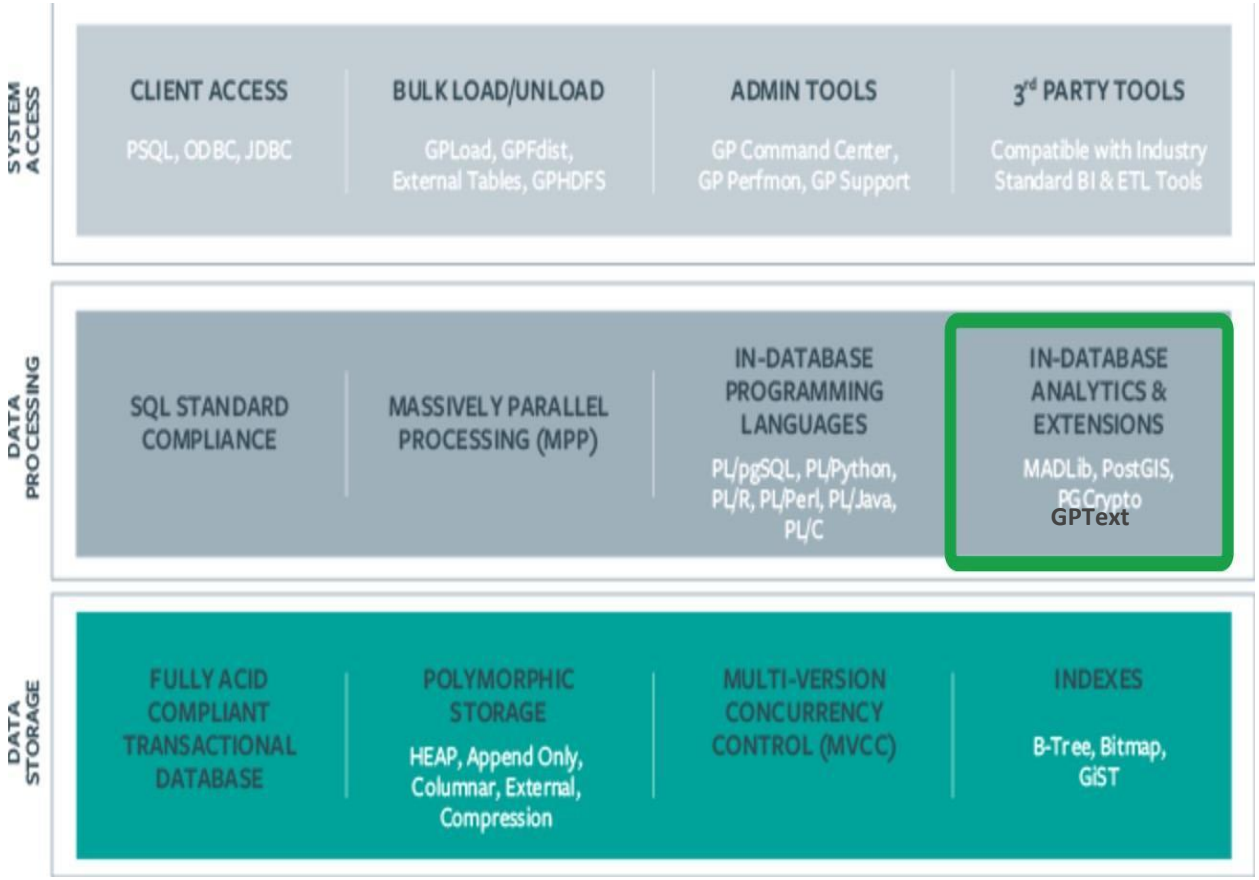
Yu Yang (myang@pivotal.io)

Greenplum Database（GPDB）



GPText Overview

- Large-scale text analytics with MPP
- Supports semi-structured and structured data
- SQL interface
- Text Machine Learning support
- Customized in depth
- Easy to maintain
- Highly scalable
- ...

