



# FAST & SCALABLE EMAIL SYSTEMS WITH APACHE SOLR

APACHE:  
**BIG\_DATA**  
EUROPE

Arnon Yogev  
IBM Research

# Background

- **IBM Verse** is a cloud based business email system

The screenshot displays the IBM Verse email interface. At the top, there is a navigation bar with options like 'Greenwell', 'Home', 'Mail', 'Calendar', 'People', 'Communities', and 'Apps'. Below this is a header area with a 'Compose' button and a search icon. The main area is divided into two columns: a list of emails on the left and a detailed view of the selected email on the right.

**Email List (Left Column):**

- Lukas Geiger** (Feb 9): WFH tomorrow. Hi team- just a reminder I'll be out of office, but will be online after 1...
- Ron Segal, Lukas Geiger** (Feb 9): **Team Dinner Plans**. Hi I would suggest NoRTH since it's close to work and can accomm...
- Heather Reeds** (Feb 9): Invitation: Renovations client visit and presentation (Tue 02/10/2015 08:00AM, ...)
- Paul Clemmons, Lukas Geiger** (Feb 9): **End of year client account reviews**. Hi Lukas, Here is a screen shot of my initial report. I will be able to h...
- Heather Reeds** (Feb 9): **Declined: scrum**
- Gail Chao** (Feb 4): Re: New Whiteboards with logo. Ron - check it out. Pretty cool. Thanks! Gail Sent from IBM Verse G...
- Heather Reeds** (Jan 22): Go to market launch plans. Attached is the budget that I need for the things that we are doing a...

**Email Detail View (Right Column):**

**Team Dinner Plans**  
Contributors: **Lukas Geiger, Ron Segal** 1 Unread

Re: Team Dinner Plans Mon, Feb 9  
Ron Segal to me (cc), Lukas Geiger, Heather Reeds, Nancy Smith [Show more](#)

Hi I would suggest NoRTH since it's close to work and can accommodate a large group.

Have a good day,  
Ron  
Sent from IBM Verse

Ron Segal  
Business Unit Executive  
85 East Pratt St Baltimore, MD 21202  
1-410-555-0044

At the bottom, there is a status bar showing 'Now: EOW close-out scrum | 12 - 2:30 pm | 555-123-1212 | Click for password' and a calendar view for 'FRI 13' with a time slot from 12pm to 11pm.

# Background – cont.

- Verse backend is based on **Apache Solr**
- Almost every user interaction with the email system is translated into Solr queries:
  - Refresh inbox
  - Search by keyword / person
  - Filter emails with attachments
  - Compose or reply to an email
  - Delete an email
  - ...

# Mission

- Support IBM Verse's future growth in terms of stability, scalability and cost-effectiveness, by optimizing its Solr infrastructure and usage



