MarkLogic®

Unit 8: Indexing

Learning Objectives

- Describe the following indexing concepts:
  - Universal and Term List Indexes
  - Range Index
  - Path Range Index
  - Word Query
  - Field
- Build Range Indexes.
- Export a database configuration with Configuration Manager.
- Automate index deployment with the Management REST API.

Indexing Concepts: Filtering

```
DOCUMENT 1
{
  "description":
  "Jack ran to the store."
}
```

```
DOCUMENT 3
{
  "description":
  "Jack drives to the market."
}
```

```
DOCUMENT 2
{
  "description":
  "Jill runs to the store."
}
```

```
DOCUMENT 4
{
  "description":
  "Jill, running up the hill."
}
```

- Which document(s) contain the word “market”?
- How did you determine the result?

Indexing Concepts: Term List / Inverted Index

TERM	DOCUMENT SET			
<description>	1	2	3	4
jack	1		3	
jill		2		4
ran	1			
runs		2		
running				4
drives			3	
to	1	2	3	
the	1	2	3	4
store	1	2		
market			**3**	
up				4
hill				4

- Which document(s) contain the word "market"?
  - How did you determine the result?
- Note that only word tokens are indexed.
  - No punctuation or whitespace.

Indexing Concepts: Stemming

TERM	DOCUMENT SET			
<description>	1	2	3	4
jack	1		3	
jill		2		4
run	**1**	**2**		**4**
drive			**3**	
to	1	2	3	
the	1	2	3	4
store	1	2		
market			3	
up				4
hill				4

- Which document(s) contain the word "running"?
- Which documents contain the word "ran"?
- How did you determine the result?
- What if a 5th document was added that contained the word "runner"?

MarkLogic

Exposing a Words Stem

- Stemming rules based on language, controlled by dictionaries (customizable).
- If no language specified, the default language set on database applies.

`cts.stem(“runner”);`	→	`runner`
`cts.stem(“ran”);`	→	`run`
`cts.stem(“chatting”, “en”);`	→	`chat`
`cts.stem(“chatting”, “fr”);`	→	`chatting`

MarkLogic

Indexing Concepts: AND Query

TERM	DOCUMENT SET			
<description>	1	2	3	4
jack	1		3	
jill		2		4
run	1	2		4
drive			3	
to	1	2	3	
the	1	2	3	4
store	1	2		
market			3	
up				4
hill				4

- Which document(s) contain both the words “jill” AND “hill”?
- Term list intersections

MarkLogic

Indexing Concepts: OR Query

TERM	DOCUMENT SET			
<description>	1	2	3	4
jack	1		3	
jill		2		4
run	1	2		4
drive			3	
to	1	2	3	
the	1	2	3	4
store	1	2		
market			3	
up				4
hill				4

- Which document(s) contain both the words “jill” OR “hill”?
- Term list unions

MarkLogic

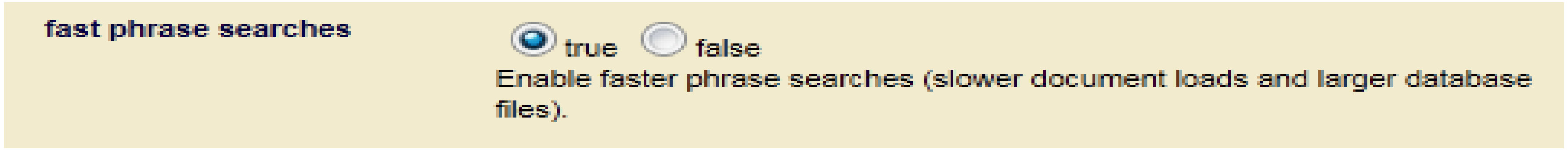
Indexing Concepts: NOT Query

TERM	DOCUMENT SET			
<description>	1	2	3	4
jack	1		3	
jill		2		4
run	1	2		4
drive			3	
to	1	2	3	
the	1	2	3	4
store	1	2		
market			3	
up				4
hill				4

- Which document(s) contains the word "jack" but not the word "run"?
- Term list subtractions

Indexing Concepts: Phrases

- Administration Tool → Databases → *YourDB* → Configure



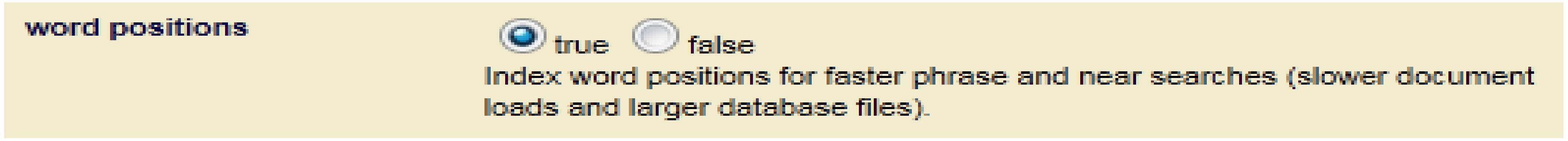
fast phrase searches [✓ true] false
Enable faster phrase searches (slower document loads and larger database files).