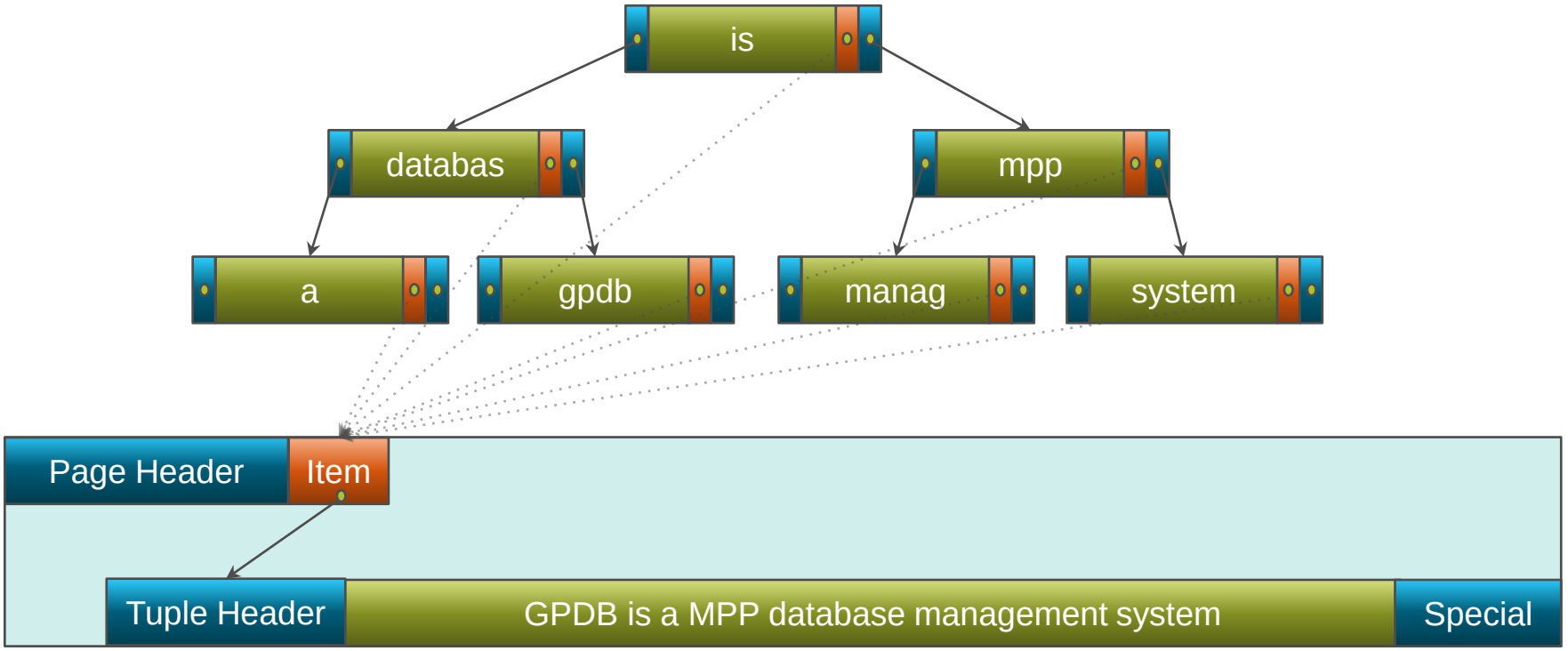
Full-Text Search Principles

Introduction

Why cannot we simply use an index for full-text search?

- Consider the phrase ‘GPDB is a MPP database management system’
- Tokenize - split into array of words: [‘GPDB’, ‘is’, ‘a’, ‘MPP’, ‘database’, ‘management’, ‘system’]::text[]
- Move to lowercase: [‘gpdb’, ‘is’, ‘a’, ‘mpp’, ‘database’, ‘management’, ‘system’]::text[]
- Stem using Porter stemming: [‘gpdb’, ‘is’, ‘a’, ‘mpp’, ‘databas’, ‘manag’, ‘system’]::text[]
- Build binary index