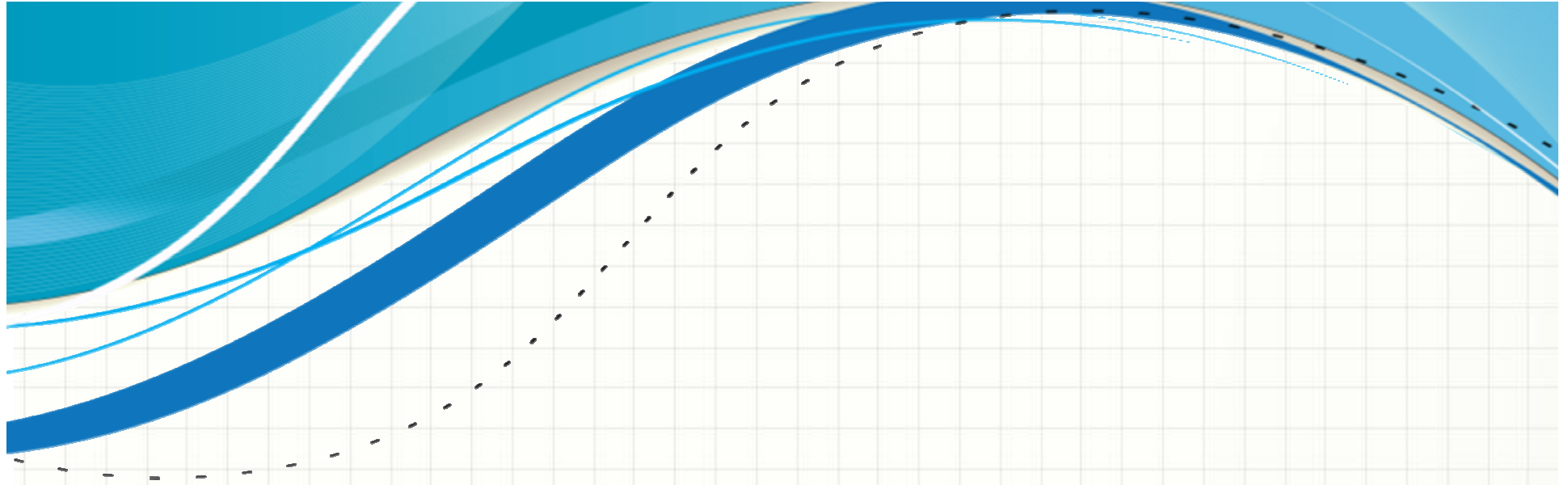
Improved  
Performance

Increased  
Server Capacity

Reduced Cost  
per User

# Premise

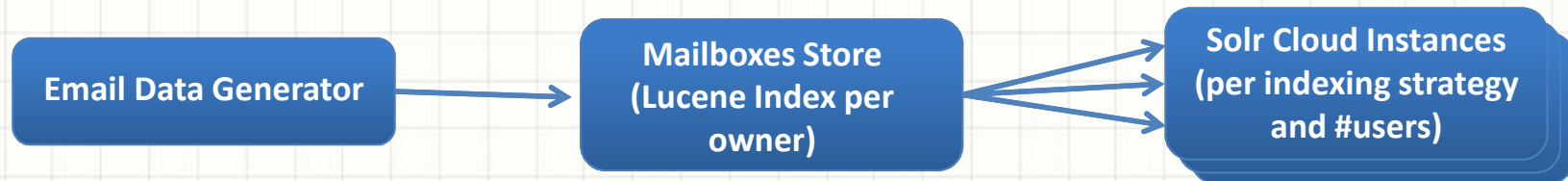
- Email search  $\neq$  web search
  - Mailboxes are private
  - Main interactions are performed on latest emails
  - Results are typically sorted by date



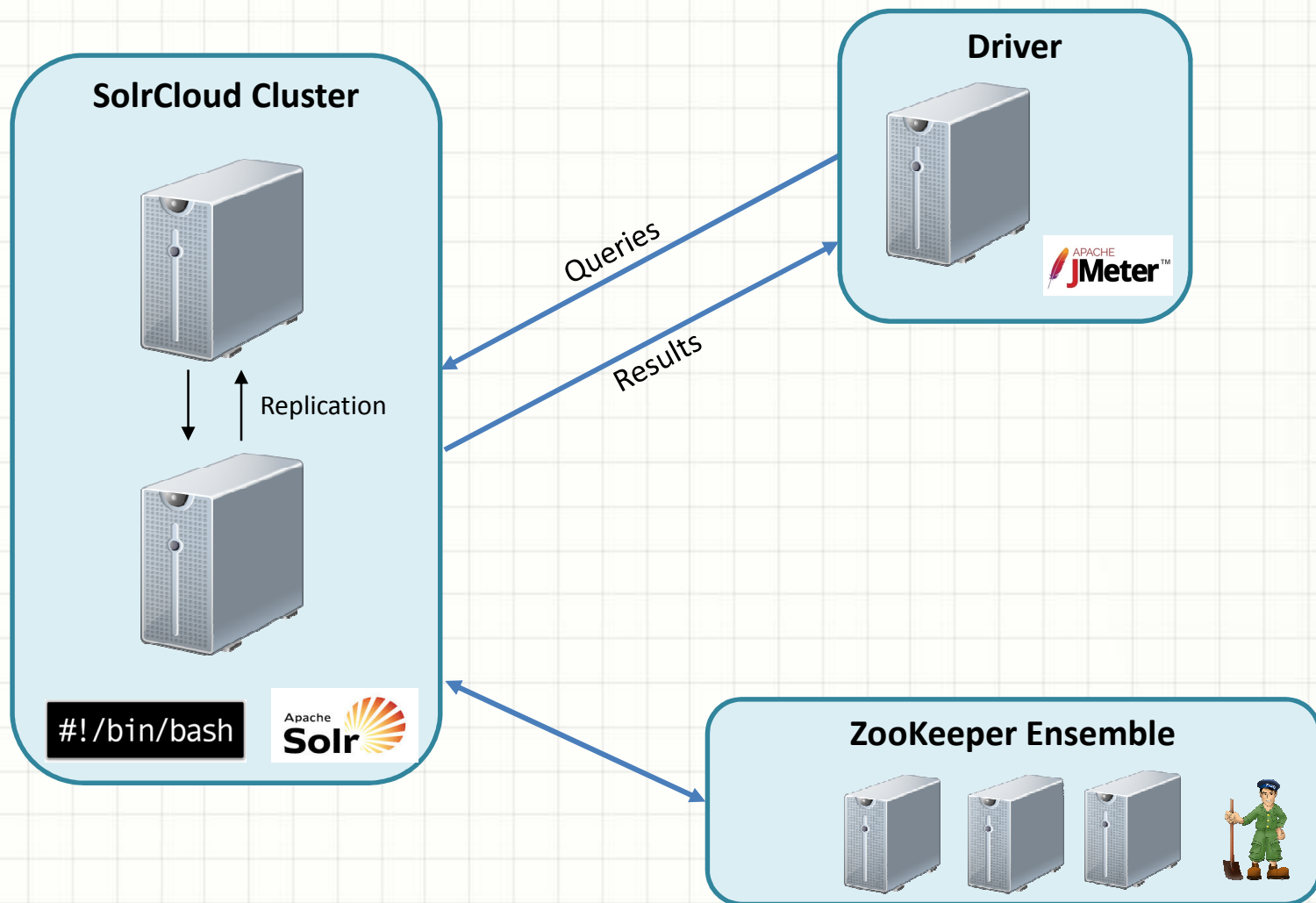
# **SETUP & ENVIRONMENT**

# Benchmark Framework

- Email Data Generator
  - Generate synthetic email data that simulates an email corpus characteristics and distributions
- Indexer Module
  - Index data into Solr using different indexing strategies
- Apache JMeter
  - Execute test scenarios based on real life usage statistics