DOCUMENT 1
```
{
  "description":
  "Jack ran to the store."
}
```

TERM	DOCUMENT SET			
<description>	1	2	3	4
jack	1		3	
run	1	2		4
to	1	2	3	
the	1	2	3	4
store	1	2		
jack run	1			
run to	1			
to the	1	2	3	
the store	1	2		

Indexing Concepts: Proximity

- Administration Tool → Databases → *YourDB* → Configure



word positions [✓ true] false
Index word positions for faster phrase and near searches (slower document loads and larger database files).

```
DOCUMENT 1
{
  "description":
  "Jack ran to the store."
}
```

TERM	DOCUMENT SET			
<description>	1	2	3	4
jack	1:1		3:1	
run	1:2	2:2		4:2
to	1:3	2:3	3:3	
the	1:4	2:4	3:4	4:4
store	1:5	2:5		

Indexing Concepts: Structure

- Structure is indexed, including parent child relationships
- Fast resolution of XPath

```
DOCUMENT 1
{
  "bookstore":
  [{
    "book":
    {
      "title": "Moby Dick",
      "author": "H. Melville"
    }
  }]
}
```

TERM	DOCUMENT SET
bookstore/book	1
book/title	1
book/author	1
<title>:Moby Dick	1
<author>:H. Melville	1

MarkLogic

Indexing Concepts: Hashing

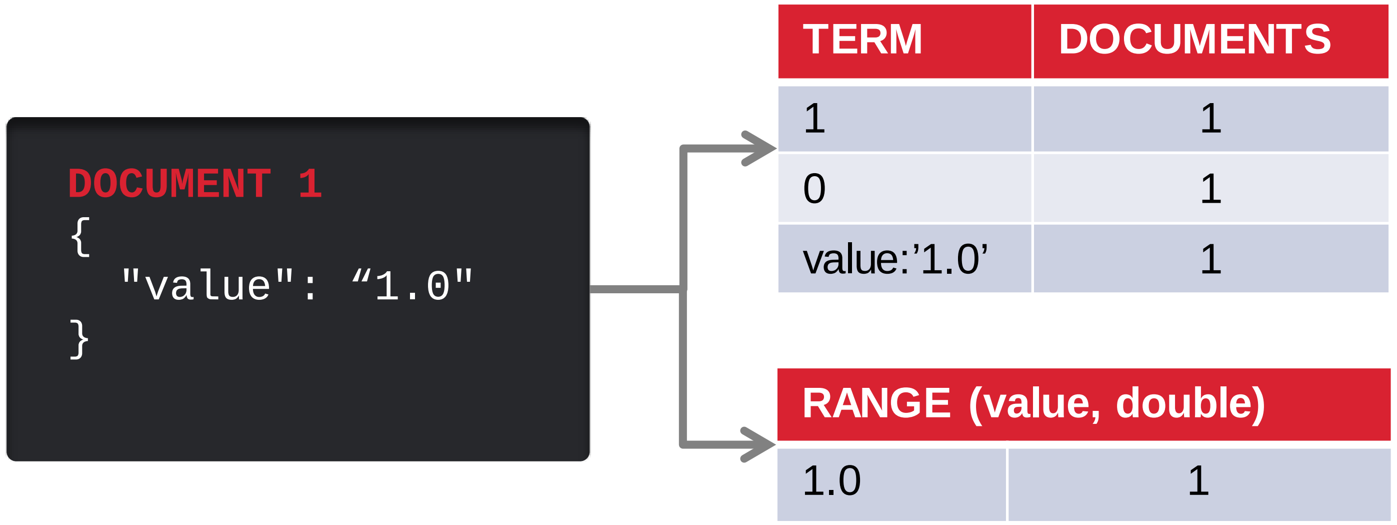
- To reduce size on disk, hashing is used for all term list keys.
- Hashing reduces text down to a smaller integer representation.
- Sizing:
  - XML / JSON + indexes can be smaller than original source data.
    - Why? Loaded XML and JSON is compressed in MarkLogic.
  - But with more indexes enabled, size most likely will increase.
    - Estimate size by turning on **your** desired indexes and loading a representative sample of **your** data.

Range Indexes

- Term lists are great at Yes / No type of questions
  - Map Values→Documents
- What about:
  - Find documents where the <price> is less than $50
  - Find documents with a <date> between 1990-01-01 and 1999-12-31
- Range Indexes…
  - Map Values←→Documents
  - Values (typed), not textual matches
  - Fast Range Queries
  - Fast Sorting
  - Fast Value Extraction
  - Faceting
- Range indexes live in memory when MarkLogic starts

Range Index vs. Term List Index

- Examples assume default indexes and unfiltered search.
  – Word Query: Find documents containing “1-0”→MATCH
  – Word Query: Find documents containing “0”→ MATCH
  – Range Index Query: Find documents containing “10”→NO MATCH



```
DOCUMENT 1
{
  "value": “1.0"
}
```

TERM	DOCUMENTS
1	1
0	1
value:’1.0’	1

RANGE (value, double)	
1.0	1

Element (Property) / Attribute Range Indexes

- Defined on a specific element or attribute
- Defined for a specific data type, sorted in value order

DOCUMENT A

```
{
  "top-song":
  {
    "artist": "the Beatles",
    "title": "Yesterday",
    "year": "1965-10-30"
  }
}
```

