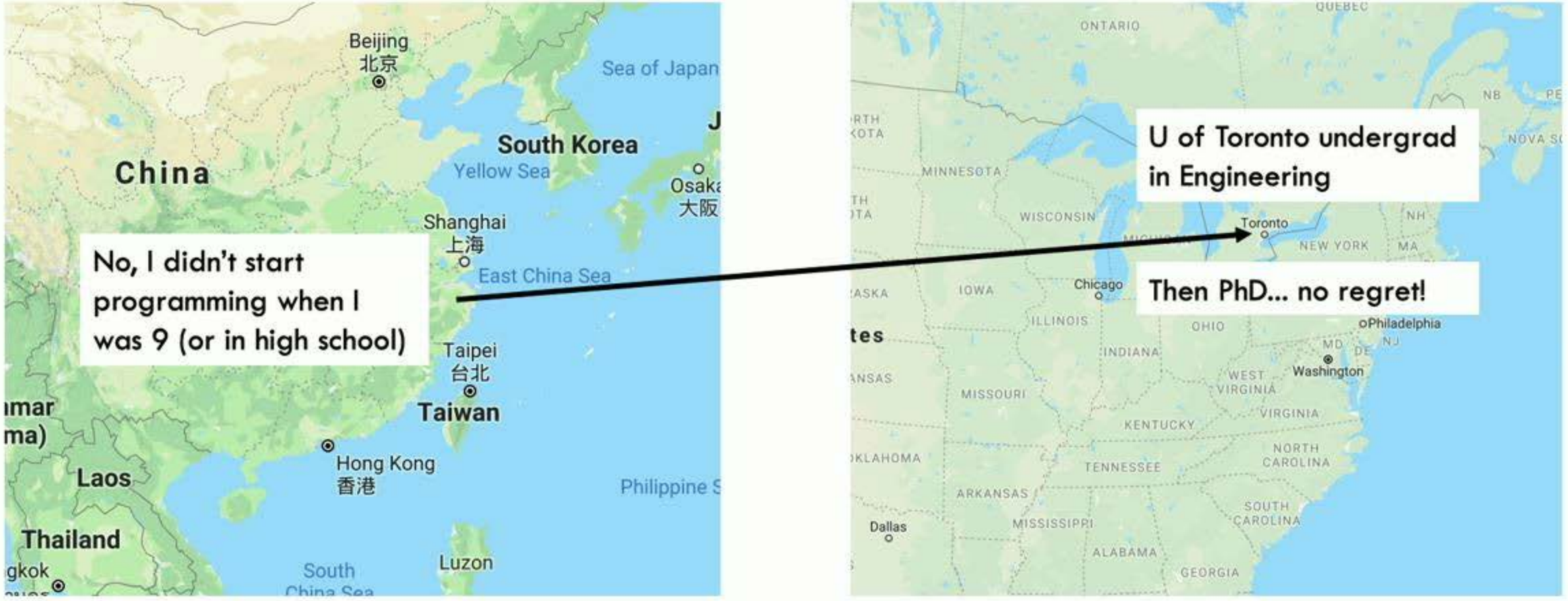
SEARCH FOR JOINABLE TABLES IN DATA LAKES

Erkang (Eric) Zhu, PhD Candidate
University of Toronto

ABOUT ME


Beijing
北京
Sea of Japan
South Korea
China
Yellow Sea
Osaka
大阪
Shanghai
上海
East China Sea
No, I didn't start programming when I was 9 (or in high school)
Taipei
台北
Taiwan
Hong Kong
香港
Laos
Philippine S
Thailand
Luzon
South China Sea
ONTARIO
QUEBEC
MINNESOTA
WISCONSIN
Toronto
NEW YORK
U of Toronto undergrad in Engineering
Chicago
IOWA
ILLINOIS
OHIO
Philadelphia
Then PhD... no regret!
INDIANA
Washington
MISSOURI
WEST VIRGINIA
VIRGINIA
KENTUCKY
NORTH CAROLINA
TENNESSEE
ARKANSAS
SOUTH CAROLINA
Dallas
MISSISSIPPI
ALABAMA
GEORGIA

ABOUT ME


Calgary
Intern at DMX Group @
MSR in 2016
Victoria
Seattle
WASHINGTON
Portland
OREGON
IDA

ABOUT ME


Calgary
Intern at DMX Group @ MSR in 2016
Victoria
Seattle
WASHINGTON
Winner of DMX Cup
Portland
OREGON
IDA


DMX CUP 2016
1st PLACE

Traditional Enterprise Analytics – Heavy IT

Intelligence,
Insights

Data Scientist

Data

Extract, Transform, Load

Data

DataBase Administrator

Data

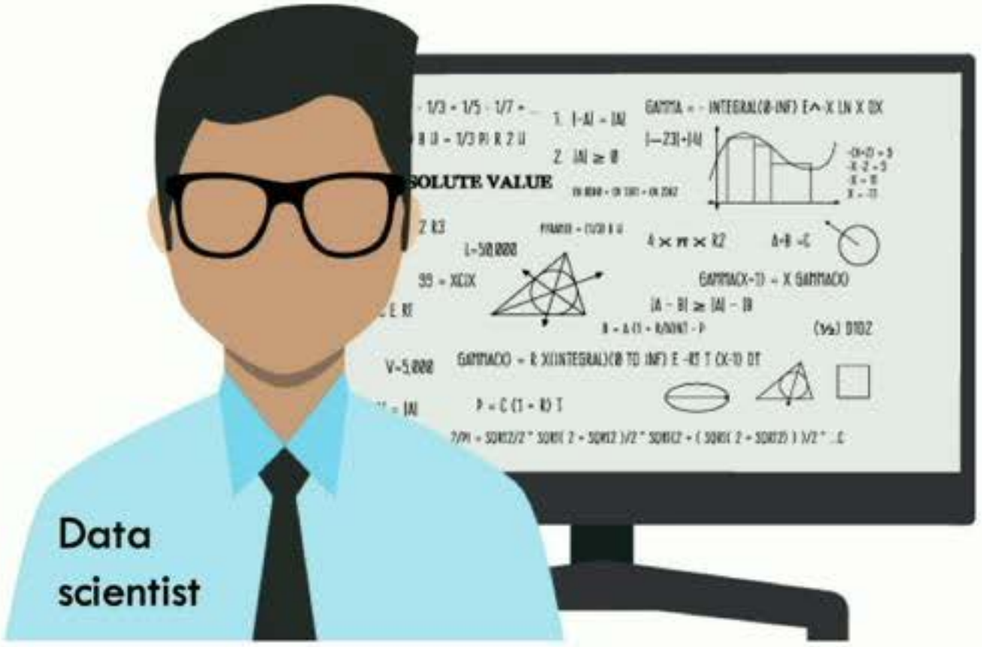
Data Lake Solution – Minimal IT

Intelligence,
Insights

Data Scientist

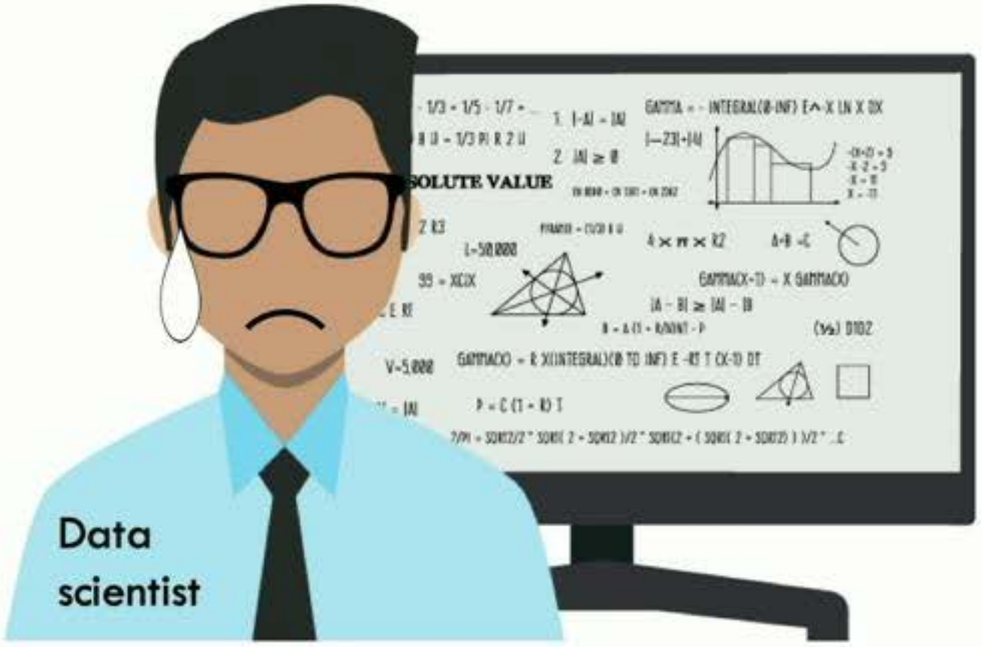
Data

Really?



