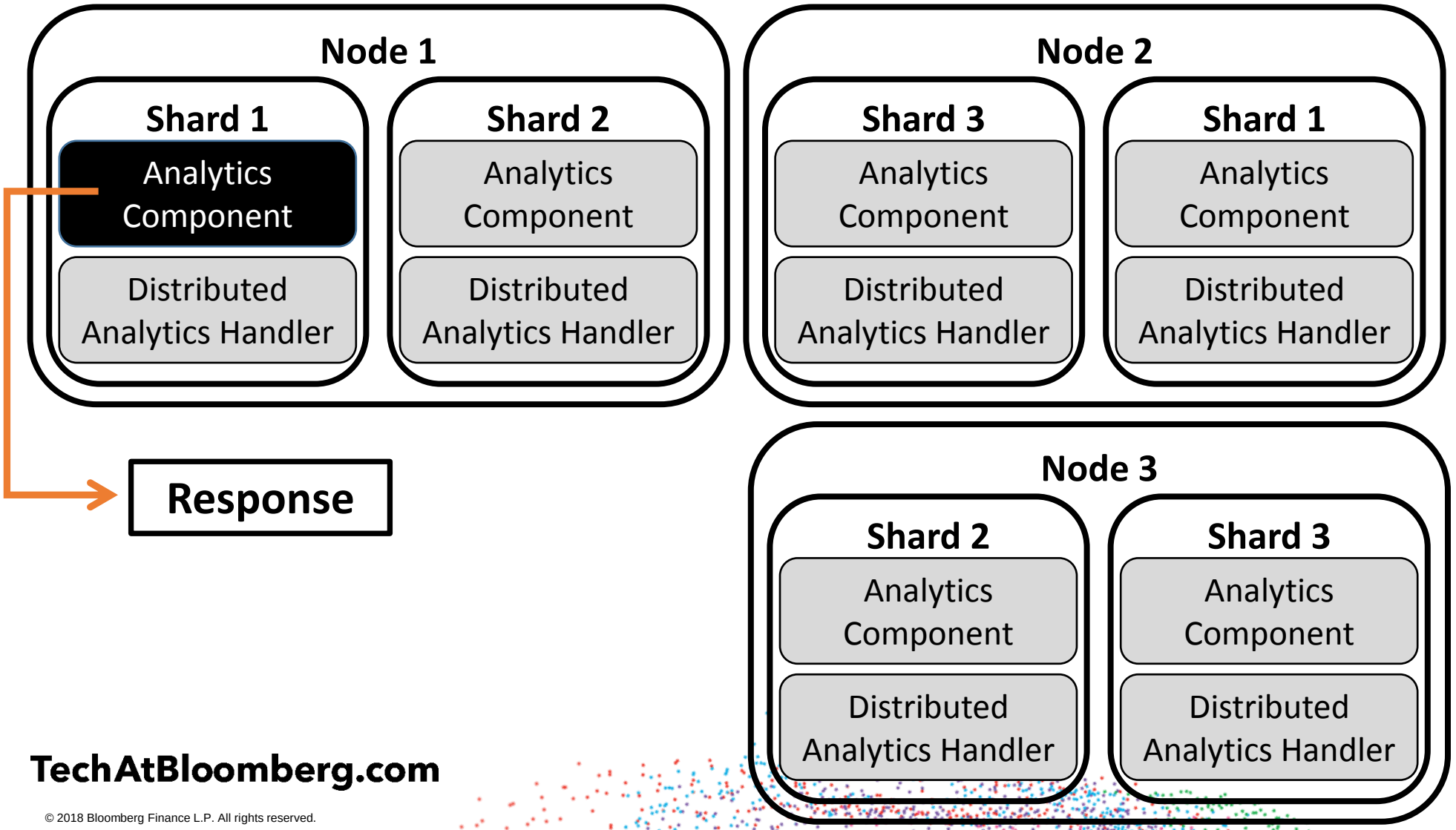
Distributed Request



Distributed Request



Distributed Request



TechAtBloomberg.com

Bloomberg
Engineering

Takeaways

- Distributed analytics lets you speed up aggregations by as many shards your data is split into
  - Associative reductions should see nearly linear scaling
- The request interface is the same for single-sharded and multi-sharded collections
  - No features/functions disabled for multi-sharded requests

Request Pipeline

Processing of a request is done in distinct phases, to allow for the maximum possible parallelization

1. Execute query, find result set to calculate analytics over
2. Read from index, calculate mapping expressions, populate reduction Data
   (for pivot and value faceted & not faceted expressions)
   - A field from a document is only read once (except for Range & Query Facets)
3. Send new queries for Range & Query Facets, each returning to **Step 1**
4. If multi-sharded, send all reduction Data back to originating shard
5. Calculate expression results from reduction Data
6. Filter and sort facet results to match request
7. Return results to user.