DOCUMENT B

```
{
  "top-song":
  {
    "artist": "the  beatles",
    "title": "Help!",
    "year": "1965-09-18"
  }
}
```

DOCUMENT A

```
{
  "top-song":
  {
    "artist": "Madonna",
    "title": "Take a Bow",
    "year": "1995-04-08"
  }
}
```

RANGE (artist)	
Madonna	C
the Beatles	A
the beatles	B

RANGE (date)	
1965-09-18	B
1965-10-30	A
1995-04-08	C

String Range Indexes & Collation

- Collations apply to String data type range indexes
- Determine what makes a unique value inside the index

```
DOCUMENT A
{
  "top-song":
  {
    "artist": "the Beatles",
    "title": "Yesterday",
    "year": "1965-10-30"
  }
}
```

```
DOCUMENT B
{
  "top-song":
  {
    "artist": "the  beatles",
    "title": "Help!",
    "year": "1965-09-18"
  }
}
```

```
DOCUMENT A
{
  "top-song":
  {
    "artist": "Madonna",
    "title": "Take a Bow",
    "year": "1995-04-08"
  }
}
```

RANGE(artist, default collation)	
Madonna	C
the Beatles	A
the beatles	B

RANGE(artist, punctuation, whitespace & case insensitive collation)	
madonna	C
the beatles	A, B

MarkLogic

Path Range Indexes

- More control over what the range index should contain

DOCUMENT A

```
{
  "book": {
    "title": "Moby Dick",
    "author": "Herman Melville",
    "chapter": [
      {
        "title": "Loomings",
        "text": "Call me Ishmael..."
      },
      {
        "title": "The Carpet-Bag",
        "text": "I stuffed a shirt..."
      }]
    }
}
```

RANGE (title)	
Moby Dick	A
Loomings	A
The Carpet-Bag	A

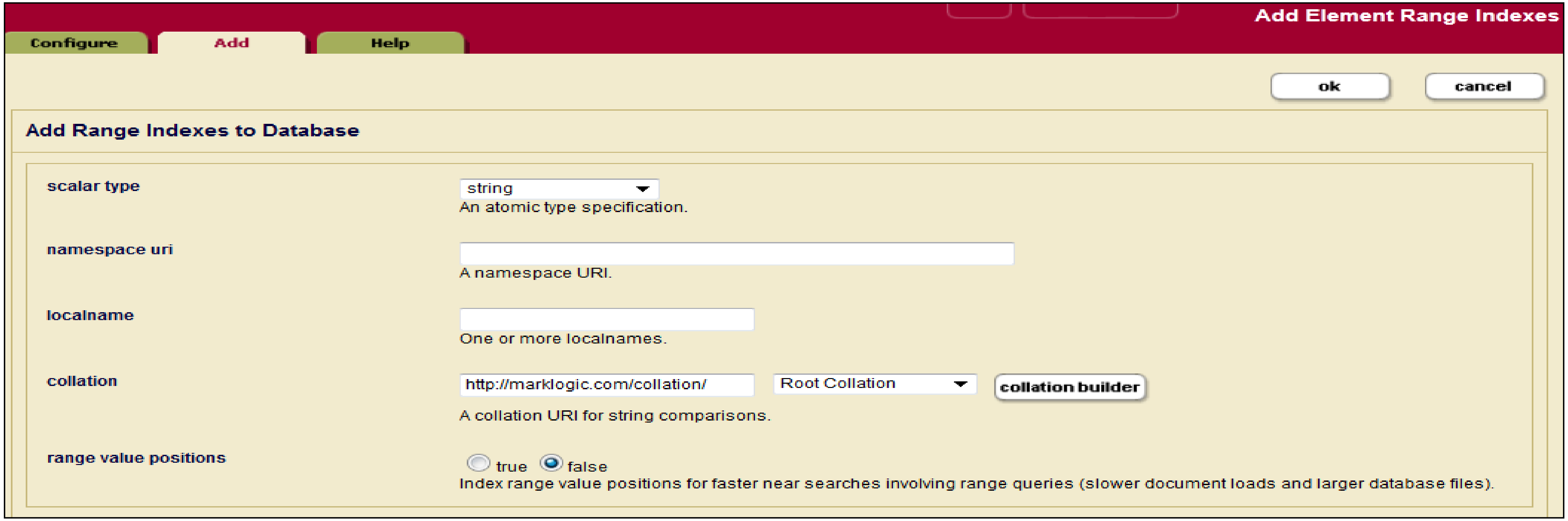
RANGE (chapter/title)	
Loomings	A
The Carpet-Bag	A