Company	Headquarter	Ticker
Alphabet Inc.	Mountain View	GOOGL (NASDAQ)
Apple Inc.	Palo Alto	AAPL (NASDAQ)
Alibaba	Hangzhou	BABA (NYSE)
…	…	…

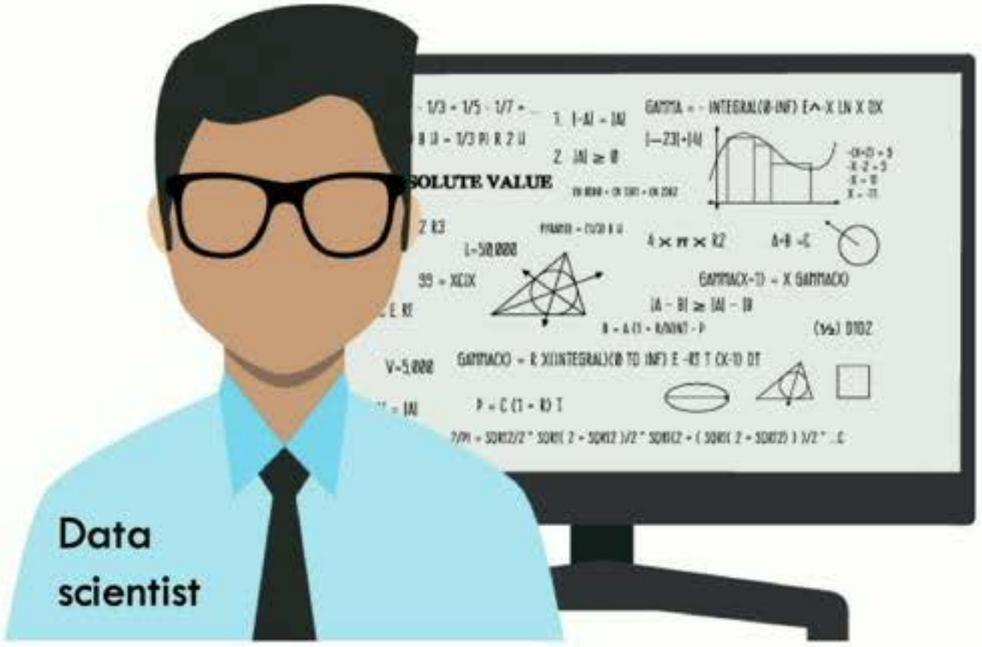
Question: More information about these companies?



Company	Headquarter	Ticker
Alphabet Inc.	Mountain View	GOOGL (NASDAQ)
Apple Inc.	Palo Alto	AAPL (NASDAQ)
Alibaba	Hangzhou	BABA (NYSE)
…	…	…

Question: More information about these companies?

Query: Tables that can be joined with my own table?

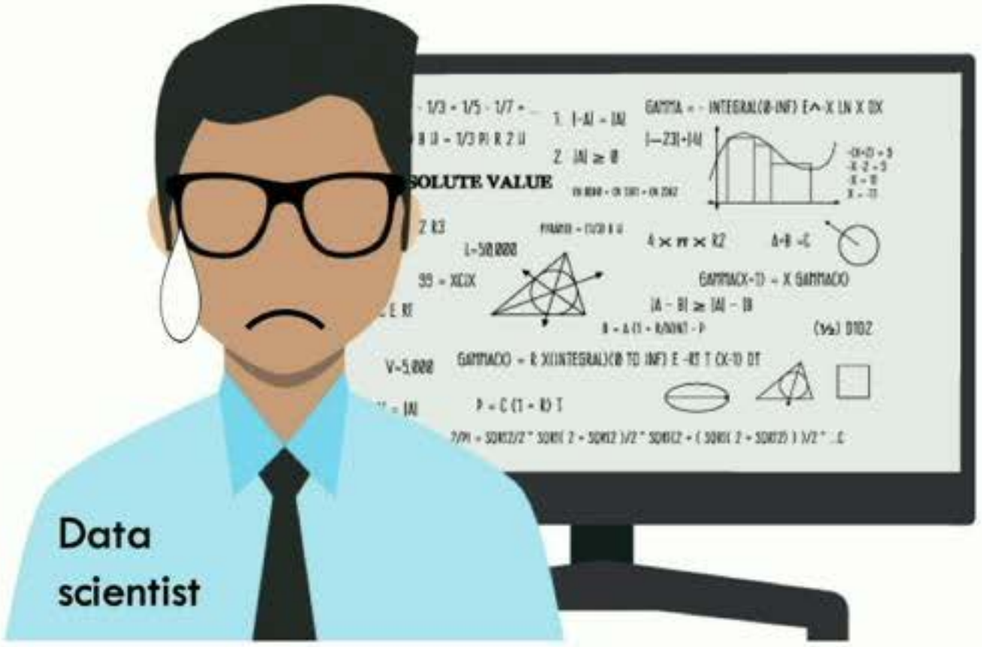


Task: write *ad hoc* data transformation script.

15 min

Company	Headquarter	Ticker
Alphabet Inc.	Mountain View	GOOGL (NASDAQ)
Apple Inc.	Palo Alto	AAPL (NASDAQ)
Alibaba	Hangzhou	BABA (NYSE)
…	…	…

Exchange	Ticker	Open	Close	Market Cap
NASDAQ	GOOGL	1,032.50	1,042.32	766 B
NASDAQ	AAPL	171.10	175.12	892 B
NYSE	BABA	179.37	174.13	427 B
…	…	…		



Task: write *ad hoc* data transformation script.

15 min **X 100**

Company	Headquarter	Ticker
Alphabet Inc.	Mountain View	GOOGL (NASDAQ)
Apple Inc.	Palo Alto	AAPL (NASDAQ)
Alibaba	Hangzhou	BABA (NYSE)
…	…	…

Exchange	Ticker	Open	Close	Market Cap
NASDAQ	GOOGL	1,032.50	1,042.32	766 B
NASDAQ	AAPL	171.10	175.12	892 B
NYSE	BABA	179.37	174.13	427 B
…	…	…		

"Jack of all trades"

Intelligence,
Insights

Data

Data

Data

Data Analytics

Transformation Scripts

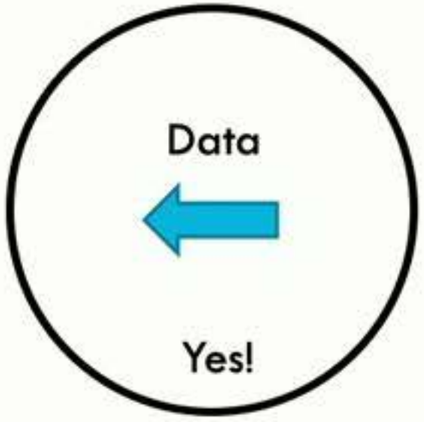
Locate Raw Files

Data Lake

CONTRIBUTIONS

Algorithmic solutions that enable data scientists to **search** for joinable tables and automatically **join** them – making data scientists more powerful.

Data Scientist



Data Lake

JOINABLE TABLE SEARCH

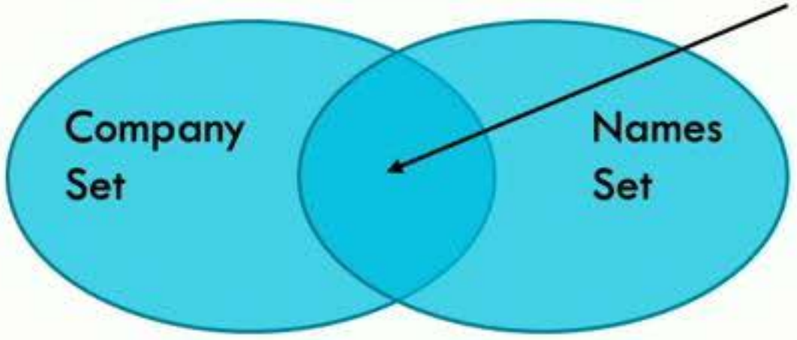
Query Table

Company	Headquarter	CEO
Google Inc.	Mountain View	Sundar Pichai
Databricks	San Francisco	Ali Ghodsi
Microsoft Corporation	Redmond	Satya Nadella
…	…	…

Query Column

Data Lake

Names	CIK	Q4 Revenue
Google Inc.	0001288776	26.06 Billions
Microsoft Corporation	0000789019	23.3 Billions
…	…	…