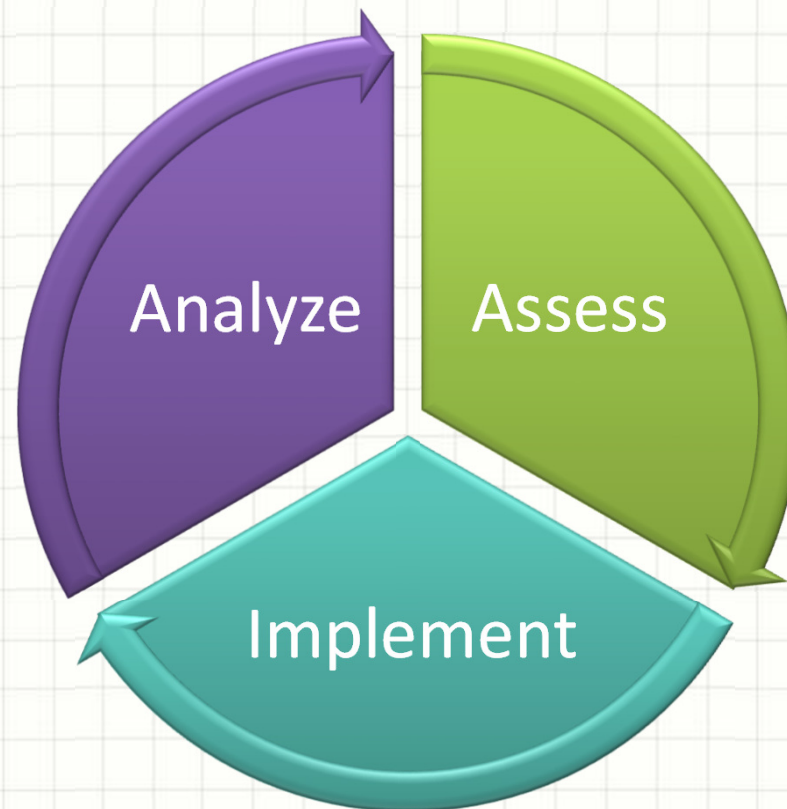
# Benchmark Architecture

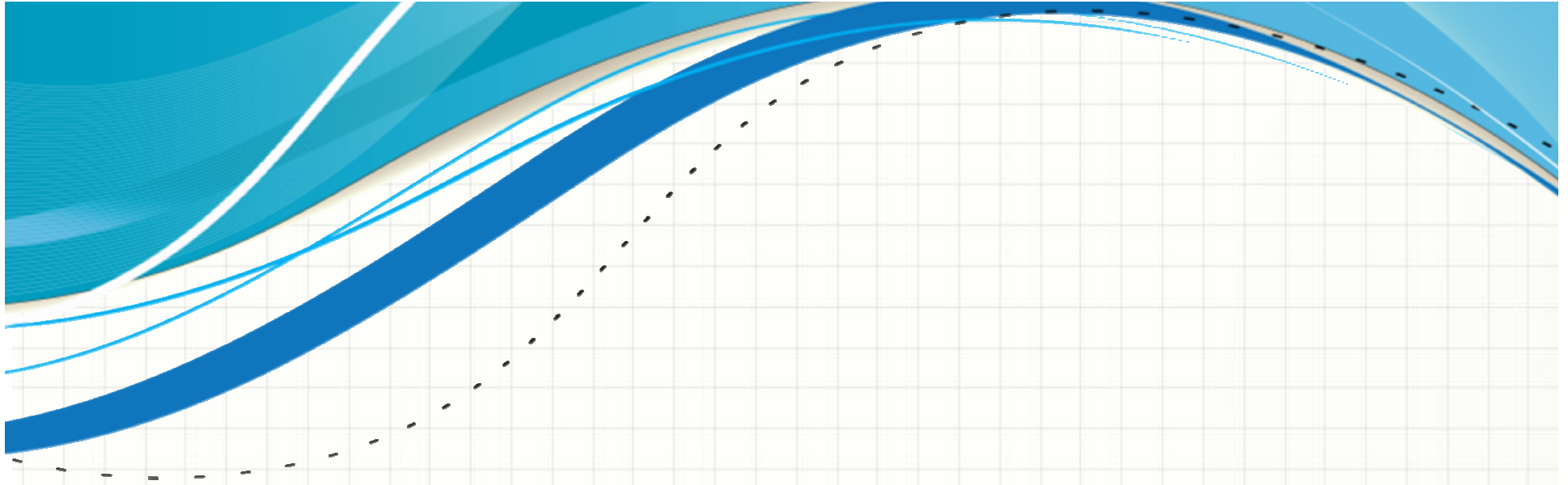




# Approach

- Short & self-contained improvement cycles
- From low-hanging fruits to advanced functionality