RANGE (book/title)	
Moby Dick	A

MarkLogic

Building Range Indexes

- Management REST API
- Admin Interface



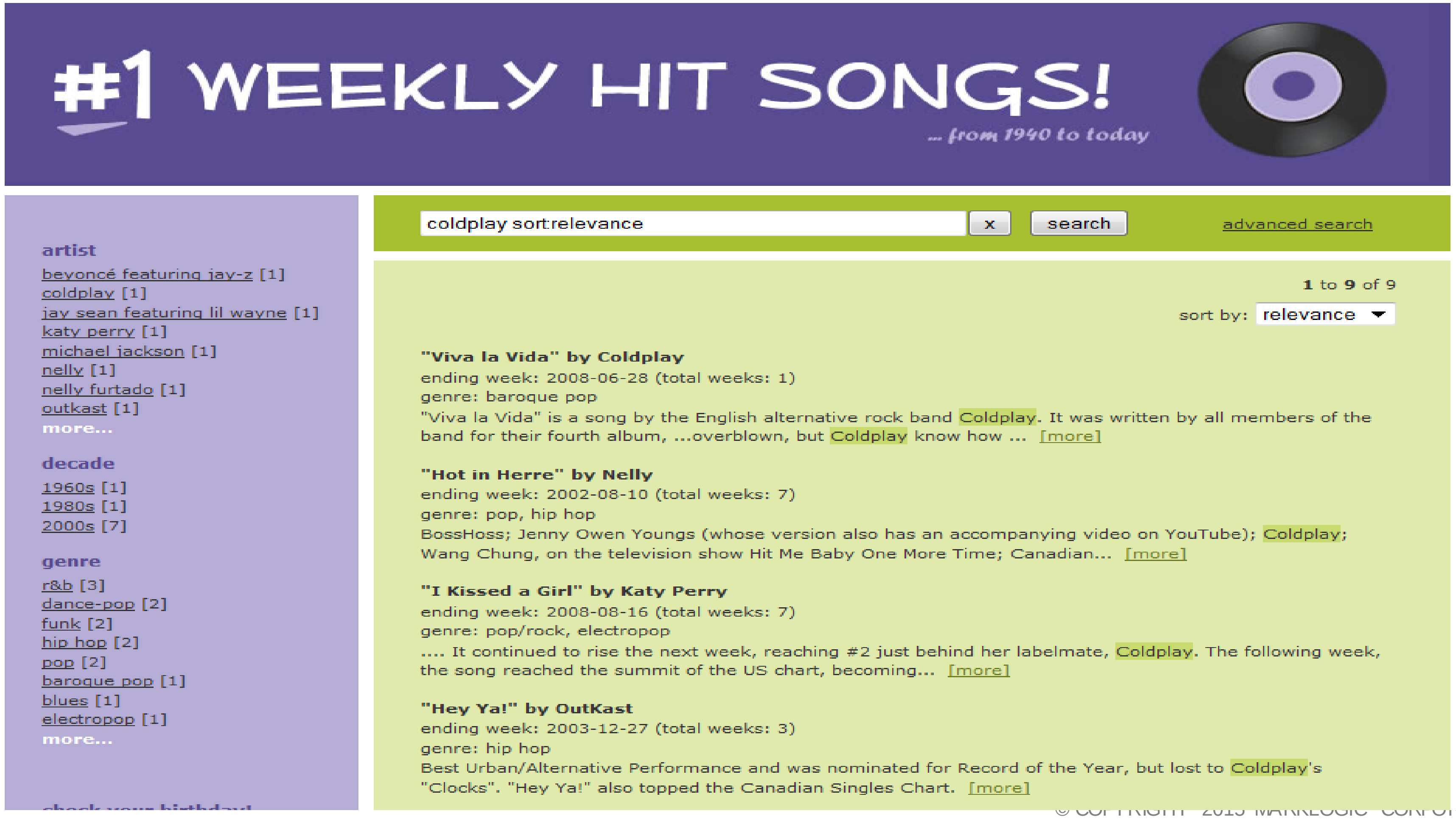
Add Range Indexes to Database

scalar type	string An atomic type specification.
namespace uri	A namespace URI.
localname	One or more localnames.
collation	http://marklogic.com/collation/ Root Collation collation builder A collation URI for string comparisons.
range value positions	true [✓ false] Index range value positions for faster near searches involving range queries (slower document loads and larger database files).

MarkLogic®

Indexing Concepts: Word Query

- Why does the Coldplay song appear first?



#1 WEEKLY HIT SONGS!
... from 1940 to today

coldplay sort:relevance [x] [search] advanced search

artist
beyoncé featuring jay-z [1]
coldplay [1]
jay sean featuring lil wayne [1]
katy perry [1]
michael jackson [1]
nelly [1]
nelly furtado [1]
outkast [1]
more...

decade
1960s [1]
1980s [1]
2000s [7]

genre
r&b [3]
dance-pop [2]
funk [2]
hip hop [2]
pop [2]
baroque pop [1]
blues [1]
electropop [1]
more...