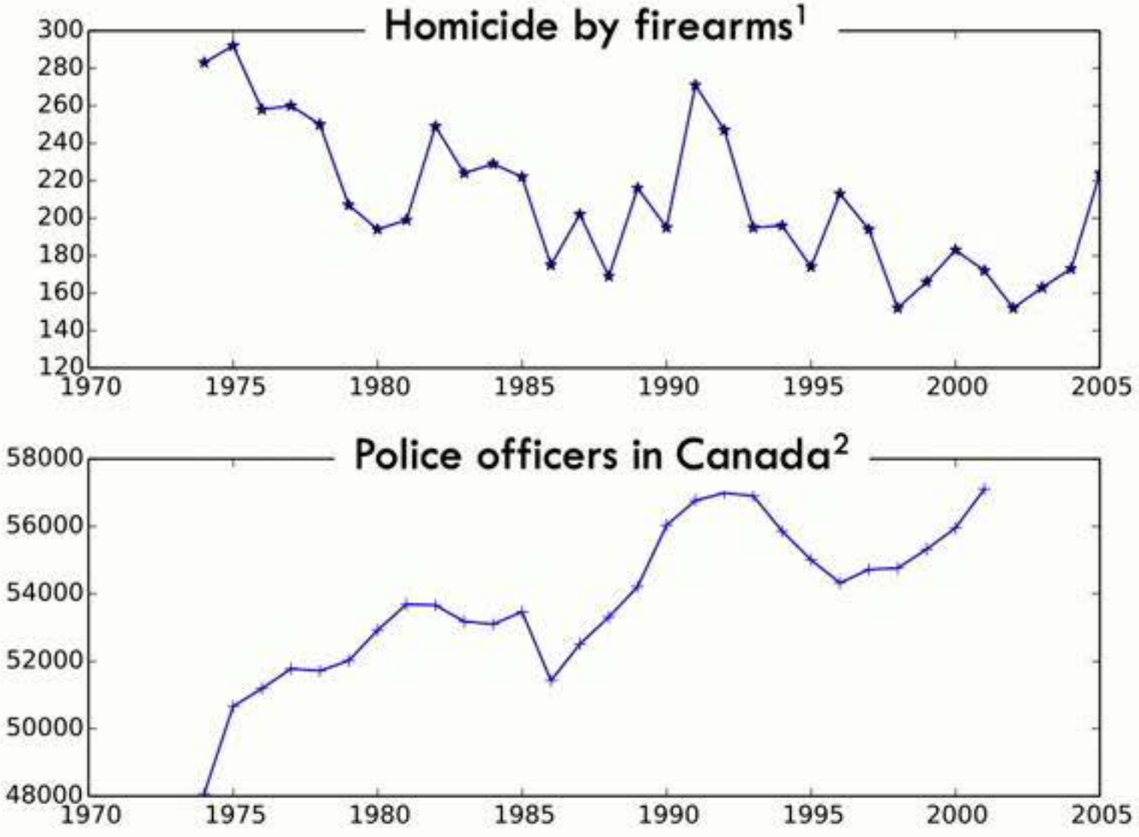


Fraction of distinct values in the query that are joinable

WHY SEARCHING FOR JOINABLE TABLES?

[1] https://open.canada.ca/data/en/dataset/be073ee2-a302-4d32-af20-a48f5fbe2e63

WHY SEARCHING FOR JOINABLE TABLES?



Discovery new/unexpected correlations

[1] https://open.canada.ca/data/en/dataset/be073ee2-a302-4d32-af20-a48f5fbe2e63

[2] https://open.canada.ca/data/en/dataset/07dcaaf5-6a58-4ce3-8876-e30feefef8dd

WHY SEARCHING FOR JOINABLE TABLES?

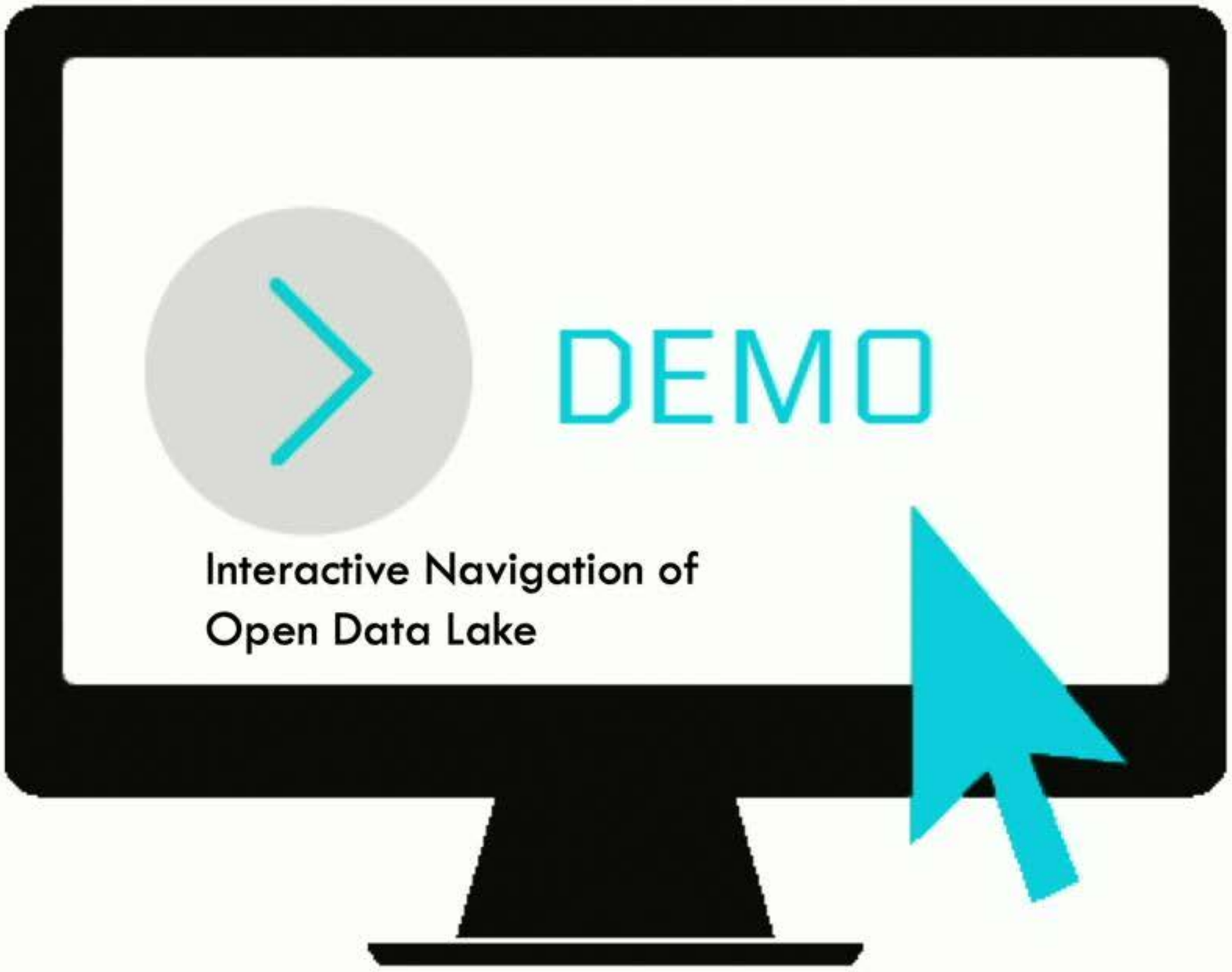
NO_x Emission	Facility	Postal	...
9430.81	Oil Sands	T9H3E3	
50.7	Carberry Factory	R0K0H0	
60.6	Grand Falls	E3Y4A5	
99	Ingleside	K0C1M0	

[1] https://open.canada.ca/data/en/dataset/f6660f68-5e78-47b3-9306-9805e1d3d6e9

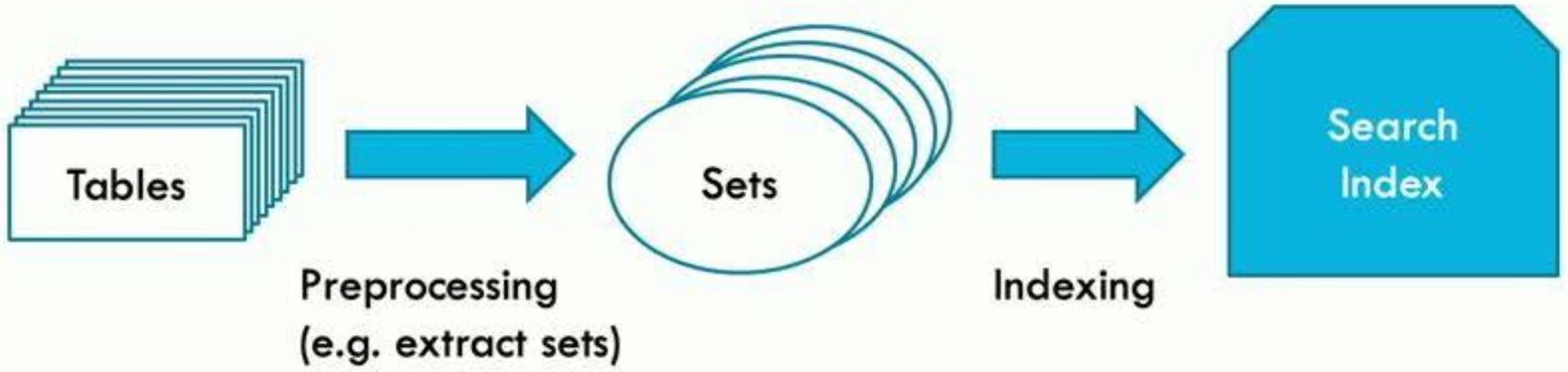
Contributor	Postal Code	Recipient Party	Amount	...
E*,B*	T9H3E3	Liberal	397.46	
J*,S*	K0K0H0	Conservative	500.00	
J*,S*	K0K0H0	Conservative	250.00	
M*,S*	K0C1M0	Liberal	400.00	