Overlapping Expressions

- A large analytics request may contain many functions and fields (sub-expressions) used multiple times
- Calculating these overlapping sub-expressions multiple times would be a waste of time
- The Analytics component saves time from reading from the index and performance unneeded computation
- **div(sum(add(HR, BB)), count(PA))**
- **count(PA)**
- **sum(HR)**
- **mean(add(HR, BB))**

Distributed Reduction Solution

- As shown before, the system for reducing distributed data

Shard 1

Date	Player	Inning	HR
17-05-01	Altuve	1	1
17-05-01	Springer	1	0
17-06-02	Bregman	7	0
17-06-02	Reddick	7	1

Min	Unique	Median	Sum
17-05-01	Altuve Springer Bregman	1 1 7 7	2

Shard 2

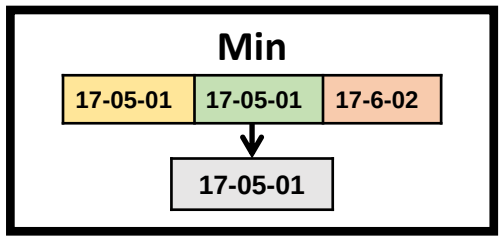
Date	Player	Inning	HR
17-05-01	Correa	1	0
17-05-01	Altuve	4	0
17-06-02	Springer	4	1

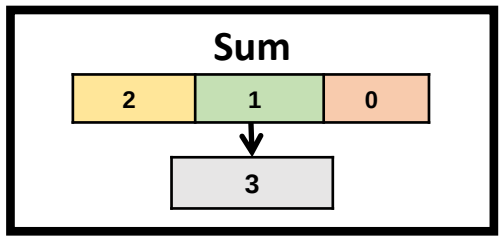
Min	Unique	Median	Sum
17-05-01	Correa Altuve Springer	1 4 4	1

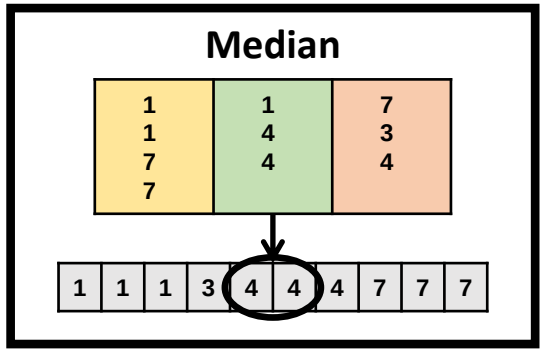
Shard 3

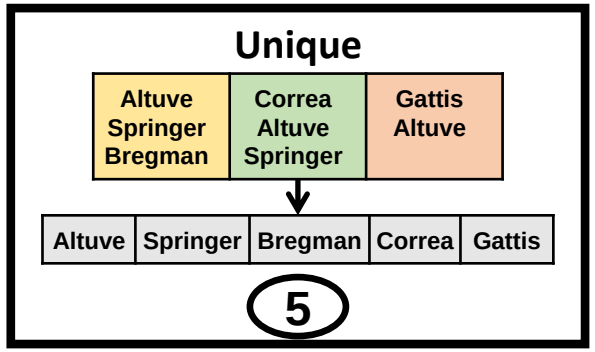
Date	Player	Inning	HR
17-06-02	Gattis	7	0
17-06-05	Altuve	3	0
17-06-05	Gattis	4	0