Introduction

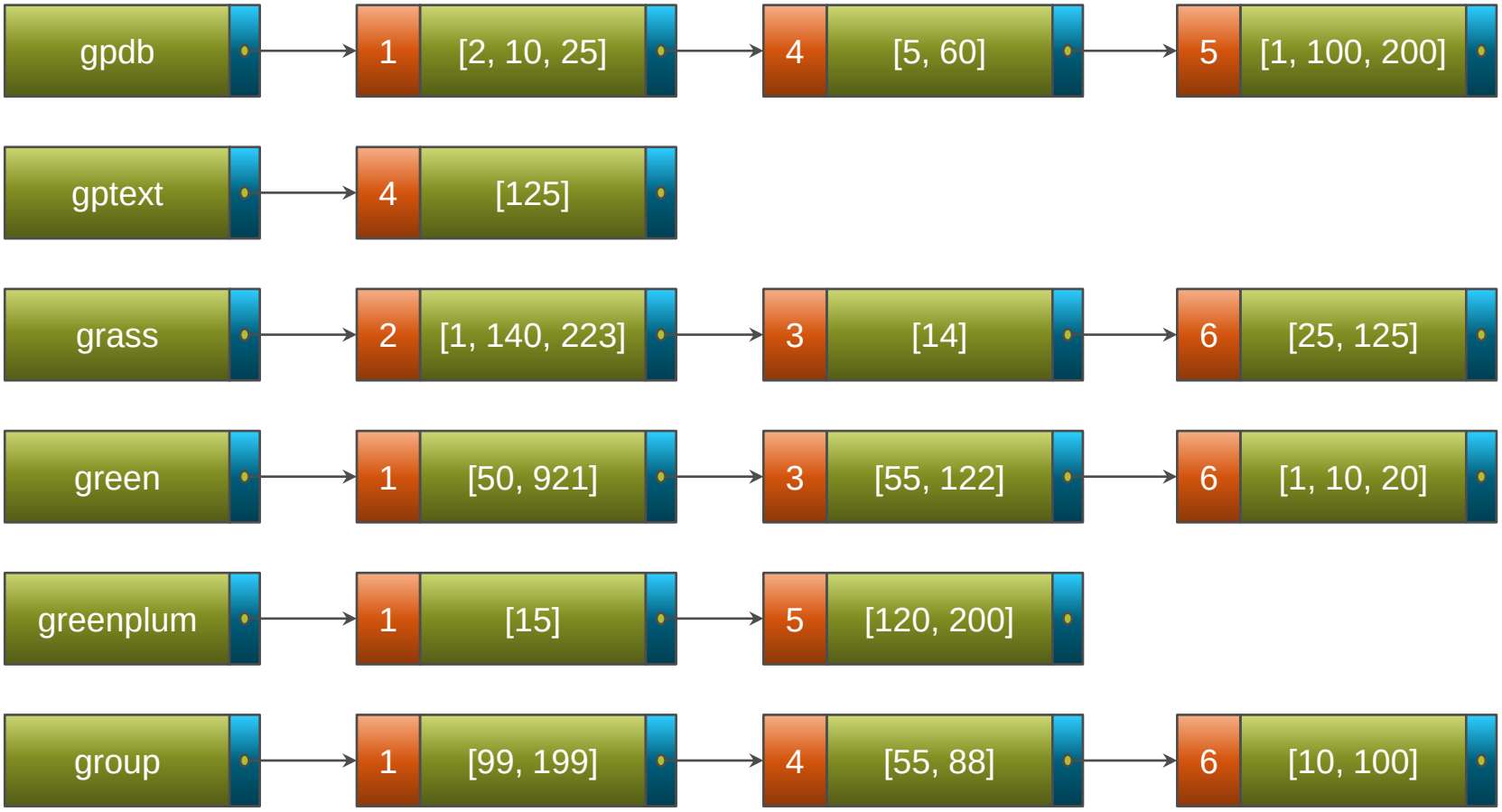


- Same term for different documents would be stored many times – no way in B-tree or B+-tree to store one key and many links to data
- How to estimate which word is more important and which one is not?
- How to handle complex queries that require exact phrase matching?
- How to rank query results?