[2] https://open.canada.ca/data/en/dataset/ef1e3528-b570-4a42-92ef-18a9749af8f2

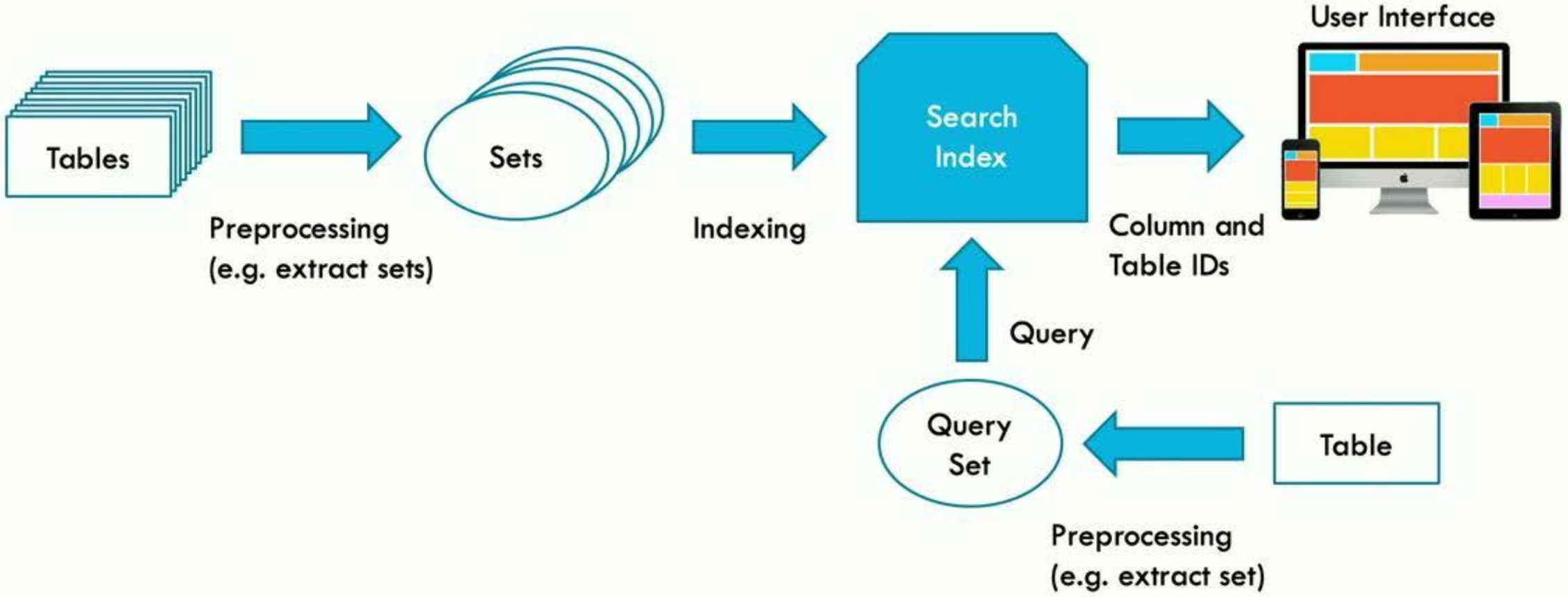
Discover new features for modeling, prediction, or insights