# **SCHEMA OPTIMIZATIONS**

# Schema Optimizations

- Replace wildcard queries with boolean (hasX) fields

Before	After
<code>attachmentName : *</code>	<code>hasAttachment : true</code>

- Avoids iterating over millions of posting lists
- Result:
  - Some queries improved by 20X (e.g. hasUrl)
  - Some queries were not affected (e.g. hasStatus)

# Schema Optimizations – Cont.

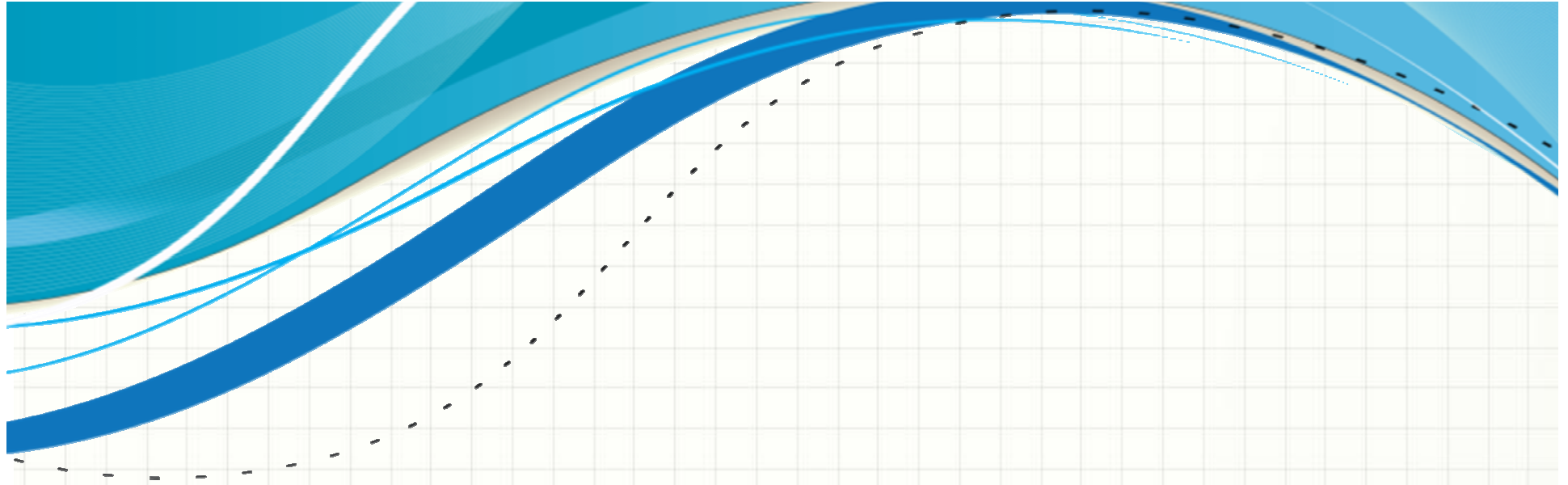
- **DocValues** for faceting / sorting
  - Densely packed column-oriented fields
  - Reduces memory consumption by field cache
  - Result: OOM problems solved!

	Field1	Field2	Field3
Doc1	1	2	3
Doc2	2	3	4
Doc3	4	3	2

Row-oriented (Stored Fields)

	Doc1	Doc2	Doc3
Field1	1	2	4
Field2	2	3	3
Field3	3	4	2

Column-oriented (docValues)



# QUERY OPTIMIZATIONS

# Query Optimizations

- **Filter Queries (fq)** instead of main query (q)
  - Cached independently in filter cache
  - Score is not affected by the filtering action

Before	After
<code>q = content:Apache AND sender:Bob AND deleted:false</code>	<code>q = content:Apache &amp; fq = sender:Bob &amp; fq = deleted:false</code>