1 to **9** of 9

sort by: relevance

"Viva la Vida" by Coldplay
ending week: 2008-06-28 (total weeks: 1)
genre: baroque pop
"Viva la Vida" is a song by the English alternative rock band Coldplay. It was written by all members of the band for their fourth album, ...overblown, but Coldplay know how ... [more]

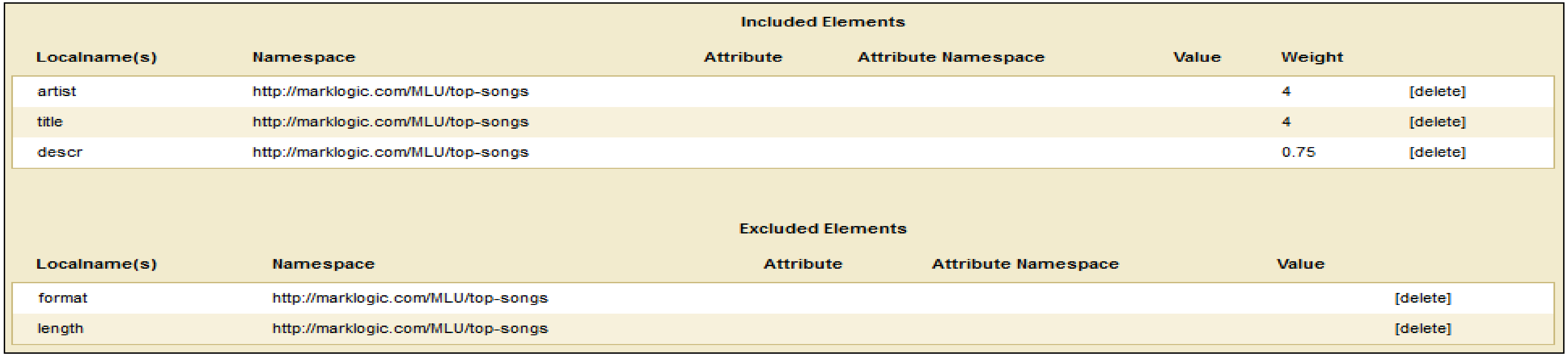
"Hot in Herre" by Nelly
ending week: 2002-08-10 (total weeks: 7)
genre: pop, hip hop
BossHoss; Jenny Owen Youngs (whose version also has an accompanying video on YouTube); Coldplay; Wang Chung, on the television show Hit Me Baby One More Time; Canadian... [more]

"I Kissed a Girl" by Katy Perry
ending week: 2008-08-16 (total weeks: 7)
genre: pop/rock, electropop
.... It continued to rise the next week, reaching #2 just behind her labelmate, Coldplay. The following week, the song reached the summit of the US chart, becoming... [more]

"Hey Ya!" by OutKast
ending week: 2003-12-27 (total weeks: 3)
genre: hip hop
Best Urban/Alternative Performance and was nominated for Record of the Year, but lost to Coldplay's "Clocks". "Hey Ya!" also topped the Canadian Singles Chart. [more]

Indexing Concepts: Word Query

- Why does the Coldplay song appear first?
  - Word Query is defined as follows on the database:



Included Elements

Localname(s)	Namespace	Attribute	Attribute Namespace	Value	Weight	
artist	http://marklogic.com/MLU/top-songs				4	[delete]
title	http://marklogic.com/MLU/top-songs				4	[delete]
descr	http://marklogic.com/MLU/top-songs				0.75	[delete]

Excluded Elements

Localname(s)	Namespace	Attribute	Attribute Namespace	Value	
format	http://marklogic.com/MLU/top-songs				[delete]
length	http://marklogic.com/MLU/top-songs				[delete]

MarkLogic

Fields – Use Cases

- Query portions of a database based on XML elements / JSON properties
  - Useful if you know that you query on specific parts of the document.
    - Ex: 80% of document data is used only on display, and only 20% is queried
- Unite XML elements / JSON properties across varying names.
  - Useful if you have many sources of data and they don't all refer to similar data points with the same markup.
- Setup using the Management REST API or Admin interface

MarkLogic

Fields - Example

```
DOCUMENT 1
{
  "top-song":
  {
    "artist":
    "The Beatles"
  }
}
```

```
DOCUMENT 2
{
  "top-song":
  {
    "singer":
    "Paul Simon"
  }
}
```