DEMO
Interactive Navigation of
Open Data Lake

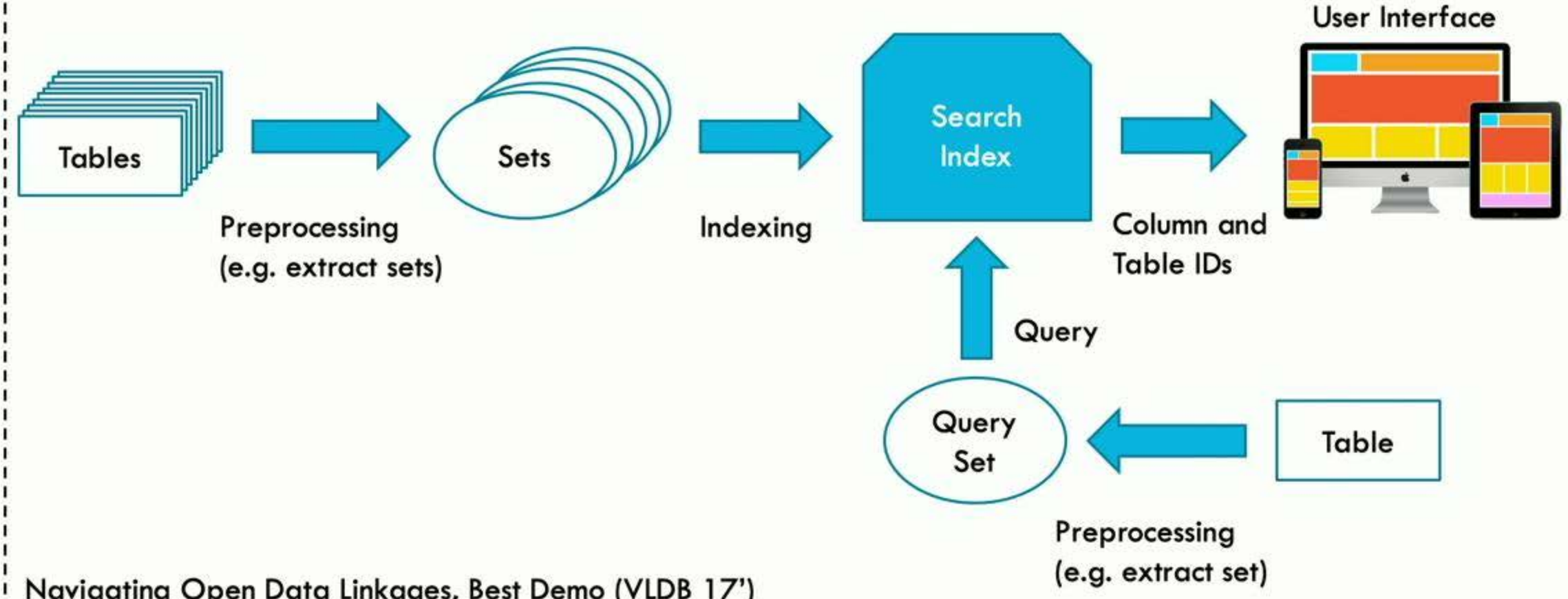
A JOINABLE TABLE SEARCH SYSTEM



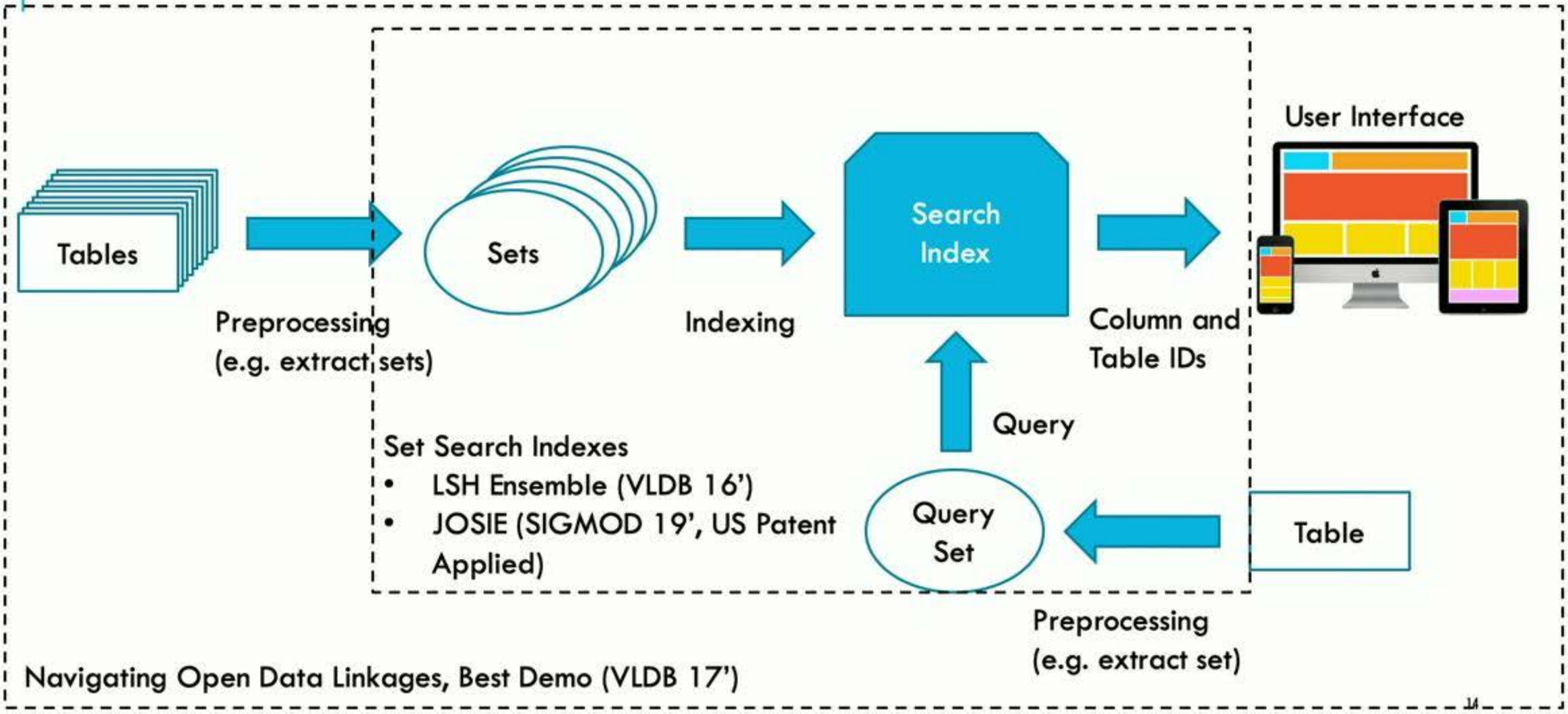
A JOINABLE TABLE SEARCH SYSTEM



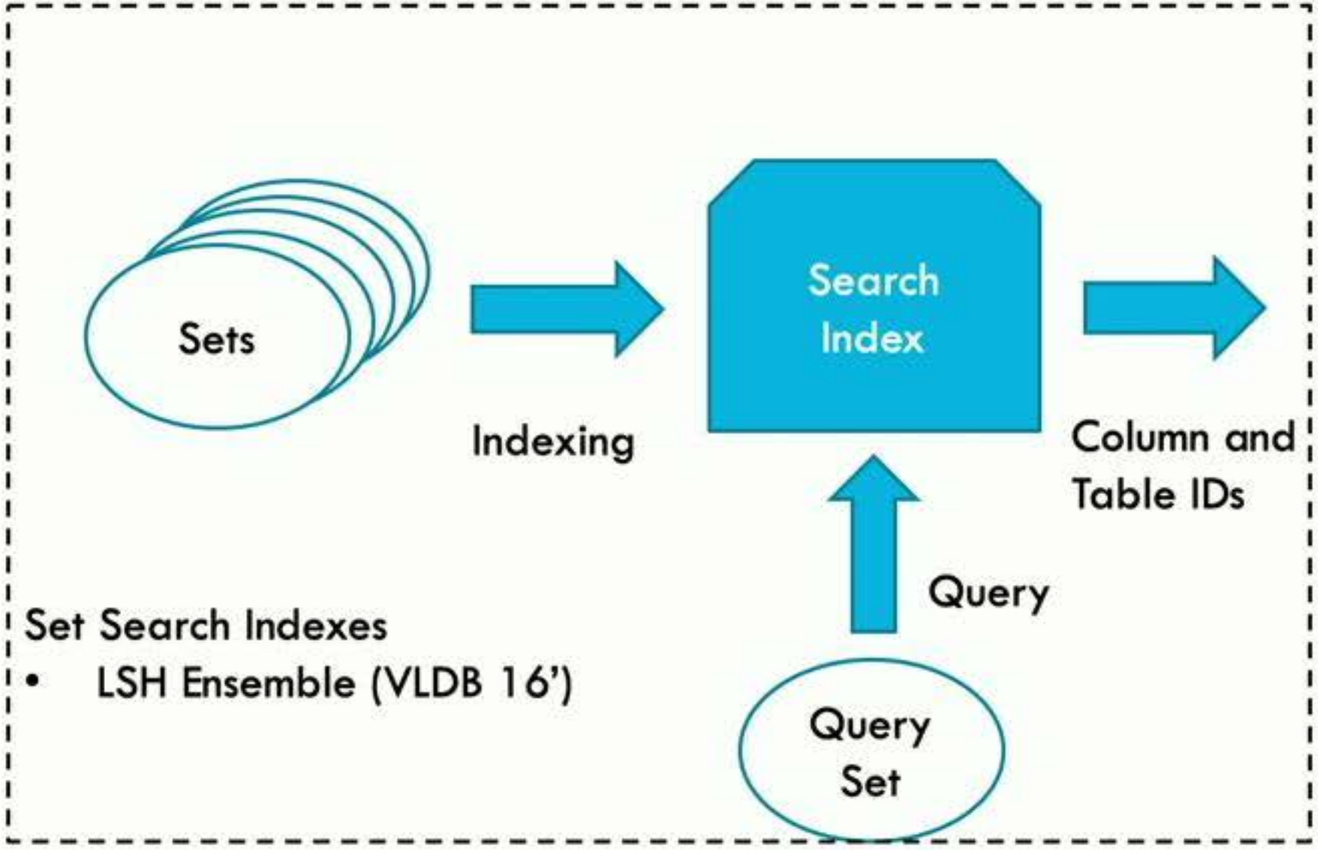
A JOINABLE TABLE SEARCH SYSTEM



A JOINABLE TABLE SEARCH SYSTEM



A JOINABLE TABLE SEARCH SYSTEM



DATA SKETCHES

Challenges:

- A set from a table column can have tens of thousands of values
- A data lake can have hundreds of thousands of tables
- An Open Data lake: 200K tables, and Max set size: 22M

Data sketches for sets:

- Small, fixed-size summaries (a few KBs)
- Probabilistic guarantee (more on this later)
- $O(|X|)$ time to build, $|X|$ is set size

We use MinHash[1] data sketches for joinable table search

[1] A. Broder. On the resemblance and containment of documents. In Compression and Complexity of Sequences, pages 21–28, 1997.

MINHASH

Random, independent hash functions (e.g., MurmurHash3)

Hash Function	Min Hash Value
h_1	
h_2	
h_3	
h_4	
h_5	
h_6	
h_7	
h_8	

h_1 {“ATL”, “LAX”, “ORD”, “DFW”, “DEN”, “JFK”, “SFO”, “LAS”, “SEA”, “CLT”}

MINHASH

Random, independent hash functions (e.g., MurmurHash3)

{“ATL”, “LAX”, “ORD”, “DFW”, “DEN”, “JFK”, “SFO”, “LAS”, “SEA”, “CHT”}

h_1

0x123, 0x3f1, 0xa1c, 0x312, 0x456, 0x42a, 0xb12, 0x98a, 0x547, 0x132

Hash Function	Min Hash Value
h_1	0x123
h_2	
h_3	
h_4	
h_5	
h_6	
h_7	
h_8	

MINHASH

Random, independent hash functions (e.g., MurmurHash3)

{“ATL”, “LAX”, “ORD”, “DFW”, “DEN”, “JFK”, “SFO”, “LAS”, “SEA”, “CLT”}

0x25b, 0xa21, 0x81a, 0x514, 0xb51, 0x431, 0x14a, 0x981, 0x1c7, 0x16b

Hash Function	Min Hash Value
h_1	0x123
h_2	0x14a
h_3	
h_4	
h_5	
h_6	
h_7	
h_8	

MINHASH

Random, independent hash functions (e.g., MurmurHash3)

Hash Function	Min Hash Value
h_1	0x123
h_2	0x14a
h_3	0x41c
h_4	0x1bc
h_5	0x2c3
h_6	0x5d1
h_7	0x613
h_8	0x312

{“ATL”, “LAX”, “ORD”, “DFW”, “DEN”, “JFK”, “SFO”, “LAS”, “SEA”, “CLT”}

[Optional]: speed up using only one hash function and universal hashing

MINHASH

For two sets X and Y, the probability of a minimum hash value collision:

$$P\left[\min\left(h_1(X)\right) = \min\left(h_1(Y)\right)\right] = \frac{|X \cap Y|}{|X \cup Y|}$$

← This is Jaccard similarity

X

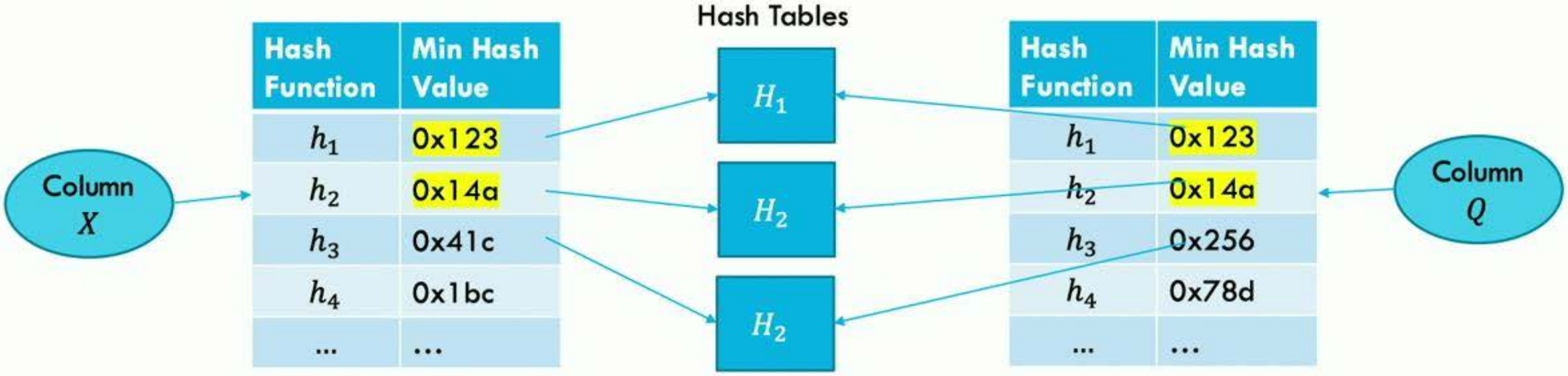
Hash Function	Min Hash Value
h_1	0x123
h_2	0x14a
h_3	0x41c
h_4	0x1bc
…	…