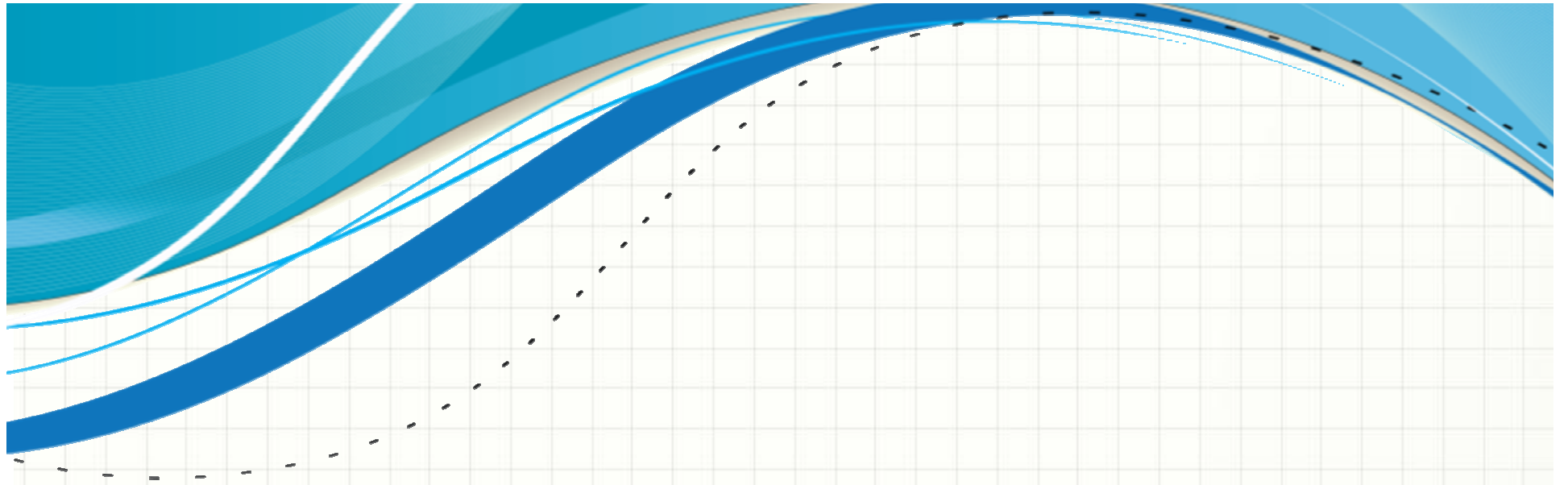
- **Use of Field List (fl)**
  - Specifies the list of fields to be returned
  - Saves bandwidth and simplifies parsing

# Query Optimizations – Cont.

- **Avoid grouping** queries when possible
  - Simple & useful (e.g. group emails by threads), yet time consuming
  - Consider query splitting

Before	After
<pre>q = content:apache &amp; group.query = folder:inbox &amp; group.query = *:*</pre>	<pre>Query 1: q = content:apache           &amp; fq = folder:inbox Query 2: q = content:apache</pre>

- **Result:** Query runtime improvement of up to 33%.



# INDEXING STRATEGY





# Indexing Strategy - Motivation

- **Observation:** Indexing emails is unique in the sense that mailboxes are private
- Data was indexed using various indexing strategies and several parameters were tested:
  - Indexing scenarios running time
  - Search scenarios running time
  - Server side behavior: CPU, Memory, GC

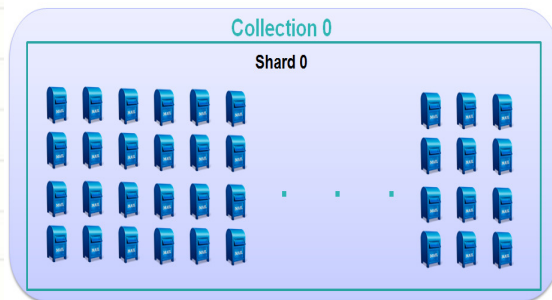


# User Mailboxes and Shards

- Data is divided into 5-node clusters
- User emails are assigned into a single cluster
- Initial designs indexed a user mailbox across all shards
- Solution: Each mailbox is assigned to a single shard
  - Reduces load on each Solr node
  - Significant improvement in query runtime

# Indexing Strategy Alternatives

1 Collection / 1 Shard



Collection per Mailbox



Shard per Mailbox




1 Collection / N Shards



N Collections / 1 Shard






# Indexing Strategy - Results

- 1 Collection / 1 Shard performed poorly
- Collection / Shard per mailbox strategies had the best performance, but failed to scale above ~2000 mailboxes
  - Solr / Zookeeper limitations
- Multi-collection strategies perform better than multi-shard strategies, and had better utilization of resources (CPU & Memory)
  - Query running time difference of 8-32%



# **SORTED INDEX**



# Sorted Index - Background

- **Observation:** Most queries require sorting by date
  - Example: Display emails in inbox
  - Default sorting is by doc ids
- **Solution:** Keep the index sorted by email dates
  - Sorting is not performed at query time
  - Early termination when requested #results is reached
  - The tradeoff: additional work during indexing

# Sorted Index - Implementation

- Two modules were developed to extend Solr:
  1. Sorting MergePolicy
  2. Early termination (with / without grouping queries)
- Contributed to Solr, available in Solr 6
  - [SOLR-5730](#) - Make Lucene's SortingMergePolicy and EarlyTerminatingSortingCollector configurable in Solr
  - [SOLR-8621](#) - solrconfig.xml: deprecate/replace <mergePolicy> with <mergePolicyFactory>



# Sorted Index - Results

- Runtime of queries that perform sorting by date improved by 6-23%
- Early termination in grouping queries did not show significant improvement
- No observable indexing time slowdown





# MULTI-TIERED INDEX



# Multi-Tiered Index - Background

- **Observation:** Users are often interested in their recent emails
  - Example: Inbox refresh, closest people etc.
- **Solution:** Index emails into tiers
  - Archive tier contains all emails
  - Recent tier contains recent emails
    - Last week, last month, last quarter etc.
  - Tiers can be placed on different HW