There are 4 main tasks

- Preprocess data before putting it to index
- Have an index structure optimized for full-text searches
- Process incoming queries quickly
- Rank the results based on how well the query match the document

Inverted Index



Example

$$\mathrm{tf}_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}}$$

$$\mathrm{idf}_i = \log \frac{|D|}{|\{j : t_i \in d_j\}|}$$

	Text #			
	1	2	3	4
is	1	1	1	1
a	1	1		
for			1	
in		1		
of			1	
with			1	1
Greenplum	1			2
MPP	1			
database	1	1	1	1
management	1			
system	1			
MADlib		1		
distributed	1		1	
library		1		
analytics		1		
GPText			1	
Solr			1	
functions			1	
interface			1	
Apache			1	1
HD				1
Hadoop				1
tightly				1
integrated				1

→

	TF*IDF			
	1	2	3	4
is	0.00	0.00	0.00	0.00
a	0.04	0.04		
for			0.05	
in		0.09		
of			0.05	
with			0.03	0.03
Greenplum	0.04			0.06
MPP	0.08			
database	0.00	0.00	0.00	0.00
management	0.08			
system	0.08			
MADlib		0.09		
distributed	0.04		0.03	
library		0.09		
analytics		0.09		
GPText			0.05	
Solr			0.05	
functions			0.05	
interface			0.05	
Apache			0.03	0.03
HD				0.06
Hadoop				0.06
tightly				0.06
integrated				0.06

Query “distributed database Greenplum”

	Inverted Index			
	1	2	3	4
is	0.00	0.00	0.00	0.00
a	0.04	0.04		
for			0.05	
in		0.09		
of			0.05	
with			0.03	0.03
Greenplum	0.04			0.06
MPP	0.08			
database	0.00	0.00	0.00	0.00
management	0.08			
system	0.08			
MADlib		0.09		
distributed	0.04		0.03	
library		0.09		
analytics		0.09		
GPText			0.05	
Solr			0.05	
functions			0.05	
interface			0.05	
Apache			0.03	0.03
HD				0.06
Hadoop				0.06
tightly				0.06
integrated				0.06

*

	Query
is	
a	
for	
in	
of	
with	
Greenplum	1.00
MPP	
database	1.00
management	
system	
MADlib	
distributed	1.00
library	
analytics	
GPText	
Solr	
functions	
interface	
Apache	
HD	
Hadoop	
tightly	
integrated	

=

	Documents			
	1	2	3	4
is				
a				
for				
in				
of				
with				
Greenplum	0.04			0.06
MPP				
database	0.00	0.00	0.00	0.00
management				
system				
MADlib				
distributed	0.04		0.03	
library				
analytics				
GPText				
Solr				
functions				
interface				
Apache				
HD				
Hadoop				
tightly				
integrated				