Y

Hash Function	Min Hash Value
h_1	0x123
h_2	0x14a
h_3	0x256
h_4	0x78d
…	…

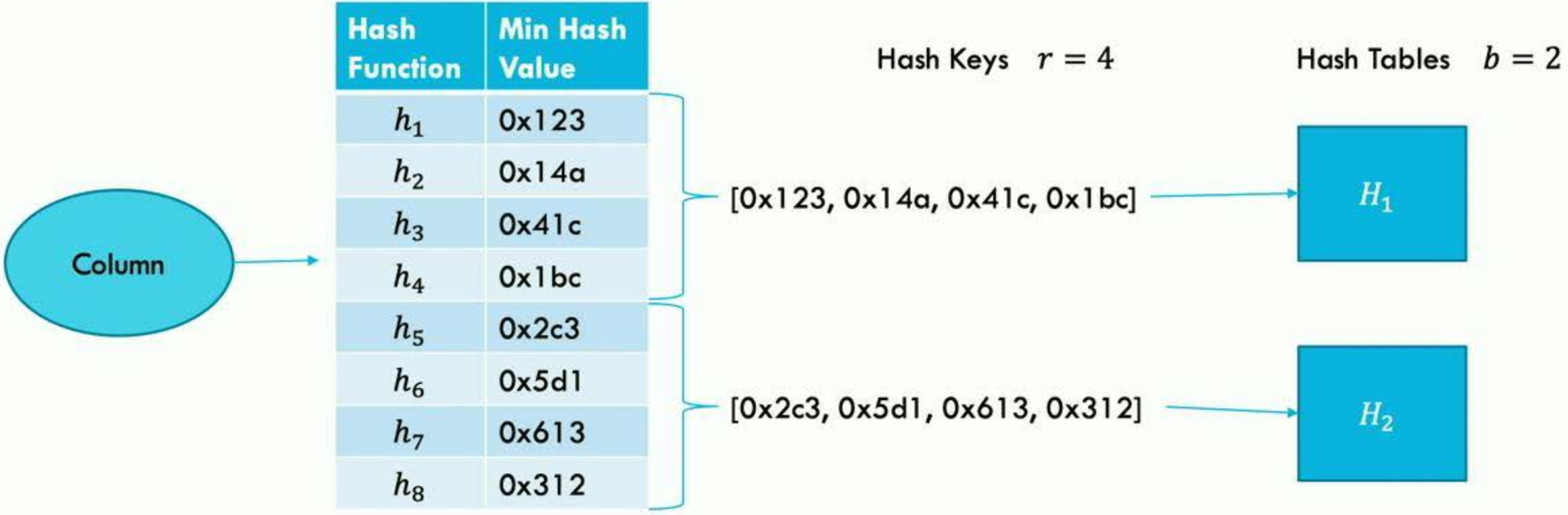
MINHASH

We can use hash tables to find "approximately" joinable columns under Jaccard similarity



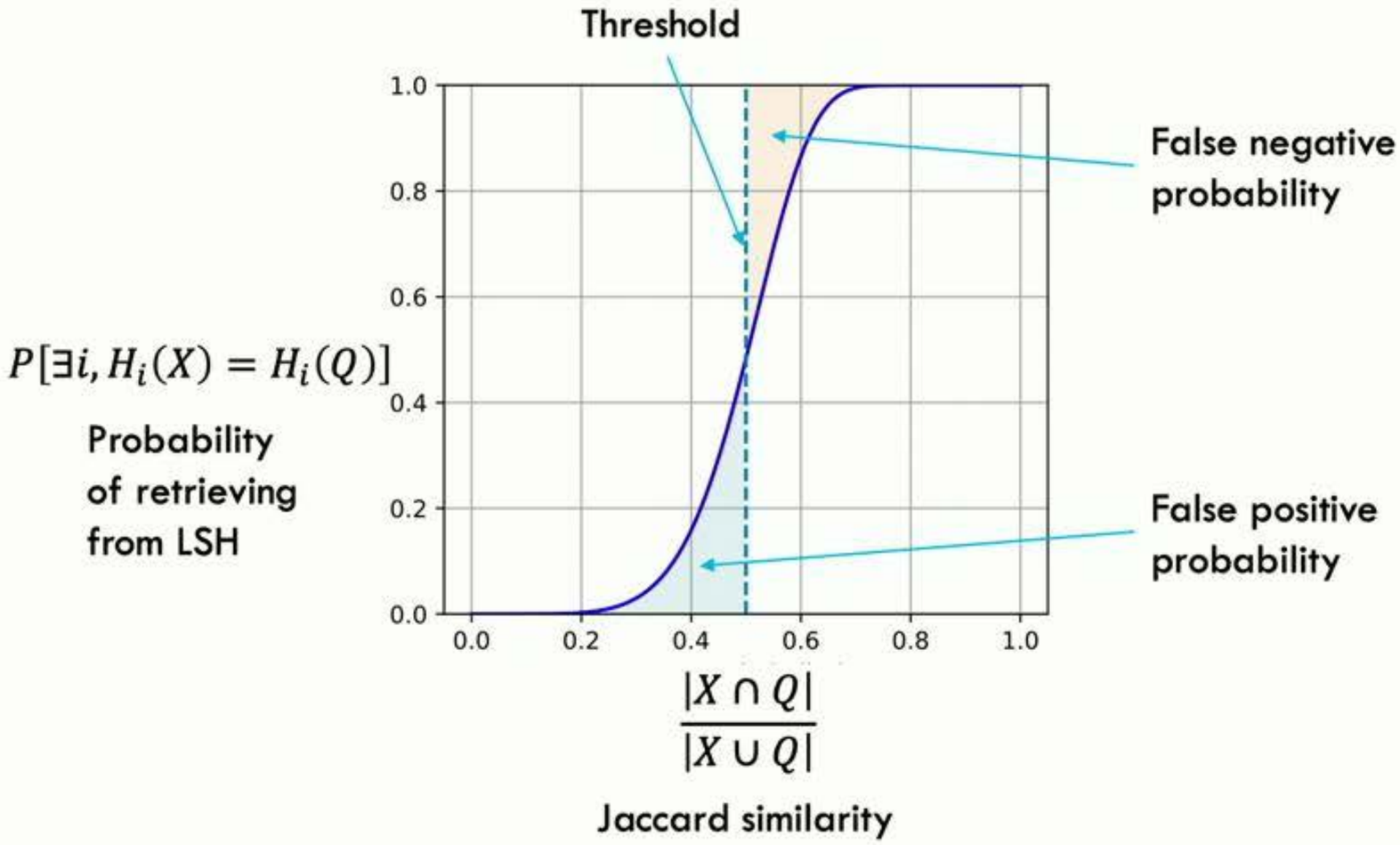
MINHASH LSH

Locality Sensitive Hashing[1] (LSH) index provides better accuracy through "boosting"

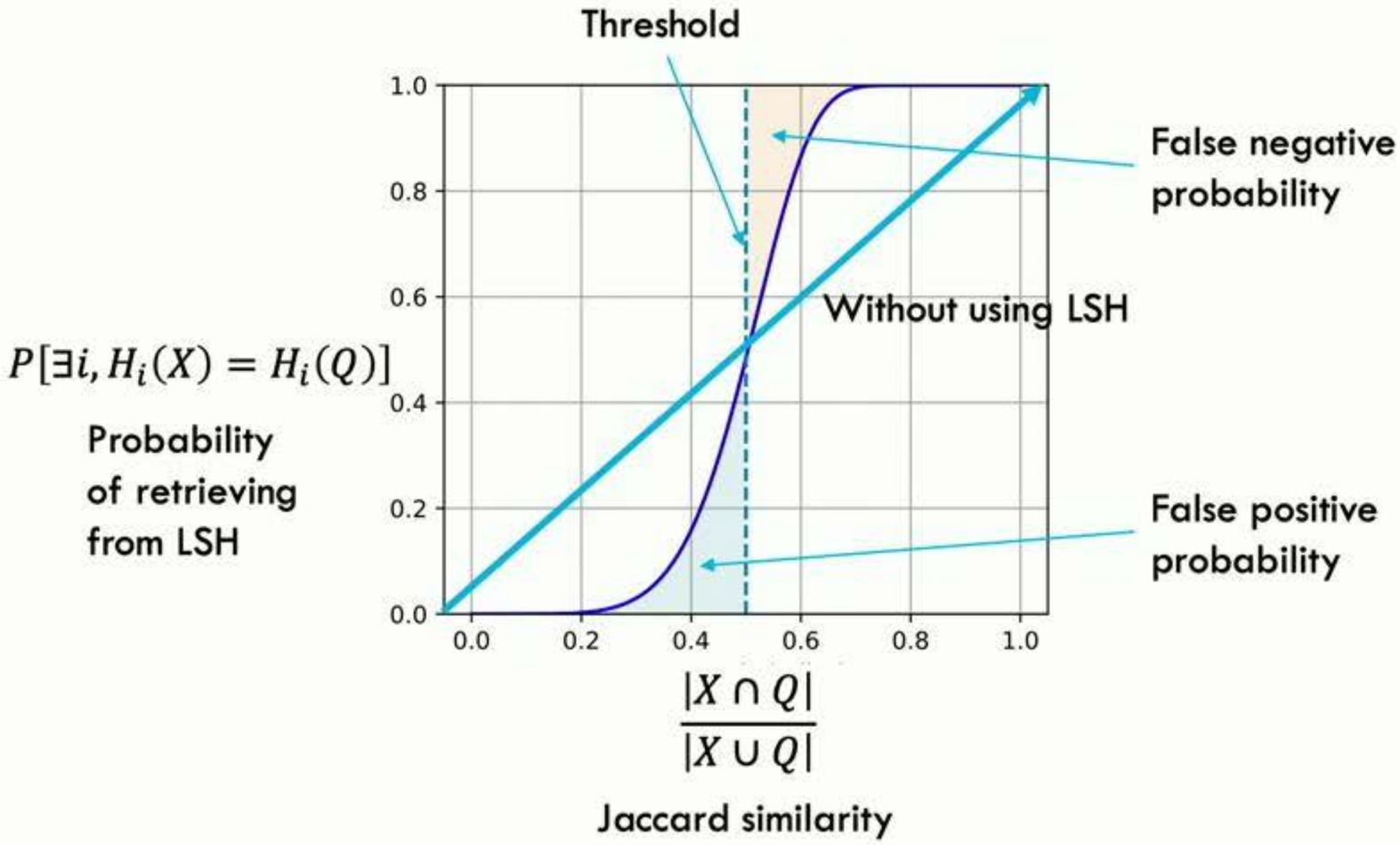


[1] Aristides Gionis, Piotr Indyk, Rajeev Motwani, , "Similarity Search in High Dimensions via Hashing", VLDB 1999