Min	Unique	Median	Sum
17-06-02	Gattis Altuve	7 3 4	0



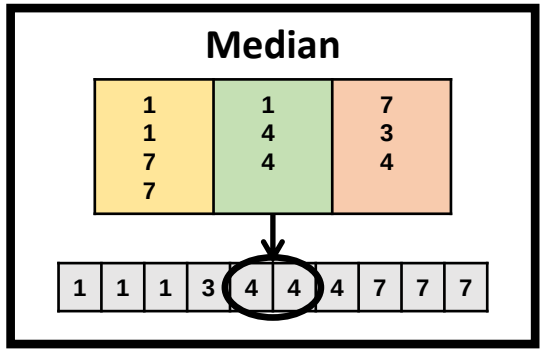


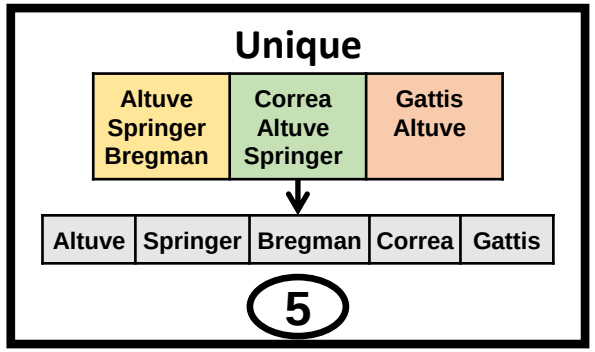




Minimizing data transfer

- For hard reductions the amount of data being sent across shards can grow linearly
  - We want to send as little data as possible across shards
  - Eliminate duplicate data
- We need a way for reduction functions to share data
- The following reduction functions would all send the same data across shards
  - Median(Inning)
  - Percentile(20, Inning)
  - Percentile(60, Inning)





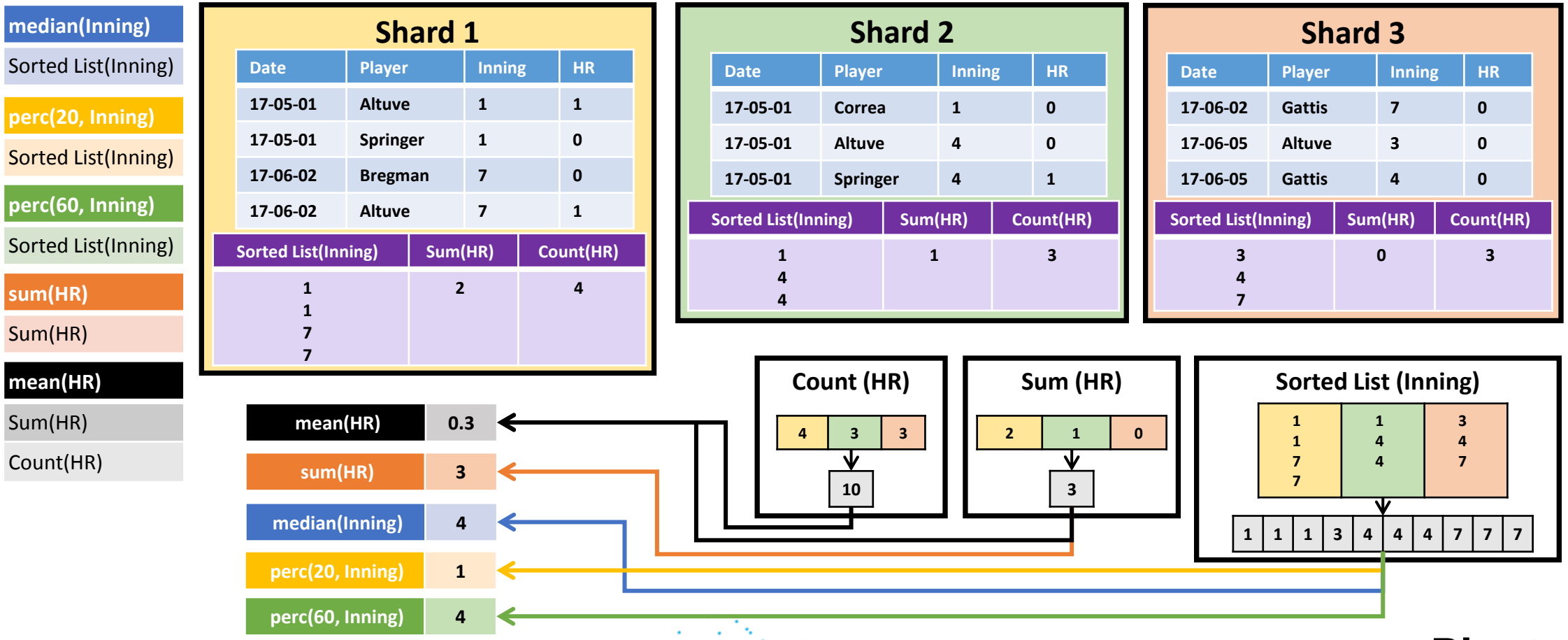
Bloomberg
Engineering

Reduction data sharing

- Reduction functions reserve specific reduction data.
  - Sorted_List(Inning)
    - Median(Inning)
    - Percentile(20,Inning)
    - Percentile(60,Inning)
  - Unique_Set(Player)
    - Unique(Player)
- Reduction data is now in charge of the shard export/merge process
  - One data transfer is made for each reduction data no matter how many reservations
  - Reduction functions use the result of the merged reduction data that they reserved
- Performance improvements for non-sharded collections