→

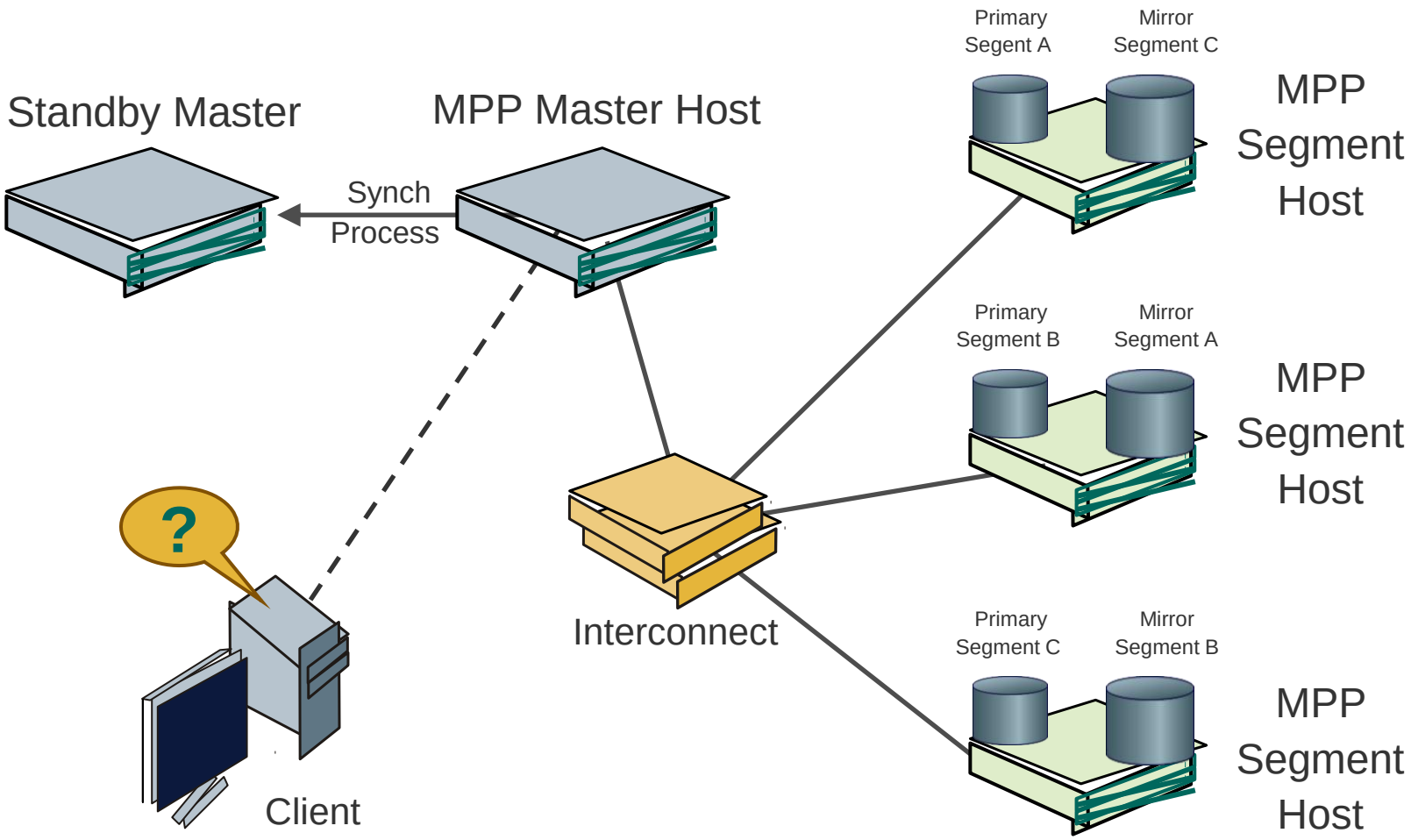
Text #	Weigth
1	0.08
2	0.00
3	0.03
4	0.06