MINHASH LSH

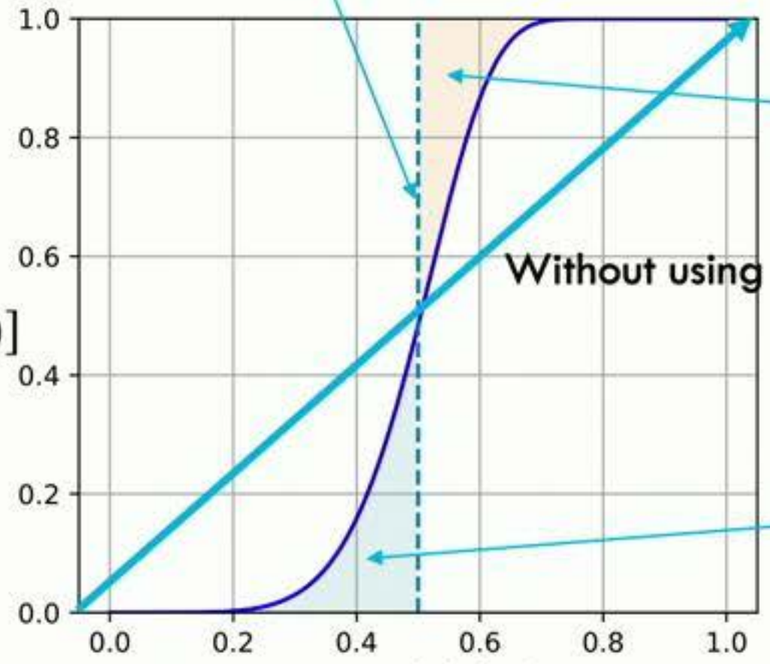

Threshold
False negative probability
False positive probability
$P[\exists i, H_i(X) = H_i(Q)]$
Probability of retrieving from LSH
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
$\frac{|X \cap Q|}{|X \cup Q|}$
Jaccard similarity

MINHASH LSH


Threshold
False negative probability
Without using LSH
False positive probability
$P[\exists i, H_i(X) = H_i(Q)]$
Probability of retrieving from LSH
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
$\frac{|X \cap Q|}{|X \cup Q|}$
Jaccard similarity

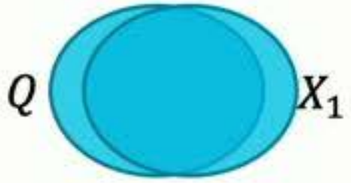
MINHASH LSH

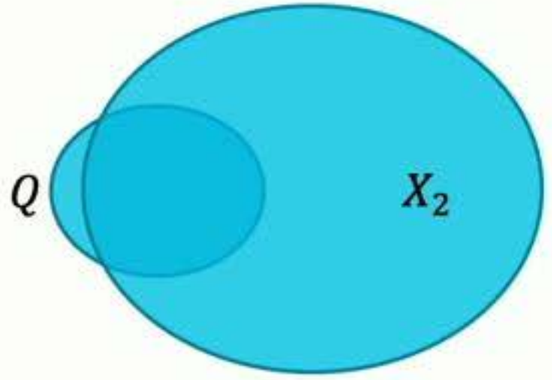
Minimize error probability with respect to the Jaccard similarity threshold, by tuning.



IS JACCARD SIMILARITY THE RIGHT MEASURE?

As $|X_1 \cap Q| = |X_2 \cap Q|$, both X_1 and X_2 are equally good for joining with Q

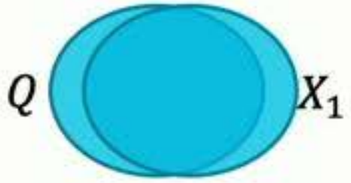


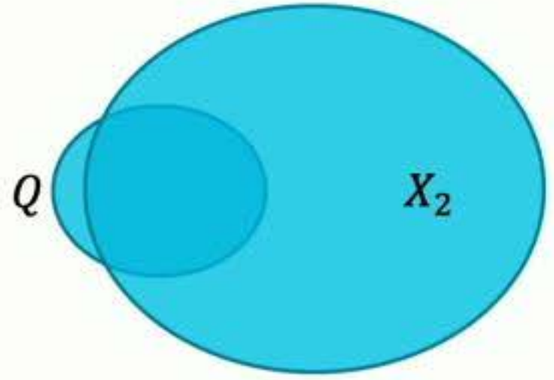


IS JACCARD SIMILARITY THE RIGHT MEASURE?

As $|X_1 \cap Q| = |X_2 \cap Q|$, both X_1 and X_2 are equally good for joining with Q

But $\frac{|X_1 \cap Q|}{|X_1 \cup Q|} > \frac{|X_2 \cap Q|}{|X_2 \cup Q|}$





Alternatively, $\frac{|X_1 \cap Q|}{|Q|} = \frac{|X_2 \cap Q|}{|Q|}$

CONTAINMENT SEARCH

We use containment, $\frac{|X \cap Q|}{|Q|}$, as a measure for joinable tables

We can “fix” MinHash LSH for containment

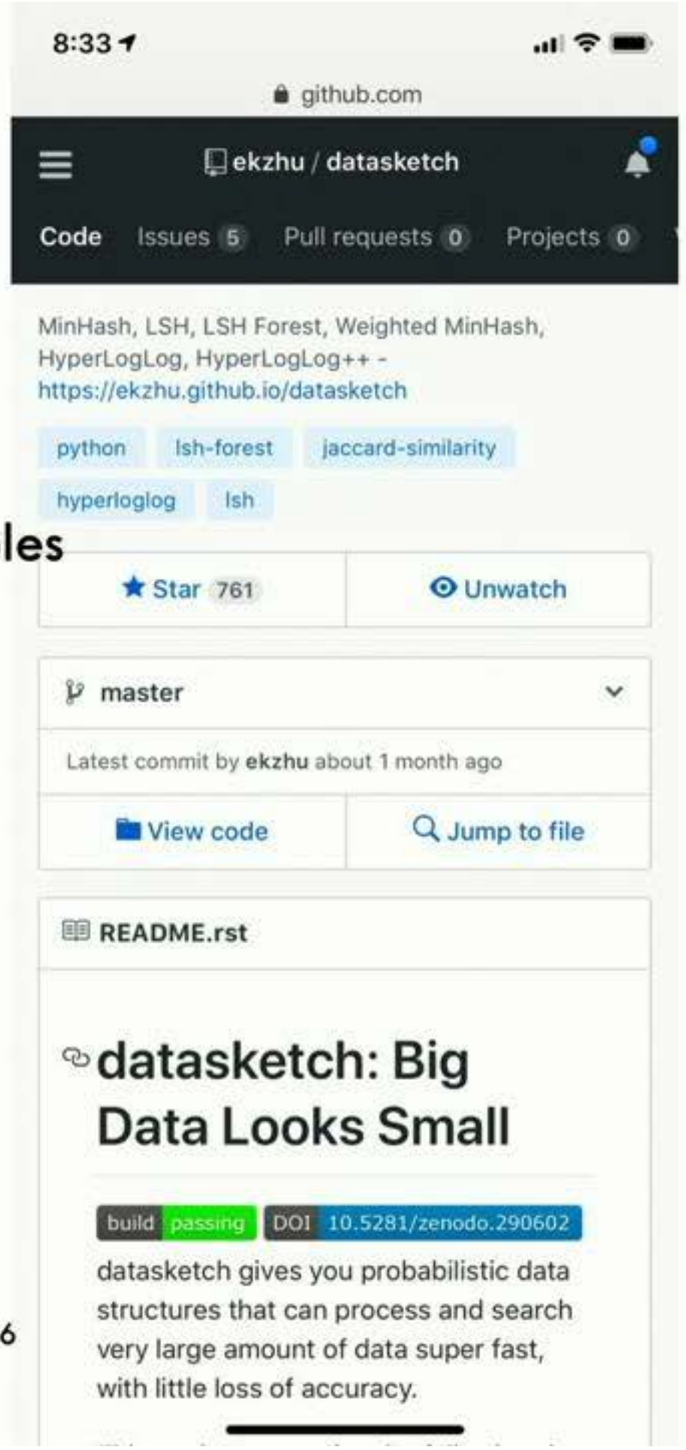
CONTAINMENT SEARCH

We use containment, $\frac{|X \cap Q|}{|Q|}$, as a measure for joinable tables

We can "fix" MinHash LSH for containment

LSH Ensemble[1]

- Part of *datasketch* Python library[2], together with MinHash LSH
- The library is used by Google (TimeSketch), MIT (Aurum Data Discovery) and Stanford (NLP)
- Over 760 stars on Github



[1] Erkang Zhu, Fatemeh Nargesian, Ken Pu, Renée J. Miller, "LSH Ensemble:L Internet-Scale Domain Search", VLDB 2016
[2] https://github.com/ekzhu/datasketch

FROM CONTAINMENT TO JACCARD

Apply Inclusion-Exclusion Principle:

$$\frac{|X \cap Q|}{|X \cup Q|} = \frac{|X \cap Q|}{|X| + |Q| - |X \cap Q|} = \frac{\frac{|X \cap Q|}{|Q|}}{1 + \frac{|X|}{|Q|} - \frac{|X \cap Q|}{|Q|}}$$

