Improving the full-text index of QLever

Felix Meisen

2025-09-12

Introduction to QLevers full-text Index

What is a word scan?

Example *WordScan* query:

```
SELECT * WHERE {
     ?doc ql:contains-word "test" .
}
```

What is a word scan?

Example *WordScan* query:

```
SELECT * WHERE {
     ?doc ql:contains-word "test" .
}
```

Results excerpt:

?score	?doc
1	In Aristotelian science, especially in biology, things he saw himself . . .
1	Gray had reinvented the variable resistance telephone, . . .
1	Alternatively, although Bell had detected a slight sound . . .

What is an entity scan?

Example *EntityScan* query:

```
SELECT * WHERE {
     ?doc ql:contains-word "test" .
     ?doc ql:contains-entity ?entity .
}
```

What is an entity scan?

Example *EntityScan* query:

```
SELECT * WHERE {
     ?doc ql:contains-word "test" .
     ?doc ql:contains-entity ?entity .
}
```

Results excerpt:

?wordScore	?doc	?entityScore	?entity
1	In Aristotelian science, . . .	100	Aristotle
1	Alternatively, although Bell . . .	200	Alexander . . . Bell
1	Early computers and the Turing test	100	Alan Turing

Word scan with a prefix

WordScan query with a prefix:

```
SELECT * WHERE {
     ?doc ql:contains-word "test*" .
}
```

Word scan with a prefix

WordScan query with a prefix:

```
SELECT * WHERE {
     ?doc ql:contains-word "test*" .
}
```

Results excerpt:

?score	?matchingword	?doc
1	testament	On 27 November 1895, . . .
1	testament	In his one-page testament, . . .
1	tested	The engine was never . . .

Word scan with a short prefix

WordScan query with a short prefix:

```
SELECT * WHERE {
     ?doc ql:contains-word "a*" .
}
```

Word scan with a short prefix

WordScan query with a short prefix:

```
SELECT * WHERE {
     ?doc ql:contains-word "a*" .
}
```

Error processing query

No words found for the given prefix. This usually means that the prefix is smaller than the configured minimum prefix size. This range spans over 3200 blocks. In file "/home/flixtastic/Uni/Bachelor/EmpiricalTests/qlever-code/src/index/TextMetaData.cpp " at line 31

Your query was:

```
SELECT * WHERE {
  ?doc ql:contains-word "a*"
}
```

What is an index used for?

What is an index used for?

Query

What is an index used for?

Query → Result

What is an index used for?

Query → Result

"word"

What is an index used for?

Query → Result

"word" → {{Text 1, Score 1}, {Text 5, Score 5}}

Inverted Index

"test" $\mapsto$ Text Records: $\{1, 7\}$

"testing" $\mapsto$ Text Records: $\{3\}$

"probe" $\mapsto$ Text Records: $\{3\}$

Inverted Index

"test" $\mapsto$ Text Records: $\{1, 7\}$

"testing" $\mapsto$ Text Records: $\{3\}$

"probe" $\mapsto$ Text Records: $\{3\}$

- Good at retrieving information for a single word
- Cannot easily retrieve information for a prefix

Half-Inverted Index

Block 1

"probe" $\mapsto$ Text Record 3

Block 2

"test" $\mapsto$ Text Record 1

"testing" $\mapsto$ Text Record 3

"test" $\mapsto$ Text Record 7

Half-Inverted Index

Block 1

"probe" ↦ Text Record 3

Block 2

"test" ↦ Text Record 1

"testing" ↦ Text Record 3

"test" ↦ Text Record 7

- Good at retrieving information for a single word or prefix
- Returns Text Records in sorted order for a block

How will the half-inverted full-text index look?