# Non-Tiered Strategy

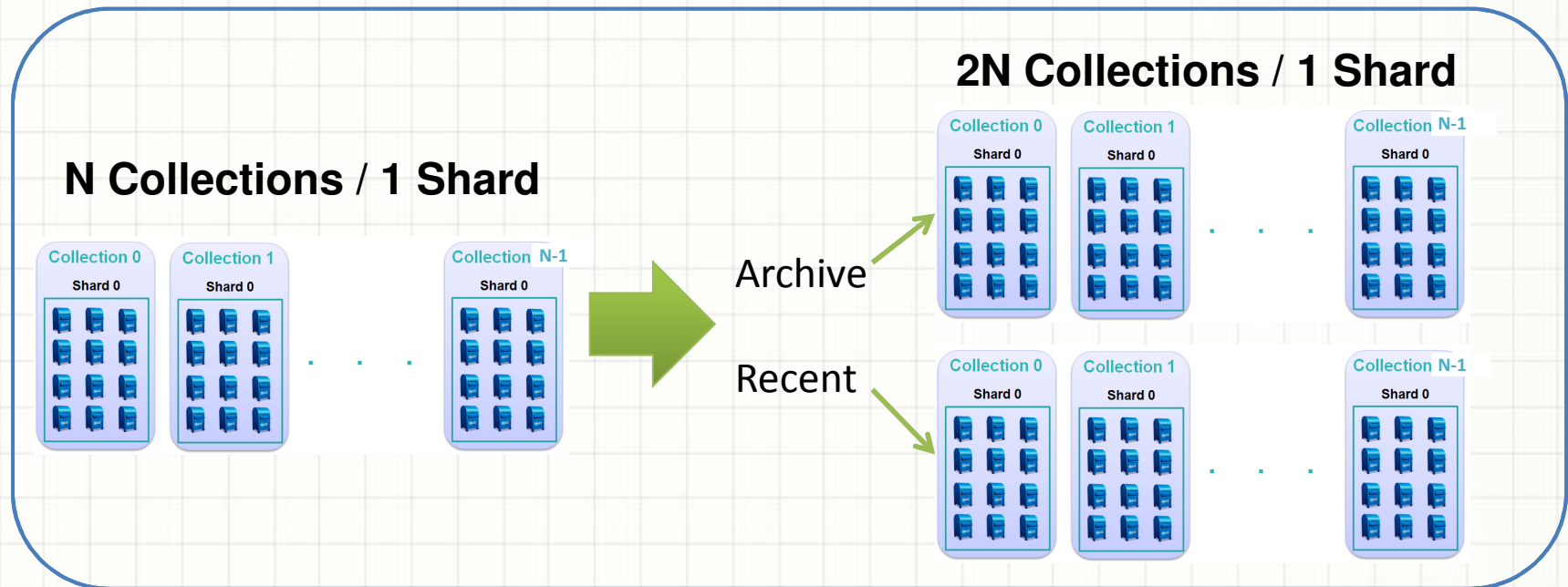
- N Collections
  - Single-shard
  - Each collection contains multiple mailboxes

## N Collections / 1 Shard

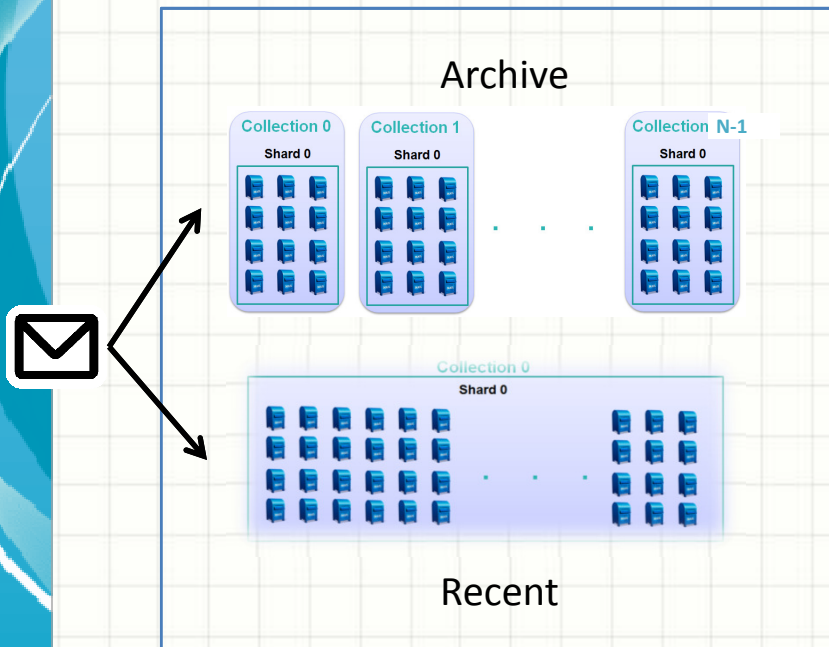


# Tiered Strategy

- N “Archive” collections + N “Last Month” collections
  - Indexing is performed on both tiers
  - Search is performed on the relevant tier per query date constraints