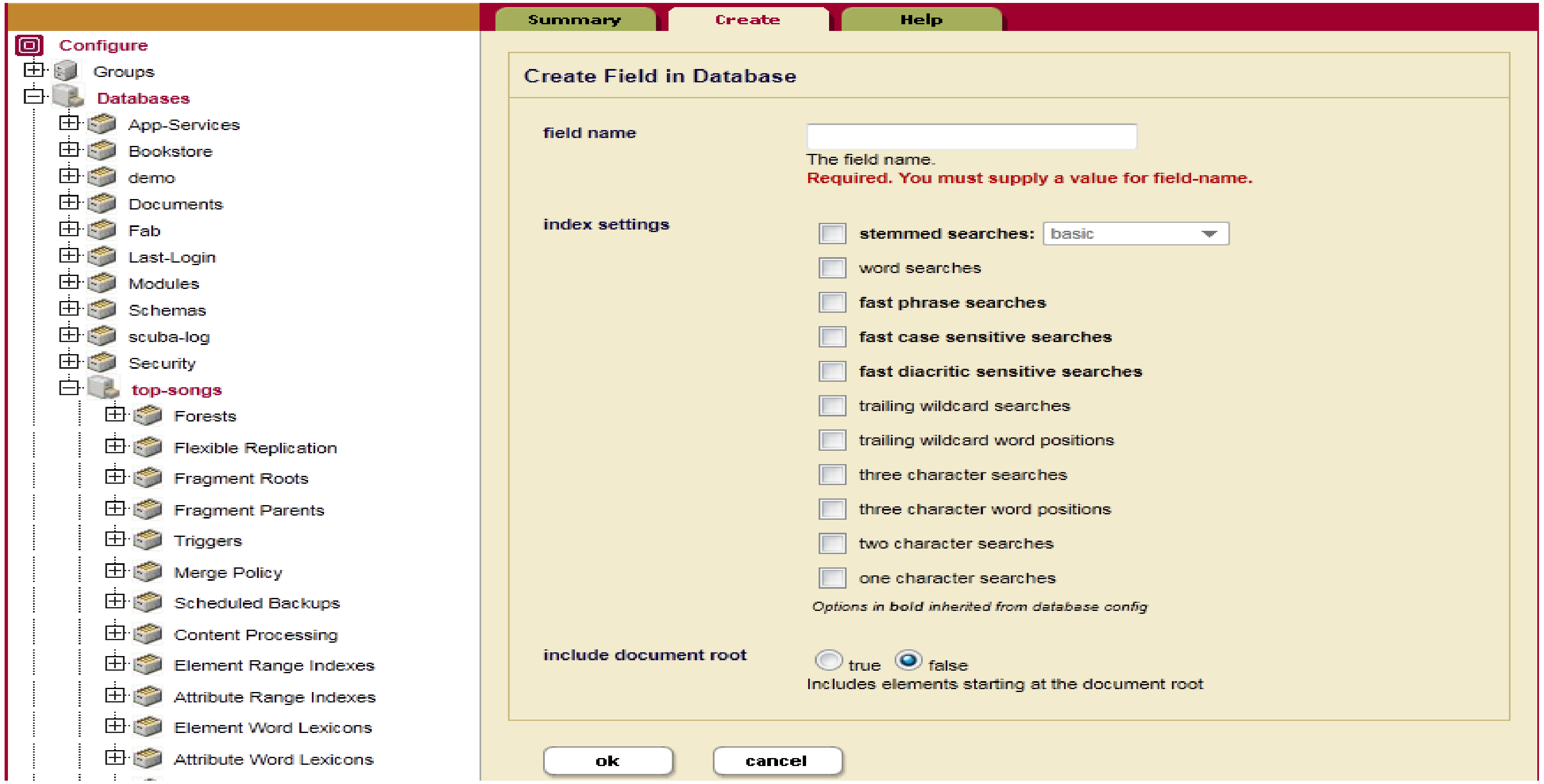
```
DOCUMENT 3
{
  "top-song":
  {
    "group":
    "Coldplay"
  }
}
```

```
DOCUMENT 4
{
  "top-song":
  {
    "band":
    "Radiohead"
  }
}
```

- Field Name: Performer
  - Include Elements (Properties):
    - <artist>|<singer>|<group>|<band>
  - Define Specific Index Settings

MarkLogic

Fields


Summary | Create | Help

Configure
Groups
Databases
App-Services
Bookstore
demo
Documents
Fab
Last-Login
Modules
Schemas
scuba-log
Security
top-songs
Forests
Flexible Replication
Fragment Roots
Fragment Parents
Triggers
Merge Policy
Scheduled Backups
Content Processing
Element Range Indexes
Attribute Range Indexes
Element Word Lexicons
Attribute Word Lexicons

Create Field in Database

field name
The field name.
Required. You must supply a value for field-name.

index settings
- [ ] **stemmed searches:** basic
- [ ] word searches
- [ ] **fast phrase searches**
- [ ] **fast case sensitive searches**
- [ ] **fast diacritic sensitive searches**
- [ ] trailing wildcard searches
- [ ] trailing wildcard word positions
- [ ] three character searches
- [ ] three character word positions
- [ ] two character searches
- [ ] one character searches

Options in bold inherited from database config

include document root true [✓ false]
Includes elements starting at the document root

ok | cancel

Indexing Concepts: Tuning

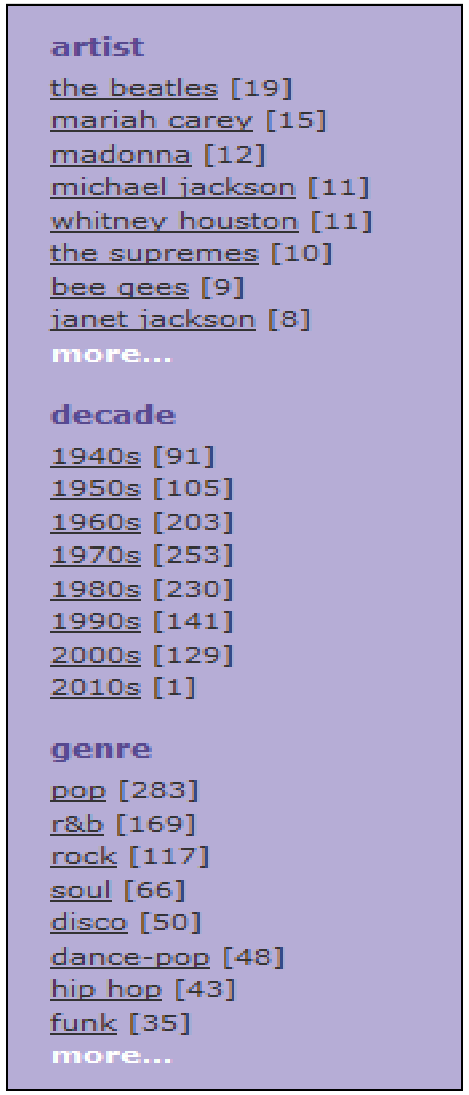
- `fn.count`
  - A 100% accurate count of your query results
  - Less efficient; requires filtering
- `xdmp.estimate`
  - May not be 100% accurate
  - Result based on indexes only
- Large gap between `fn.count` and `xdmp.estimate`?
  - Tune your query and/or indexes
  - Query Console Profile Function
  - `xdmp.query-meters()`
  - `xdmp.plan()`