Block 1:

Text Record Index	Word or Entity Index	Score	Word or Entity
1	0	1	`astronomer`
1	1	1	`astronomy`
2	0	0	`astronomer`
2	1	0	`astronomy`
1	0	0	`<Astronomer>`
1	1	0	`<Space>`
2	0	0	`<Astronomer>`
2	1	0	`<Space>`

Block 2:

Text Record Index	Word or Entity Index	Score	Word or Entity
1	2	1	`space`
1	2	1	`space`
1	0	0	`<Astronomer>`
1	1	0	`<Space>`
2	0	0	`<Astronomer>`
2	1	0	`<Space>`

Retrieval

Retrieval

For a *WordScan*:

- Look up in which block the word or prefix occurs

Retrieval

For a *WordScan*:

- Look up in which block the word or prefix occurs
- Read word list of the block

Retrieval

For a *WordScan*:

- Look up in which block the word or prefix occurs
- Read word list of the block
- Filter word list by queried range

Retrieval

For a *WordScan*:

- Look up in which block the word or prefix occurs
- Read word list of the block
- Filter word list by queried range
- Return human readable result

Retrieval

For a *WordScan*:

- Look up in which block the word or prefix occurs
- Read word list of the block
- Filter word list by queried range
- Return human readable result

For an *EntityScan*:

- Look up in which block the word or prefix occurs

Retrieval

For a *WordScan*:

- Look up in which block the word or prefix occurs
- Read word list of the block
- Filter word list by queried range
- Return human readable result

For an *EntityScan*:

- Look up in which block the word or prefix occurs
- Read entity list of the block

Retrieval

For a *WordScan*:

- Look up in which block the word or prefix occurs
- Read word list of the block
- Filter word list by queried range
- Return human readable result

For an *EntityScan*:

- Look up in which block the word or prefix occurs
- Read entity list of the block
- If fixed entity requested, filter entity list

Retrieval

For a *WordScan*:

- Look up in which block the word or prefix occurs
- Read word list of the block
- Filter word list by queried range
- Return human readable result

For an *EntityScan*:

- Look up in which block the word or prefix occurs
- Read entity list of the block
- If fixed entity requested, filter entity list
- Join *EntityScan* with a *WordScan* on the same text variable

Retrieval

For a *WordScan*:

- Look up in which block the word or prefix occurs
- Read word list of the block
- Filter word list by queried range
- Return human readable result

For an *EntityScan*:

- Look up in which block the word or prefix occurs
- Read entity list of the block
- If fixed entity requested, filter entity list
- Join *EntityScan* with a *WordScan* on the same text variable
- Return human readable result

Questions?

Input for full-text index

Input for full-text index

- Literals of the RDF vocabulary

Input for full-text index

- Literals of the RDF vocabulary
- The `wordsfile` and `docsfile`

The `docsfile`

`docsfile` excerpt:

`Document-Index`	Document
4	An astronomer is a scientist . . .
7	They look at stars, planets, . . .
22	Examples of topics or fields . . .
25	There are also related but distinct . . .

From `docsfile` to `wordsfile`

Document:

An astronomer is a scientist in the field of astronomy who concentrates their studies on a specific question or field outside of the scope of Earth.

Text Records:

- An astronomer is a scientist in the field of astronomy
- An astronomer is a scientist in the field of astronomy
- astronomy who concentrates their studies on a specific question or field outside the scope of Earth.
- astronomy who concentrates their studies on a specific question or field outside the scope of Earth.

The wordsfile

Text Record: "An astronomer is a scientist in the field of astronomy"

wordsfile excerpt:

Word or Entity	isEntity	TextRecordIndex	Score
astronomer	0	1	1
<Astronomer>	1	1	0
scientist	0	1	1
field	0	1	1
astronomy	0	1	1

Building the full-text Index

Building the full-text Index

1. Load RDF vocabulary

Building the full-text Index

1. Load RDF vocabulary
2. Build the text vocabulary

Building the full-text Index

1. Load RDF vocabulary
2. Build the text vocabulary
3. Optionally get scoring data for BM25 or TF-IDF

Building the full-text Index

1. Load RDF vocabulary
2. Build the text vocabulary
3. Optionally get scoring data for BM25 or TF-IDF
4. Calculate the block boundaries

Building the full-text Index

1. Load RDF vocabulary
2. Build the text vocabulary
3. Optionally get scoring data for BM25 or TF-IDF
4. Calculate the block boundaries
5. Build the half-inverted full-text index table

How will the half-inverted full-text index look?