Reduction data sharing

- Median(Inning), percentile(20, Inning), percentile(60, Inning), sum(HR), mean(HR)



TechAtBloomberg.com

Bloomberg
Engineering

Other

- Adding new expressions to be returned won't necessarily add a large amount of computation time
  - Overlapping expressions reuse as many of the same parts as possible
  - Reduction functions using the same pieces of reduction data will share.
- Non associative reductions require a significant amount of memory for large result sets
  - Calculating the median requires the starting node to hold all of the values in memory
- Lower memory consumption for high cardinality facets than previous versions of the component
- All fields used in expressions must have docValues enabled

Multi-valued Expressions

- Expressions over multi-valued fields supported
- Therefore the existing mapping functions need to be modified to accept multi-valued arguments
- Consistency in the way multi-valued expressions are handled as input
- **(single) → single**
- **(single, single) → single**
- **(single…) → single**

Multi-valued Expressions

- **(single) → single**
  - Log, Negate
  - **(Multi) → (Multi)** : For each value in the input, apply the map

Date	Player	Inning	PA	Strikes	Balls	NEG(Inning)	NEG(Strikes)
17-05-01	Altuve	1	T	1 2	3 4 5	−1	−1 −2
17-05-01	Altuve	4	T			−4	
17-05-01	Altuve	7	T	3	1 2	−7	−3
17-05-02	Altuve	1	T	2 3	1	−1	−2 −3
17-05-01	Altuve	3	T	1		−3	−1

Multi-valued Expressions

- **(single, single) → single**
  - Subtract, Power, Add, Equals, Less Than, etc.
  - **(single, multi) → multi** : For each value in the second parameter apply the function to the value of the first parameter
  - **(multi, single) → multi** : For each value in the first parameter apply the function to the value of the second parameter

Date	Player	Inning	1B	Strikes	SUB(Inning,1B)	SUB(Inning,Strikes)	POW(Strikes,Inning)
17-05-01	Altuve	1	1	1 2	0	0 −1	1 2
17-05-01	Altuve	4	0		4		
17-05-01	Altuve	7	1	3	6	−3	3
17-05-02	Altuve	1	0	2 3	1	−2 −3	2 3
17-05-01	Altuve	3	1	1	2	−1	1

Multi-valued Expressions

- **(single…) → single**
  - Concat, Add, Multiply, Top, Bottom, etc.
  - **(multi) → single** : Apply the function on all of the values in the parameter
  - Order out of multi-valued fields cannot be guaranteed

Date	Player	Inning	1B	Strikes	CONCAT_SEP(“,”, Inning, 1B)	CONCAT_SEP(“,”, Strikes)
17-05-01	Altuve	1	1	1 2	1,1	1,2
17-05-01	Altuve	4	0		4,0	
17-05-01	Altuve	7	1	3	7,1	3
17-05-02	Altuve	1	0	2 3	1,0	3,2
17-05-01	Altuve	3	1	1	3,1	1

Newly supported mapping functions

- Logical
  - And, Or, …
- Comparison
  - Equal, Less Than, Greater Than, …
- Conditional
  - If(boolExpr,thenExpr,elseExpr)
  - Fill_Missing(expr,withValue)
  - Remove(expr,removeValue)
  - Filter(expr,boolExpr)

Variable functions

- Typing out expressions with similar logic multiple times is error prone
- Give users the ability to write custom functions that utilize built-in functions
- `foo(param1, param2) =` Expression using param1 and param2
  - `mean(a) = div(sum(a),count(a))`
  - `mean_inning = mean(inning)`
- Variable length parameters
  - `csv(exprs..) = concat_sep(',', exprs)`