# Tiered Strategy - Alternatives

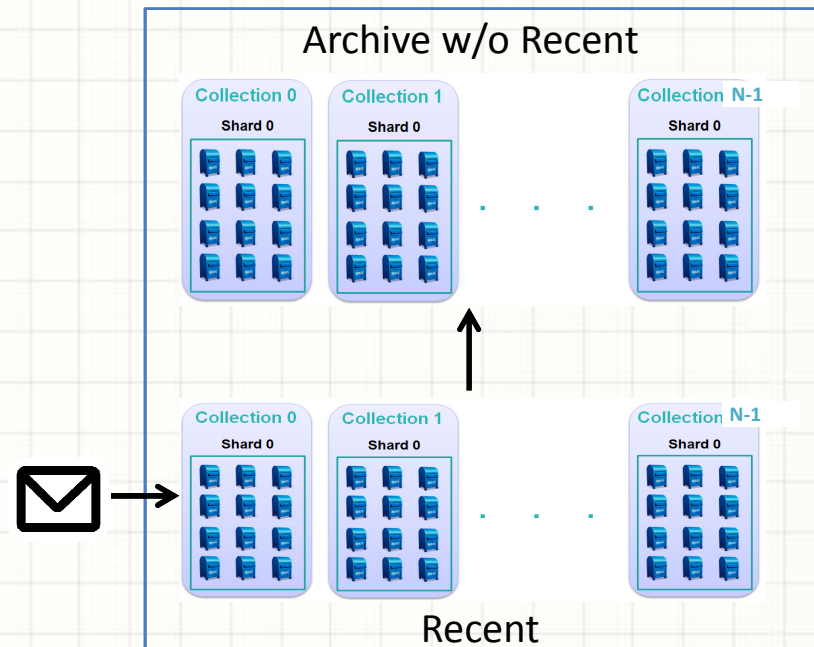


## Pros:

- Less collections to manage

## Cons:

- Does not perform as well as multi-collection recent tier



## Pros:

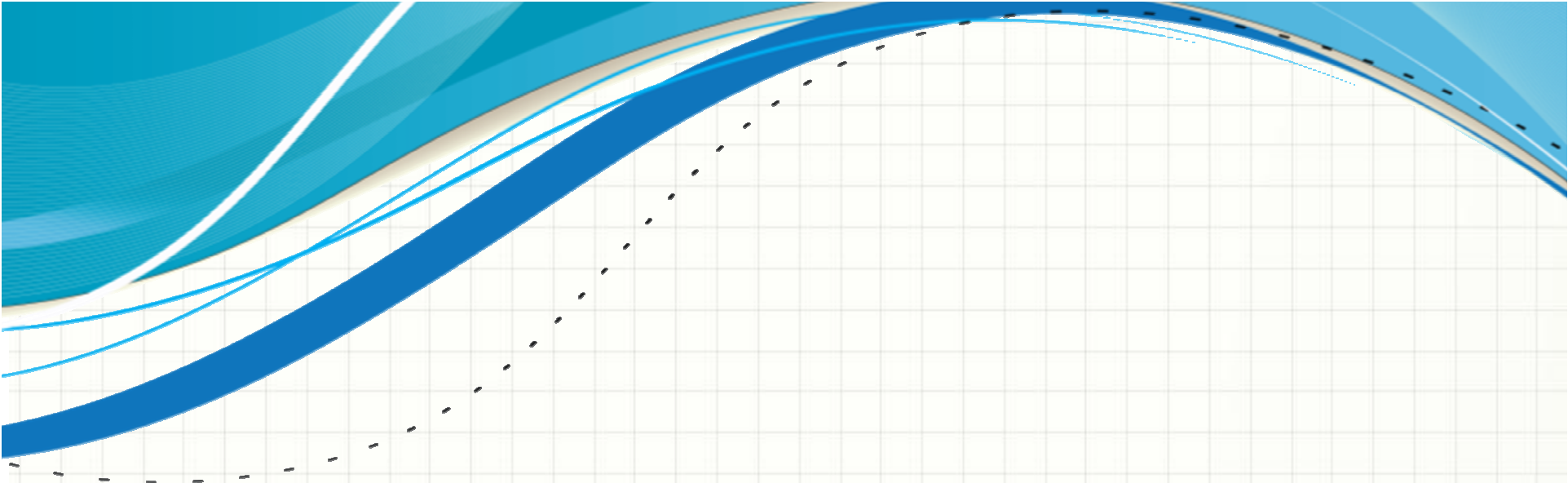
- Saves disk space
- Indexing to a single collection

## Cons:

- Archive queries must merge results
- Emails should move from recent to archive tiers

# Multi-Tiered Index - Results

- Queries that target the recent tier ran 2-15X faster
- Indexing scenarios experience a ~2X slowdown. However, since indexing running time is < 25ms, the effect on user experience is minor
- Client side changes are required in order to have more queries use the recent tier
  - Example: Keyword search