Block 1:

Text Record Index	Word or Entity Index	Score	Word or Entity
1	0	1	`astronomer`
1	1	1	`astronomy`
2	0	0	`astronomer`
2	1	0	`astronomy`
1	0	0	`<Astronomer>`
1	1	0	`<Space>`
2	0	0	`<Astronomer>`
2	1	0	`<Space>`

Block 2:

Text Record Index	Word or Entity Index	Score	Word or Entity
1	2	1	`space`
1	2	1	`space`
1	0	0	`<Astronomer>`
1	1	0	`<Space>`
2	0	0	`<Astronomer>`
2	1	0	`<Space>`

From text record to table entry

Text Record 1: "An astronomer is a scientist in the field of astronomy"

From text record to table entry

Text Record 1: "An astronomer is a scientist in the field of astronomy"

astronomer → Score 1

scientist → Score 1

field → Score 1

astronomy → Score 1

From text record to table entry

Text Record 1: "An astronomer is a scientist in the field of astronomy"

astronomer → Score 1

scientist → Score 1

field → Score 1

astronomy → Score 1

Block-Index	Text-Record-Index	Word-Vocab-Index	Score
1	1	1	1
3	1	4	1
2	1	3	1
1	1	2	1

From text record to table entry

Text Record 1: "An astronomer is a scientist in the field of astronomy"

astronomer → Score 1

scientist → Score 1

field → Score 1

astronomy → Score 1

<Astronomer> → Score 0

Block-Index	Text-Record-Index	Word-Vocab-Index	Score
1	1	1	1
3	1	4	1
2	1	3	1
1	1	2	1

From text record to table entry

Text Record 1: "An astronomer is a scientist in the field of astronomy"

astronomer → Score 1

scientist → Score 1

field → Score 1

astronomy → Score 1

<Astronomer> → Score 0

Block-Index	Text-Record-Index	Word-Vocab-Index	Score
1	1	1	1
3	1	4	1
2	1	3	1
1	1	2	1

Block-Index	Text-Record-Index	Vocab-Index	Score
1	1	0	1
3	1	0	1
2	1	0	1

Building the half-inverted full-text index

Text Record 1: "An astronomer is a scientist in the field of astronomy"

BlockIndex	isEntity	TextRecord-Index	WordVocab-Index or VocabIndex	Score	Word or Entity
1	0	1	1	1	astronomer
3	0	1	4	1	scientist
2	0	1	3	1	field
1	0	1	2	1	astronomy
1	1	1	0	0	<Astronomer>
3	1	1	0	0	<Astronomer>
2	1	1	0	0	<Astronomer>

Building the half-inverted full-text index

Text Record 1: "An astronomer is a scientist in the field of astronomy"

Text Record 2: "An astronomer is a scientist in the field of astronomy"

BlockIndex	isEntity	TextRecord-Index	WordVocab-Index or VocabIndex	Score	Word or Entity
1	0	1	1	1	astronomer
3	0	1	4	1	scientist
2	0	1	3	1	field
1	0	1	2	1	astronomy
1	1	1	0	0	<Astronomer>
3	1	1	0	0	<Astronomer>
2	1	1	0	0	<Astronomer>
1	0	2	1	0	astronomer
3	0	2	4	0	scientist
2	0	2	3	0	field
1	0	2	2	0	astronomy
1	1	2	0	0	<Astronomer>
3	1	2	0	0	<Astronomer>
2	1	2	0	0	<Astronomer>

Building the half-inverted full-text index

Text Record 1: "An astronomer is a scientist in the field of astronomy"

Text Record 2: "An astronomer is a scientist in the field of astronomy"

BlockIndex	isEntity	TextRecord-Index	WordVocab-Index or VocabIndex	Score	Word or Entity
1	0	1	1	1	astronomer
1	0	1	2	1	astronomy
1	0	2	1	0	astronomer
1	0	2	2	0	astronomy
1	1	1	0	0	<Astronomer>
1	1	2	0	0	<Astronomer>
2	0	1	3	1	field
2	0	2	3	0	field
2	1	1	0	0	<Astronomer>
2	1	2	0	0	<Astronomer>
3	0	1	4	1	scientist
3	0	2	4	0	scientist
3	1	1	0	0	<Astronomer>
3	1	2	0	0	<Astronomer>