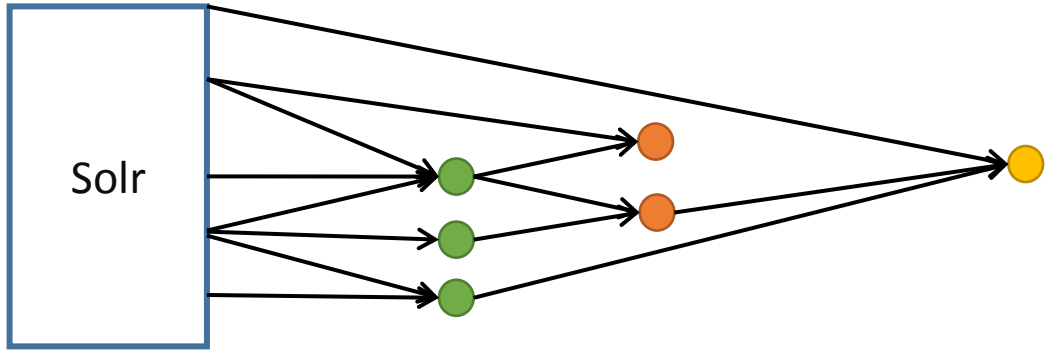
    `csv('one', 'two')` → `concat_sep(',', 'one', 'two')` → 'one,two'
- Wrapping variable length parameters – lambda functions
  - `csv(a..) = concat_sep(',', fill_missing(a, 'N/A'))`
  - `csv(a..) = concat_sep(',', a:fill_missing(_, 'N/A'))`

    `csv('one', null)` →

    `concat_sep(',', fill_missing('one', 'N/A'), fill_missing(null, 'N/A'))` → 'one,N/A'

First Order Analytics

- Solr Analytics is built to analyze first order analytics very quickly, and with as much parallelization as possible
  - First order analytics are expressions that only rely on the underlying data set
  - Second order analytics rely on the underlying data set and the results of first order expressions
  - Third order…



Uses in Bloomberg

- Solr Analytics doesn't fit all use cases
  - Bloomberg still uses Hadoop and Spark heavily
  - Several teams use Streaming Expressions and a few use JSON Facets
- Solr Analytics is used heavily within the hundreds of client teams supported by Search Infrastructure
  - Replaced many high-priced external and custom in-house solutions
- Use cases range:
  - Analyzing 100s of results to 100,000,000s of results
  - 1 shard to dozens of shards
  - Non-faceted to facets with 100,000s of values

Future Work

- Ability to add custom functions in the schema
- Ability to pivot over different types of facets
- Integration with Streaming Expressions
- Support n-order expressions natively

TechAtBloomberg.com

Bloomberg

Engineering

Conclusion

- Compute complex analytics w/o spending resources on exporting data
- Running in production at scale at Bloomberg
- Available starting with the Solr 7.0 release! (Originally included with Solr 5)
  - Documentation and important bug fixes included in 7.2
- **If you have a Solr cloud and need analytics, Solr analytics is for you**

TechAtBloomberg.com

Bloomberg

Engineering

Questions?

Reference - https://lucene.apache.org/solr/guide/analytics

Join us at Bloomberg - bit.ly/JavaSearch_Bloomberg

Current Work

- SOLR-12045 – Moving the Analytics Component from contrib to core

History

- SOLR-10123 – Introduction of Analytics 2.0
- SOLR-11146 – Important bug fixes
- SOLR-5302 – Original (Solr 5) Analytics Component

TechAtBloomberg.com

Bloomberg

Engineering

Where does Solr Analytics fit in?

- There are many established analytics engines available, such as Spark and Hadoop
- Solutions have been proposed to combine Solr with these projects in order to leverage search capability with analytics
- Using external analytics engines requires exporting the needed data set from Solr
  - The benefits of using external analytics engines come from analyzing large amounts of data, therefore most problems you need Spark to solve will require long exporting tasks
- Spark and Hadoop have many tools for data scientists to play with data
  - Solr Analytics isn’t a complete replacement for these systems

When does Solr Analytics make sense?

- Using an internal analytics engine, such as the Analytics component allows you to perform complex data introspection without spending the time of exporting data from Solr
  - Solr Analytics was built using map-reduce principles
  - Using an internal engine reduces the complexity of the data pipeline
- Solr is as live as the data ingested into it, applications that want to take advantage will have a hard time exporting
  - Bloomberg users demand analytics over live data
- Hadoop and Spark have such rich ecosystems due to the community involvement
  - The Analytics Component was written to be very modular
  - Improvements and new features/functions are always welcome