↓

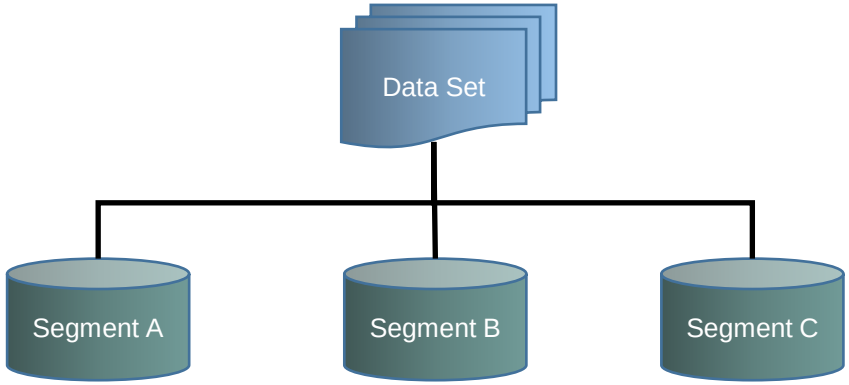
Ranked Result
1
4
3
2

Full-text search in GPDB

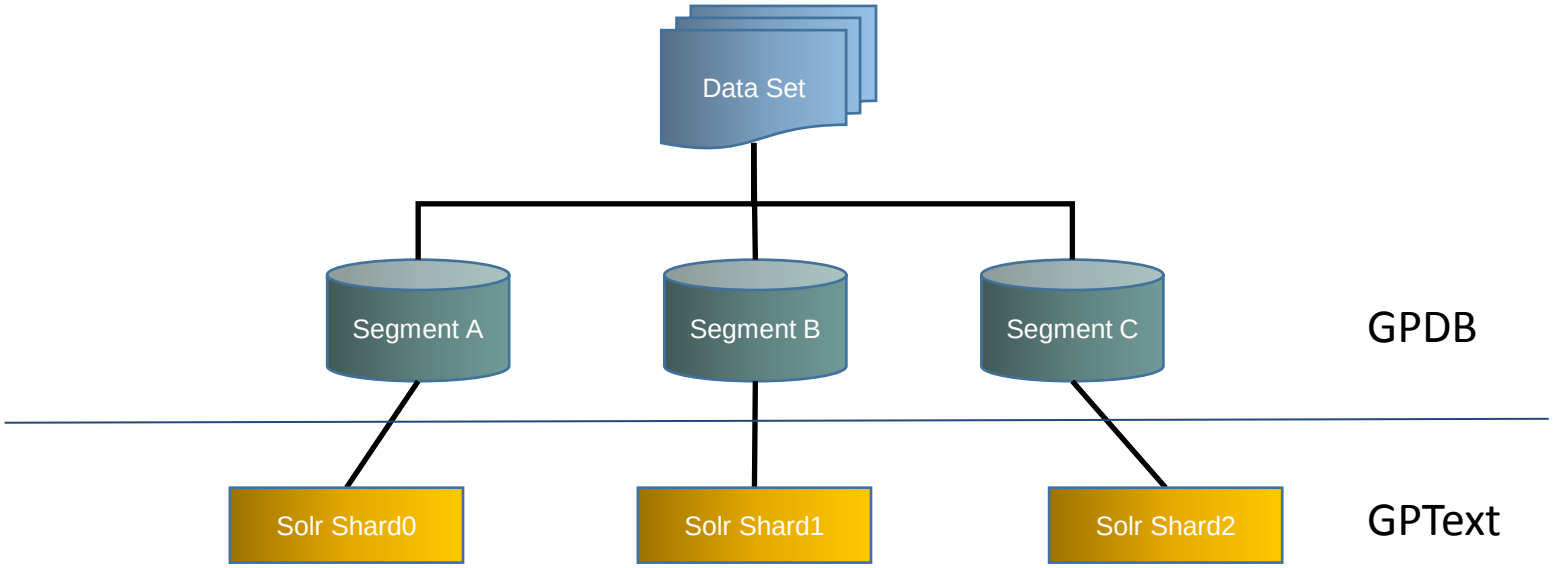
GPDB Architecture Review



Full-text search in GPDB



Full-text search in GPDB



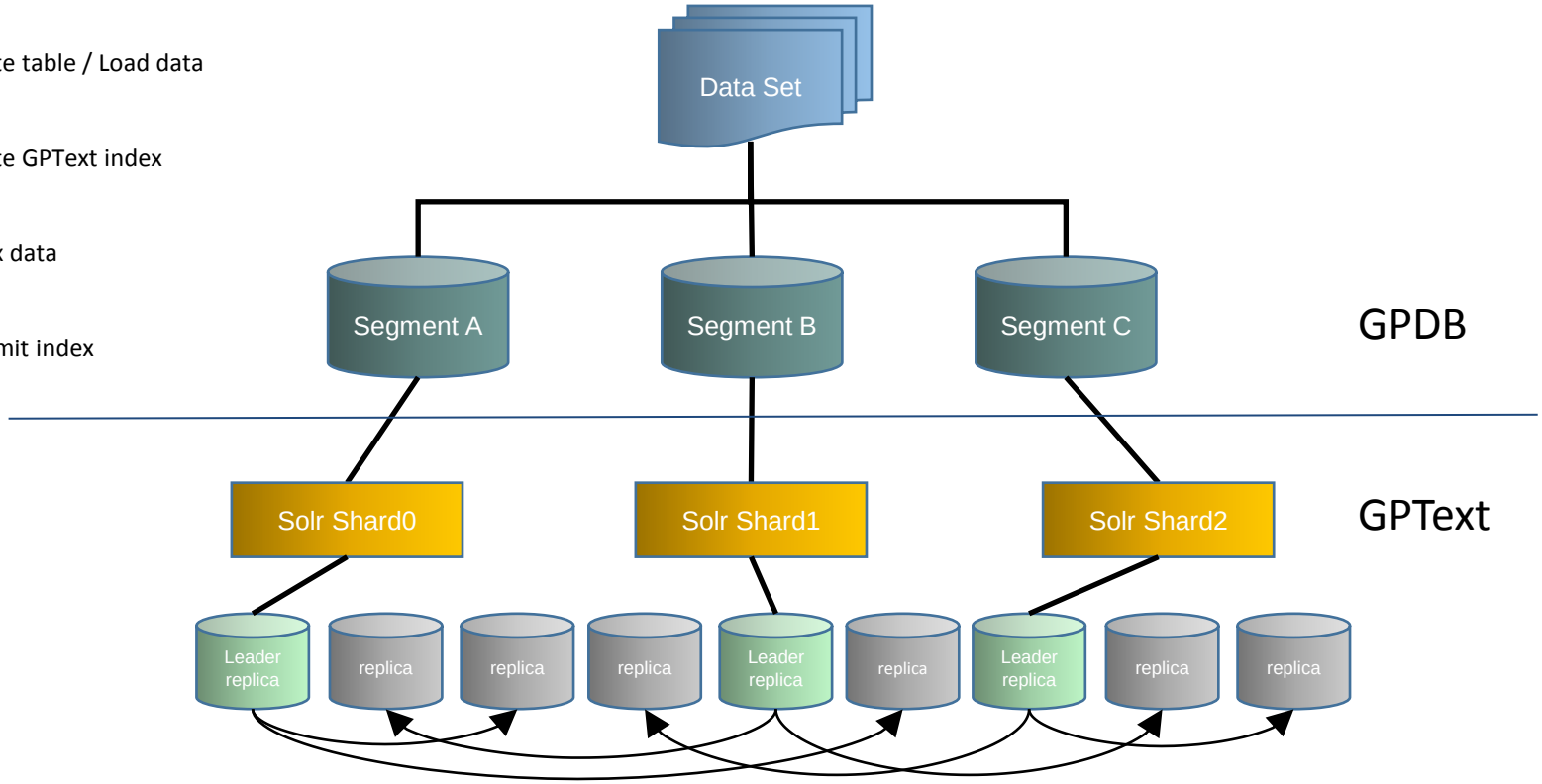
How GPText works

① Create table / Load data

② Create GPText index

③ Index data

④ Commit index



Key to MPP - Table Function

Table Value Expressions:

```
TABLE( SELECT * FROM table SCATTER BY id )
```

```
SELECT * FROM gptext.index(
        TABLE( SELECT * FROM <table> SCATTER BY id ),
        <index_name>
);
```

- Used as the input to an Enhanced Table Function (ETF)
- Creates a pseudo-table, in parallel, that allows users to pipeline data processing for the purpose of improving query performance
- Resulting data can be ordered or distributed with ‘ORDER BY’ and ‘SCATTER BY’ SQL clauses

High Availability with SolrCloud

Main Functions

- Index functions
  - create index, index, drop index, commit index, etc.
- Search functions
  - search, faceted search, etc.
- Terms operations
  - enable terms for index, document vector creation, etc.
- General functions
  - Index statistics, gptext status, gptext version, etc.

GPText vs. Solr

- Distributed Solr - Highly scalable
  - Leverages GPDB MPP architecture to distribute nodes and shards across multiple hosts.
- SQL Interface
  - All Solr calls and functionality are exposed through SQL UDFs for a consistent interface through psql
- Ease of Configurability
  - GUC Configuration management
  - gptext-config

GPText vs. Solr (cont.)