Building the full-text index

1. Load RDF vocabulary
2. Build the text vocabulary
3. Optionally get scoring data for BM25 or TF-IDF
4. Calculate the block boundaries
5. Build the half-inverted full-text index table
6. Sort the table

Building the full-text index

1. Load RDF vocabulary
2. Build the text vocabulary
3. Optionally get scoring data for BM25 or TF-IDF
4. Calculate the block boundaries
5. Build the half-inverted full-text index table
6. Sort the table
7. Write the table to file in blocks

Building the full-text index

1. Load RDF vocabulary
2. Build the text vocabulary
3. Optionally get scoring data for BM25 or TF-IDF
4. Calculate the block boundaries
5. Build the half-inverted full-text index table
6. Sort the table
7. Write the table to file in blocks
8. Build the 'docsDB'

Questions?

Block changes

Potential problems with old blocks

The 10 most common prefixes in the scientists dataset

Number of Results	Prefix
91,824	work
80,856	scie
79,722	publ
78,069	univ
54,155	comp
51,967	awar
51,965	inte
48,689	rese
47,413	book
44,533	phys

Building the full-text Index

- Load RDF vocabulary
- Build the text vocabulary
- Optionally get scoring data for BM25 or TF-IDF
- ~~Calculate the block boundaries~~
- Build the half-inverted full-text index table
- Sort the table
- Write the table to file in blocks
- Build the 'docsDB'

From text record to tables

Text Record 1: "An astronomer is a scientist in the field of astronomy"

From text record to tables

Text Record 1: "An astronomer is a scientist in the field of astronomy"

astronomer → Score 1

scientist → Score 1

field → Score 1

astronomy → Score 1

From text record to tables

Text Record 1: "An astronomer is a scientist in the field of astronomy"

astronomer → Score 1

scientist → Score 1

field → Score 1

astronomy → Score 1

Text-Record-Index	Word-Vocab-Index	Score
1	1	1
1	4	1
1	3	1
1	2	1

From text record to tables

Text Record 1: "An astronomer is a scientist in the field of astronomy"

astronomer → Score 1

scientist → Score 1

field → Score 1

astronomy → Score 1

<Astronomer> → Score 0

Text-Record-Index	Word-Vocab-Index	Score
1	1	1
1	4	1
1	3	1
1	2	1

From text record to tables

Text Record 1: "An astronomer is a scientist in the field of astronomy"

astronomer → Score 1

scientist → Score 1

field → Score 1

astronomy → Score 1

<Astronomer> → Score 0

Text-Record-Index	Word-Vocab-Index	Score
1	1	1
1	4	1
1	3	1
1	2	1

Word-Vocab-Index	Text-Record-Index	Vocab-Index	Score
1	1	0	1
4	1	0	1
3	1	0	1
2	1	0	1

Building the full-text Index with a set block size

Word	WordVocabIndex	TextRecordIndex	Score
astronomer	1	1	1
scientist	4	1	1
field	3	1	1
astronomy	2	1	1
astronomer	1	2	0
scientist	4	2	0
field	3	2	0
astronomy	2	2	0

Word	WordVocab-Index	VocabIndex	TextRecord-Index	Score	Entity
astronomer	1	0	1	0	<Astronomer>
scientist	4	0	1	0	<Astronomer>
field	3	0	1	0	<Astronomer>
astronomy	2	0	1	0	<Astronomer>
astronomer	1	0	2	0	<Astronomer>
scientist	4	0	2	0	<Astronomer>
field	3	0	2	0	<Astronomer>
astronomy	2	0	2	0	<Astronomer>

Building the full-text Index with a set block size

Word	WordVocabIndex	TextRecordIndex	Score
astronomer	1	1	1
astronomer	1	2	0
astronomy	2	1	1
astronomy	2	2	0
field	3	1	1
field	3	2	0
scientist	4	1	1
scientist	4	2	0

Word	WordVocab-Index	VocabIndex	TextRecord-Index	Score	Entity
astronomer	1	0	1	0	<Astronomer>
astronomer	1	0	2	0	<Astronomer>
astronomy	2	0	1	0	<Astronomer>
astronomy	2	0	2	0	<Astronomer>
field	3	0	1	0	<Astronomer>
field	3	0	2	0	<Astronomer>
scientist	4	0	1	0	<Astronomer>
scientist	4	0	2	0	<Astronomer>

