# MySQL Fulltext Search

# Introduction to MySQL Full-Text Search

1. FULLTEXT is the index type of full-text index
2. InnoDB or MyISAM tables use Full-text indexes
3. Can be created only for VARCHAR, CHAR or TEXT columns

# Defining FULLTEXT Indexes for MySQL Full-Text Searching

CREATE FULLTEXT INDEX `idx\_movie\_title` ON `workshop`.`movie` (title);

MySQL Natural Language Full-Text Searches

Sorted by relevance.

SELECT \*

FROM movie

WHERE

MATCH (title) AGAINST ('Star First Last' IN NATURAL LANGUAGE MODE);

SELECT \*,

MATCH (title) AGAINST ('Star First Last' IN NATURAL LANGUAGE MODE) as score

FROM movie;

Score calculation math <https://dev.mysql.com/doc/internals/en/full-text-search.html>

<https://mysqlserverteam.com/rankings-with-innodb-full-text-search/>

# Introduction to MySQL Boolean full-text searches

Not sorted by relevance

+ Include, the word must be present.

– Exclude, the word must not be present.

> Include, and increase ranking value.

< Include, and decrease the ranking value.

() Group words into subexpressions (allowing them to be included, excluded, ranked, and so forth as a group).

~ Negate a word’s ranking value.

\* Wildcard at the end of the word.

“” Defines a phrase (as opposed to a list of individual words, the entire phrase is matched for inclusion or exclusion).

SELECT \*,

MATCH (name) AGAINST ('Lexi Alexander' IN BOOLEAN MODE) as score

FROM person

ORDER BY score DESC;

SELECT \*,

MATCH (name) AGAINST ('"Lexi Alexander"' IN BOOLEAN MODE) as score

FROM person

ORDER BY score DESC;

SELECT \*,

MATCH (name) AGAINST ('+Lexi +Alexander"' IN BOOLEAN MODE) as score

FROM person

ORDER BY score DESC;

SELECT \*,

MATCH (name) AGAINST ('"Lexi Alexander"' IN BOOLEAN MODE) as score

FROM person

ORDER BY score DESC;

SELECT \*,

MATCH (name) AGAINST ('"Lexi Alexander"' IN BOOLEAN MODE) as score,

MATCH (name) AGAINST ('"Lexi Alexander" @2' IN BOOLEAN MODE) as score2

FROM person

ORDER BY score DESC,

score2 DESC;

SELECT \*,

MATCH (name) AGAINST ('Lexi Alexander' IN BOOLEAN MODE) as score

FROM person

ORDER BY score DESC,

levenshtein('Lexi Alexander', name) ASC

LIMIT 20;

SELECT \* FROM (

SELECT \*, MATCH (name) AGAINST ('Lexi Alexander' IN BOOLEAN MODE) as score

FROM person

HAVING score > 0

ORDER BY score DESC

LIMIT 20

) s ORDER BY score DESC, levenshtein('Lexi Alexander', name) ASC;

SELECT \*,

MATCH (name) AGAINST ('-Lexi +Alexander' IN BOOLEAN MODE) as score

FROM person

ORDER BY score DESC;

# Using MySQL Query Expansion

SELECT productName

FROM products

WHERE MATCH(productName) AGAINST('1992');

SELECT productName

FROM products

WHERE MATCH(productName)

AGAINST('1992' WITH QUERY EXPANSION);

# What is next?

Elasticsearch, Sphinx, Solr..