- International/Social Media Analyzer Chains
  - Additional Analyzer Chains with custom Tokenizers/Filters to be able to accurately parse out International Text (including CJK characters) and Social Media (hashtags, emoticons, links, etc.)
- Unified Query Parser
  - Combines complex query parsers into a single unified interface
- Enables termvector support for text analytics with MADLib
  - LDA, K-Means Clustering, SVM

GPText tour

UDF interfaces

```
SELECT * FROM gptext.index_status();
SELECT * FROM gptext.create_index();
SELECT * FROM gptext.drop_index();
SELECT * FROM gptext.index(Table(), ...);
SELECT * FROM gptext.commit_index();
SELECT * FROM gptext.delete();
SELECT * FROM gptext.search(TABLE(), ...);
SELECT * FROM gptext.search_count(Table(), ...);
SELECT * FROM gptext.faceted_field_search();
SELECT * FROM gptext.terms();
...
```

Flexible search functions

Top 10 results:

```
SELECT t.id t.title, q.score FROM
         mytable t,
         SELECT * FROM gptext.search(
                  TABLE( SELECT 1 SCATTER BY 1),     // Table functions
                  Index_name,                          // Index name to query
                  “cat AND dog”,                     // Query clause
                  NULL,                              // Filter query
                  ‘rows=10’                          // Other options
         ) q
         WHERE t.id = q.id
         ORDER BY q.score DESC LIMIT 10
);
```

- Uses <options> search function parameter
- Returns top 10 results sorted by Solr relevancy score

Flexible search functions

Faceted Query Search Example:

```
SELECT * FROM gptext.faceted_query_search(
  Index_name,                                                          //Index name
  'cat OR dog',                                                        //Query clause
  'country:USA',                                                       //Filter query
  '{price:[* TO 200], price:[201 TO 250], price:[251 TO 300], price:[301 TO 400]}',//Facet field
  -1,                                                                  //Facet limit
  1                                                                    //Minimum match
);
```

- Aggregates or performs a 'groups by' on the results
- Provides a count of matching documents for each distinct 'facet_field' in the result
- facet_limit: maximum number of facets to return
- minimum: only return values with more than 'minimum' matching documents

Text Analytics with MADlib

KMeans example:

```
SELECT * FROM madlib.kmeans_plusplus(
        ‘<tfidf_table>’,
        ‘<tfidf_column’>,
        ‘<document_id’>,
        … MADLib Kmeans paramters …
);
```

- Executes KMeans clustering algorithm from MADlib
- Executes in parallel inside the database
- See MADlib documentation for complete parameter list
- Many other MADlib functions follow a similar model

GPText utilities tools

- gptext-start/stop： start/stop GPText
- gptext-state： check GPText status
- zkManager： zookeeper manager
- gptext-recover： recover GPText instances
- gptext-replica： index’s replica operation
- gptext-expand： GPText expand
- gptext-config： GPText configuration operation
- gptext-backup: Index backup
- gptext-restore Index restore
- ...

GPText GUCs - The important ones!

- replication_factor - The number of replicas per shard for a newly created index.
- failover_factor - Minimum ratio of Solr nodes that must be up in order to create a new index.
- extension_factor - Maximum number of replicas that can be added for an index per GPText node after the index is created.
- search_batch_size - Batch size for search requests. Performance vs. Resource tradeoff
- terms_batch_size - Batch size for terms operations.
- ……

Roadmap

Short-Term

Improved Memory Utilization and Monitoring

Partition table Support

Mid-Term

Leader/Replica Rebalancing

GPText support

Improved Indexing Pipeline + Analytics

Long-Term

Common Document Formats Support (PDF, Word, etc.)

Improved NLP (OpenNLP)

Thanks!

Online document:
http://gptext.docs.pivotal.io/

Download **GPText 2.0:**
https://network.pivotal.io/

We are hiring!
GPDB / HAWQ Engineer
Please contact:
pivotalrnd_china_jobs@pivotal.io