Retrieval

Retrieval

For a *WordScan*:

- Look up in which blocks the word or prefix occurs

Retrieval

For a *WordScan*:

- Look up in which blocks the word or prefix occurs
- Read word lists of the blocks

Retrieval

For a *WordScan*:

- Look up in which blocks the word or prefix occurs
- Read word lists of the blocks
- Filter word lists by queried range

Retrieval

For a *WordScan*:

- Look up in which blocks the word or prefix occurs
- Read word lists of the blocks
- Filter word lists by queried range
- Merge intermediate results

Retrieval

For a *WordScan*:

- Look up in which blocks the word or prefix occurs
- Read word lists of the blocks
- Filter word lists by queried range
- Merge intermediate results
- Return human readable result

Retrieval

For a *WordScan*:

- Look up in which blocks the word or prefix occurs
- Read word lists of the blocks
- Filter word lists by queried range
- Merge intermediate results
- Return human readable result

For an *EntityScan*:

- Look up in which blocks the word or prefix occurs

Retrieval

For a *WordScan*:

- Look up in which blocks the word or prefix occurs
- Read word lists of the blocks
- Filter word lists by queried range
- Merge intermediate results
- Return human readable result

For an *EntityScan*:

- Look up in which blocks the word or prefix occurs
- Read entity lists of the blocks

Retrieval

For a *WordScan*:

- Look up in which blocks the word or prefix occurs
- Read word lists of the blocks
- Filter word lists by queried range
- Merge intermediate results
- Return human readable result

For an *EntityScan*:

- Look up in which blocks the word or prefix occurs
- Read entity lists of the blocks
- If fixed entity requested, filter entity lists

Retrieval

For a *WordScan*:

- Look up in which blocks the word or prefix occurs
- Read word lists of the blocks
- Filter word lists by queried range
- Merge intermediate results
- Return human readable result

For an *EntityScan*:

- Look up in which blocks the word or prefix occurs
- Read entity lists of the blocks
- If fixed entity requested, filter entity lists
- Merge intermediate results

Retrieval

For a *WordScan*:

- Look up in which blocks the word or prefix occurs
- Read word lists of the blocks
- Filter word lists by queried range
- Merge intermediate results
- Return human readable result

For an *EntityScan*:

- Look up in which blocks the word or prefix occurs
- Read entity lists of the blocks
- If fixed entity requested, filter entity lists
- Merge intermediate results
- Remove duplicate text record and entity combinations

Retrieval

For a *WordScan*:

- Look up in which blocks the word or prefix occurs
- Read word lists of the blocks
- Filter word lists by queried range
- Merge intermediate results
- Return human readable result

For an *EntityScan*:

- Look up in which blocks the word or prefix occurs
- Read entity lists of the blocks
- If fixed entity requested, filter entity lists
- Merge intermediate results
- Remove duplicate text record and entity combinations
- Join *EntityScan* with a *WordScan* on the same text variable

Retrieval

For a *WordScan*:

- Look up in which blocks the word or prefix occurs
- Read word lists of the blocks
- Filter word lists by queried range
- Merge intermediate results
- Return human readable result

For an *EntityScan*:

- Look up in which blocks the word or prefix occurs
- Read entity lists of the blocks
- If fixed entity requested, filter entity lists
- Merge intermediate results
- Remove duplicate text record and entity combinations
- Join *EntityScan* with a *WordScan* on the same text variable
- Return human readable result

Arbitrary prefix search

Example *WordScan* query:

```
SELECT * WHERE {
     ?t ql:contains-word "a*" .
}
```

Arbitrary prefix search

Example *WordScan* query:

```
SELECT * WHERE {
    ?t ql:contains-word "a*" .
}
```

WordScan query with a small prefix:

?score	?matchingword	?doc
1	astronomer	An astronomer is a scientist . . .
1	astronomy	An astronomer is a scientist . . .
0	astronomer	An astronomer is a scientist . . .

Document only full-text index building

Problem of the wordsfile

Document:

An astronomer is a scientist in the field of astronomy who concentrates their studies on a specific question or field outside of the scope of Earth.