Indexing Concepts: Summary

- Approach to Query Resolution
  - Look at the query
  - Decide what indexes can help
  - Use indexes to narrow down the result set
    - More indexes = tighter result set
  - Filter the result set to confirm the match
- Tradeoffs
  - More indexes = more time during ingestion
  - More indexes = greater storage size on disk
  - Less indexes = more filtering = slower search
  - Range Indexes costs RAM

MarkLogic

What indexes are required for certain app functions?


artist

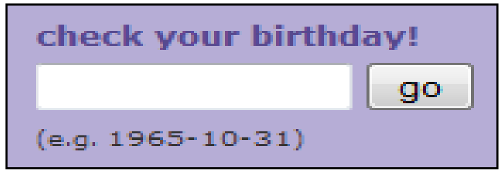
the beatles [19]
mariah carey [15]
madonna [12]
michael jackson [11]
whitney houston [11]
the supremes [10]
bee gees [9]
janet jackson [8]
more...

decade

1940s [91]
1950s [105]
1960s [203]
1970s [253]
1980s [230]
1990s [141]
2000s [129]
2010s [1]

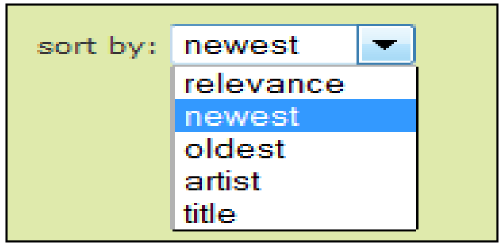
genre

pop [283]
r&b [169]
rock [117]
soul [66]
disco [50]
dance-pop [48]
hip hop [43]
funk [35]
more...

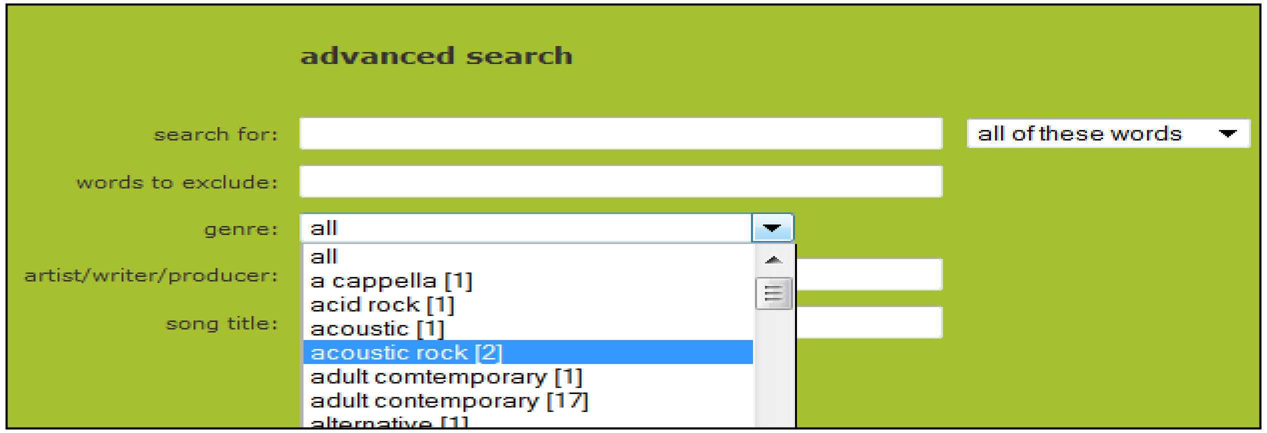

check your birthday!

go

(e.g. 1965-10-31)


sort by: newest

relevance
newest
oldest
artist
title


advanced search

search for: [all of these words]