# **ADDITIONAL RECOMMENDATIONS**



# Hardware Configuration

- **Observation:** SSD infrastructure leads to better performance, but higher server costs
- As improved performance results in a larger server capacity, and SSD gets cheaper, the question is which configuration optimizes cost per user
- **Result:** Storing the Solr instances on SSDs results in a significant runtime improvement, which translates to major cost per user savings





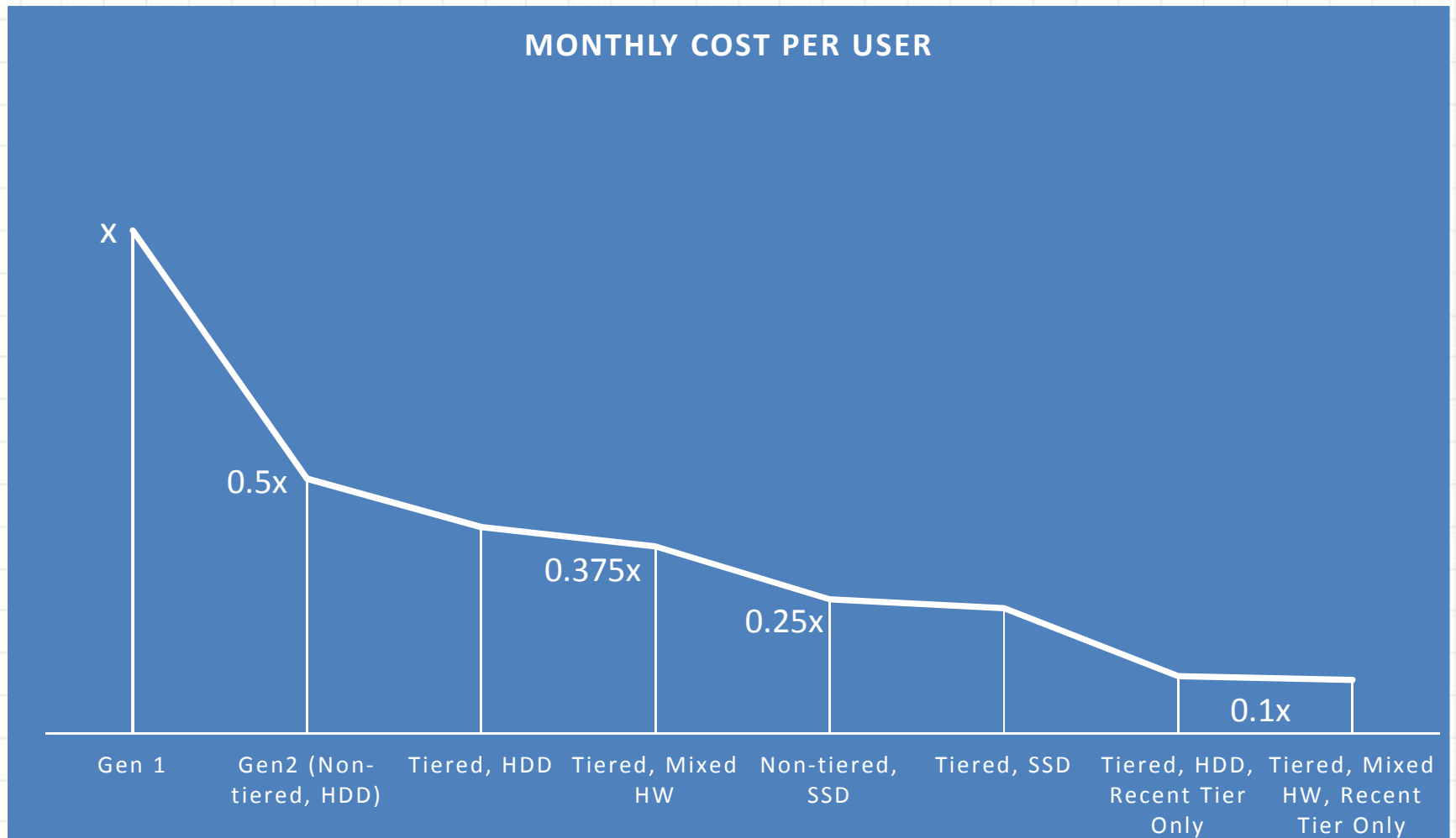
# Solr Version

- **Recommendation:** Upgrade Solr version (at least) with every major release
  - Ongoing support by the community
  - Bug fixes and improved stability
  - New features and APIs
  - Enables to extend Solr and contribute new code
- **Result:** Improved stability and performance



# SUMMARY

# Monthly Cost per User





# Lessons Learned

Make Solr perfect for **your** use case!

- Know your data
  - What does it contain? How is it structured?
- Know your user
  - Who uses the system? How is it used?
- Start simple, then go advanced
  - Schema & query optimizations
  - Indexing strategy
  - Advanced features
- Benchmark every change you make
  - Make an effort to build a reliable and flexible benchmarking framework



**THANK YOU!**

**ARNON YOGEV**  
**ARNONY@IL.IBM.COM**