Text Records:

- An astronomer is a scientist in the field of astronomy
- An astronomer is a scientist in the field of astronomy
- astronomy who concentrates their studies on a specific question or field outside the scope of Earth.
- astronomy who concentrates their studies on a specific question or field outside the scope of Earth.

Building the full-text Index

- Load RDF vocabulary
- Build the text vocabulary
- Optionally get scoring data for BM25 or TF-IDF
- Calculate the block boundaries
- Build the half-inverted full-text table
- Sort the table
- Write the table to file in blocks
- Build the 'docsDB'

Stop-word search

WordScan query with a stop word:

```
SELECT * WHERE {
     ?doc ql:contains-word "is" .
}
```

Stop-word search

WordScan query with a stop word:

```
SELECT * WHERE {
     ?doc ql:contains-word "is" .
}
```

Results excerpt:

?score	?doc
0	An astronomer is a . . .
0	The number of professional . . .
0	The American Astronomical . . .

Entity Scan

EntityScan query:

```
SELECT * WHERE {
     ?doc ql:contains-word "space" .
     ?doc ql:contains-entity <Astronomer> .
}
```

Entity Scan

EntityScan query:

```
SELECT * WHERE {
     ?doc ql:contains-word "space" .
     ?doc ql:contains-entity <Astronomer> .
}
```

Results excerpt:

?word Score	?doc	?entity Score
0	Karl Gordon Henize, Ph. D. . . .	0
0	David C. Jewitt (born 1958) . . .	0
0	Spencer Jones's successor . . .	0
0	Woolley is known for his . . .	0

Literal filtering

Problem with literal retrieving

WordScan query with a number:

```
SELECT * WHERE {
     ?doc ql:contains-word "1986" .
}
```

Problem with literal retrieving

WordScan query with a number:

```
SELECT * WHERE {
     ?doc ql:contains-word "1986" .
}
```

Results excerpt:

?score	?doc
1	-
1	-
1	-
1	-
1	-

Problem with adding all literals

Subject	Predicate	Object
Harcourt Arboretum	hasLongitude	"-1.1968"
(11024) 1986 QC1	label	"(11024) 1986 QC1"@en
(11024) 1986 QC1	label	"(11024) 1986 QC1"@vie
wordnet designer drug 103179489	hasGloss	"a psychoactive drug deliberately synthesized to avoid anti-drug laws; mimics the effects of a banned drug; law was revised in 1986 to ban designer drugs"@eng

Solution

Solution

- During the RDF index building: Filter and save all literals to a literal file

Solution

- During the RDF index building: Filter and save all literals to a literal file
- During full-text index building: Use the literal file to retrieve literals directly instead of parsing the whole RDF vocabulary

Solution

- During the RDF index building: Filter and save all literals to a literal file
- During full-text index building: Use the literal file to retrieve literals directly instead of parsing the whole RDF vocabulary
- During text scan retrieval: Use the literal file together with the RDF vocabulary to find and return literals

Query on a number

WordScan query with a number:

```
SELECT * WHERE {
     ?doc ql:contains-word "1986" .
}
```

Query on a number

WordScan query with a number:

```
SELECT * WHERE {
     ?doc ql:contains-word "1986" .
}
```

Results excerpt:

?score	?t
1	-1.1986
1	-105.1986
1	-106.1986
1	-11.1986
1	-112.1986

Query on a number

WordScan query with a number:

```
SELECT * WHERE {
     ?doc ql:contains-word "1986" .
}
```

Results excerpt:

?score	?t
1	-1.1986
1	-105.1986
1	-106.1986
1	-11.1986
1	-112.1986

Results excerpt:

?score	?t
1	"a psychoactive drug deliberately synthesized to avoid anti-drug laws; mimics the effects of a banned drug; law was revised in 1986 to ban designer drugs"

Conclusion

Improvements

- Remove minimal prefix size for *WordScan*s

Improvements

- Remove minimal prefix size for *WordScan*s
- Add simpler format for the text corpus input

Improvements

- Remove minimal prefix size for *WordScan*s
- Add simpler format for the text corpus input
- Add filtering when adding literals to the full-text index to improve result quality