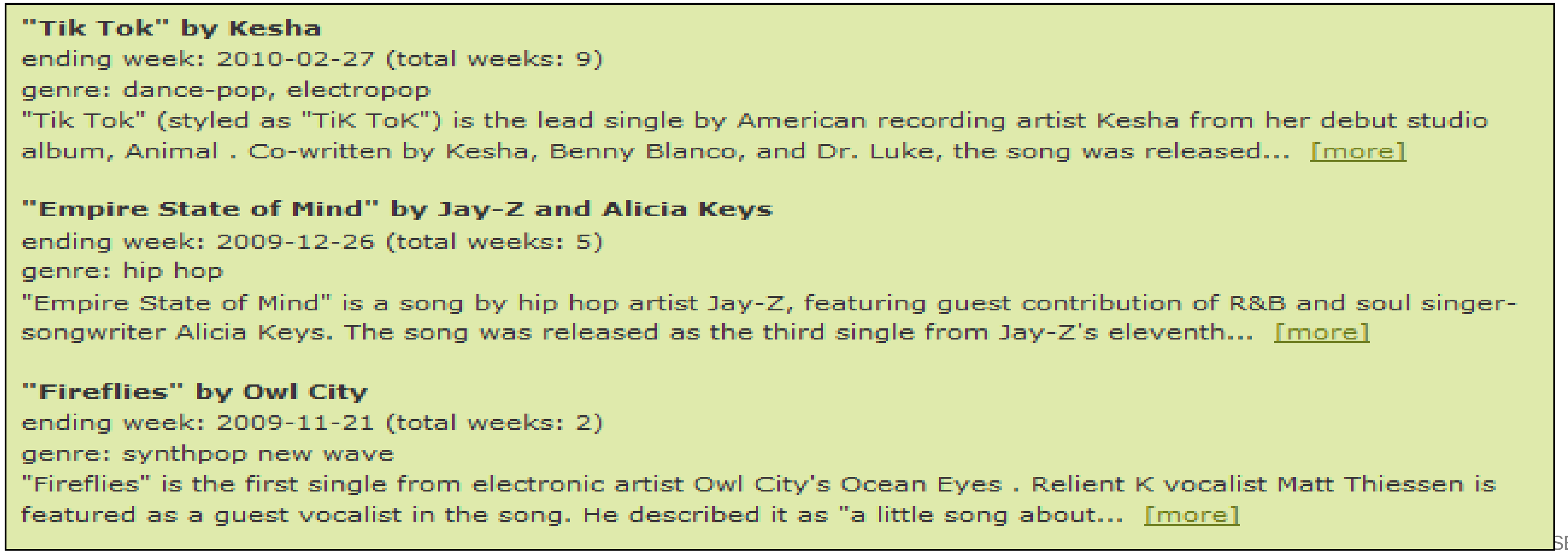
words to exclude:

genre: all

artist/writer/producer:

song title:

all
a cappella [1]
acid rock [1]
acoustic [1]
acoustic rock [2]
adult comtemporary [1]
adult contemporary [17]
alternative [1]


"Tik Tok" by Kesha
ending week: 2010-02-27 (total weeks: 9)
genre: dance-pop, electropop
"Tik Tok" (styled as "TiK ToK") is the lead single by American recording artist Kesha from her debut studio album, Animal . Co-written by Kesha, Benny Blanco, and Dr. Luke, the song was released... [more]

"Empire State of Mind" by Jay-Z and Alicia Keys
ending week: 2009-12-26 (total weeks: 5)
genre: hip hop
"Empire State of Mind" is a song by hip hop artist Jay-Z, featuring guest contribution of R&B and soul singer-songwriter Alicia Keys. The song was released as the third single from Jay-Z's eleventh... [more]

"Fireflies" by Owl City
ending week: 2009-11-21 (total weeks: 2)
genre: synthpop new wave
"Fireflies" is the first single from electronic artist Owl City's Ocean Eyes . Relient K vocalist Matt Thiessen is featured as a guest vocalist in the song. He described it as "a little song about... [more]

Demo: Samplestack Index Configuration
Demo: Filtered vs. Unfiltered Search

Labs: Unit 8

Exercise 1: Modify a Database Configuration
Exercise 2: Build a Range Index
Exercise 3: Automate Index Deployment with the Management REST API
Exercise 4: Capture a Database Configuration
DIY: Setup Star Wars Indexes

Unit Review Question 1:

Which of the following gets indexed by the Universal Index?

1. Word, whitespace, and punctuation tokens
2. Word and punctuation tokens
3. Word tokens
4. None of the above

Unit Review Question 1:

Which of the following gets indexed by the Universal Index?

1. Word, whitespace, and punctuation tokens
2. Word and punctuation tokens
3. **Word tokens**
4. None of the above

Unit Review Question 2:

A collation applies to which type of range index?

1. Date
2. Integer
3. Double
4. String

Unit Review Question 2:

A collation applies to which type of range index?

1. Date
2. Integer
3. Double
4. **String**

Unit Review Question 3:

Select all that apply:

Range indexes are…

1. Open in memory
2. Not persisted to disk
3. Sorted
4. Defined on data typed values

Unit Review Question 3:

Select all that apply:

Range indexes are…

1. **Open in memory**
2. Not persisted to disk
3. **Sorted**
4. **Defined on data typed values**

Unit Review Question 4:

You wish to be able to do wild card search on a few select properties in your database.

What plan of action would you choose:

1. Create string range indexes on each desired property
2. Turn on the wild card indexes for the database
3. Create a field and turn on its wild card indexes
4. Turn on fast phrase searches and word position indexes

Unit Review Question 4:

You wish to be able to do wild card search on a few select properties in your database.

What plan of action would you choose:

1. Create string range indexes on each desired property
2. Turn on the wild card indexes for the database
3. **Create a field and turn on its wild card indexes**
4. Turn on fast phrase searches and word position indexes