Jaccard

Containment

FROM CONTAINMENT TO JACCARD

A containment threshold (t^*) can be translated into a Jaccard threshold (s^*)

$$s^*_{|X|} = \frac{t^*}{1 + \frac{|X|}{|Q|} - t^*}$$

FROM CONTAINMENT TO JACCARD

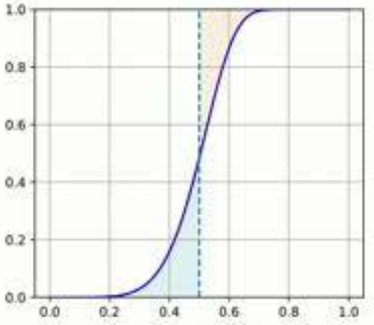
A containment threshold (t^*) can be translated into a Jaccard threshold (s^*)

User-defined containment threshold t^*

$$s^*_{|X|} = \frac{t^*}{1 + \frac{|X|}{|Q|} - t^*}$$

Internal Jaccard threshold $s^*_{|X|}$

Tune MinHash LSH index using $s^*_{|X|}$



FROM CONTAINMENT TO JACCARD

A containment threshold (t^*) can be translated into a Jaccard threshold (s^*)

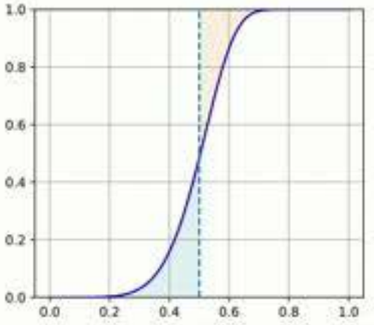
User-defined containment threshold t^*

$$s^*_{|X|} = \frac{t^*}{1 + \frac{|X|}{|Q|} - t^*}$$

Internal Jaccard threshold $s^*_{|X|}$

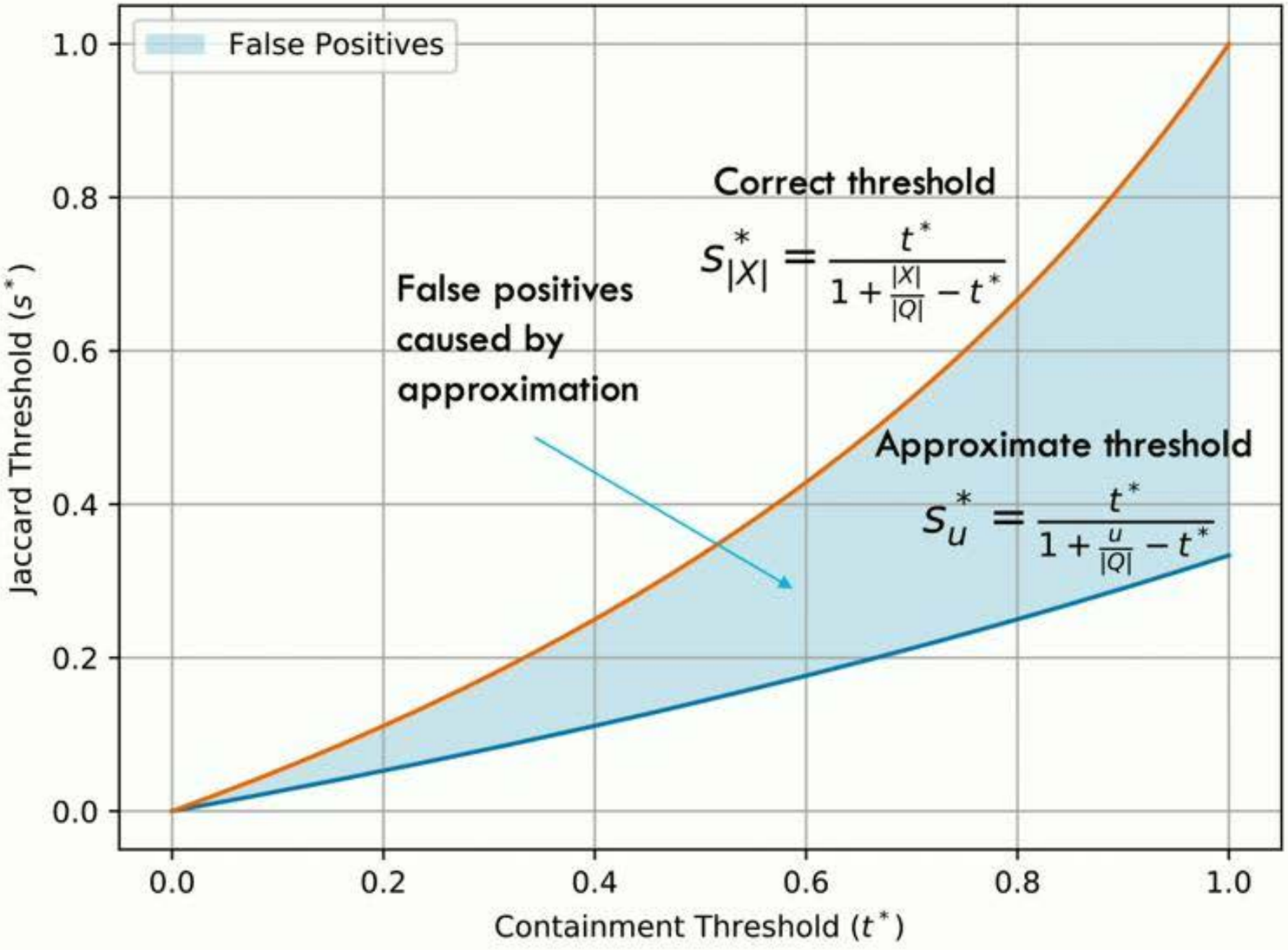
$|X|$ is not a constant!

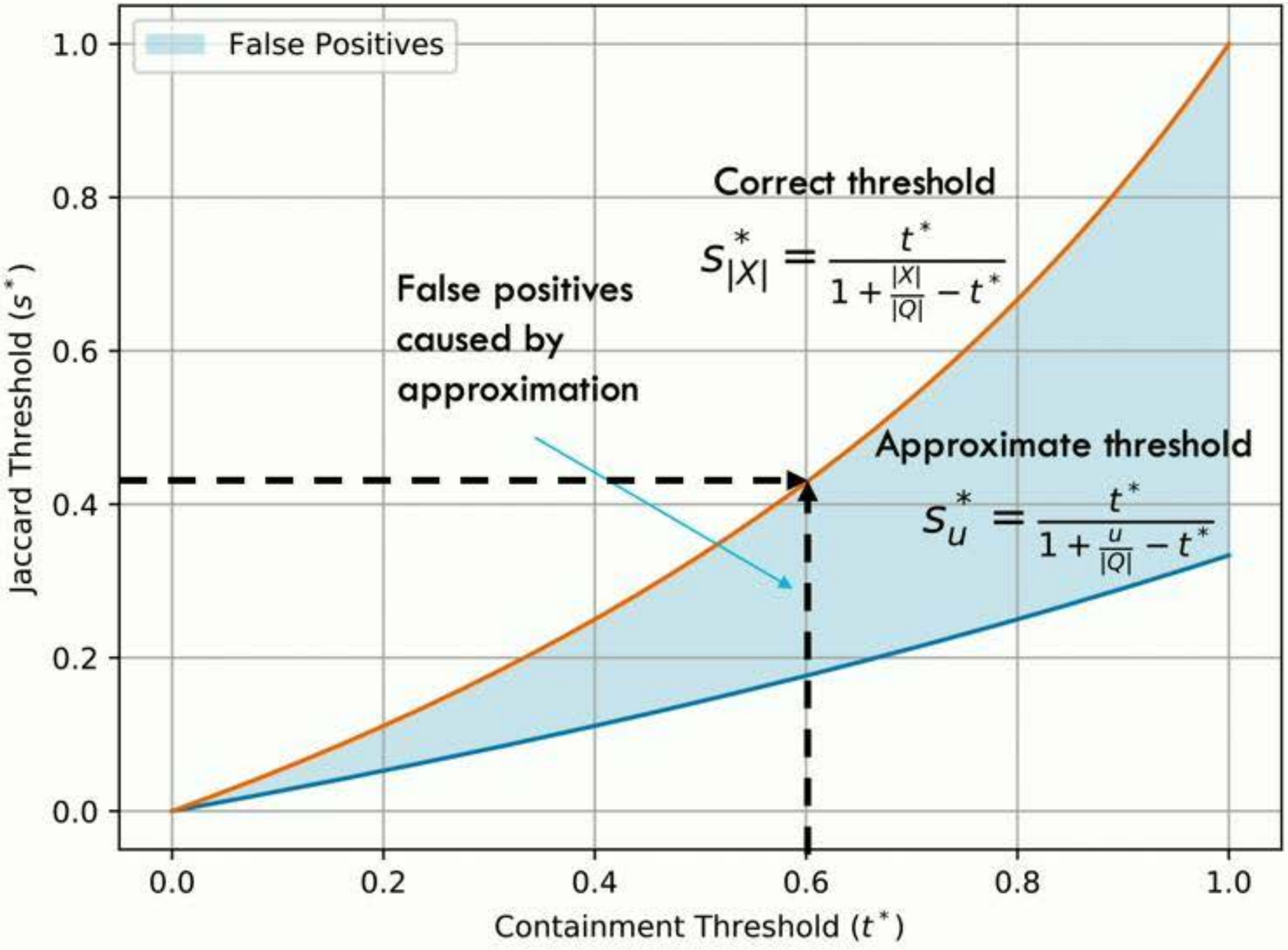
Tune MinHash LSH index using $s^*_{|X|}$

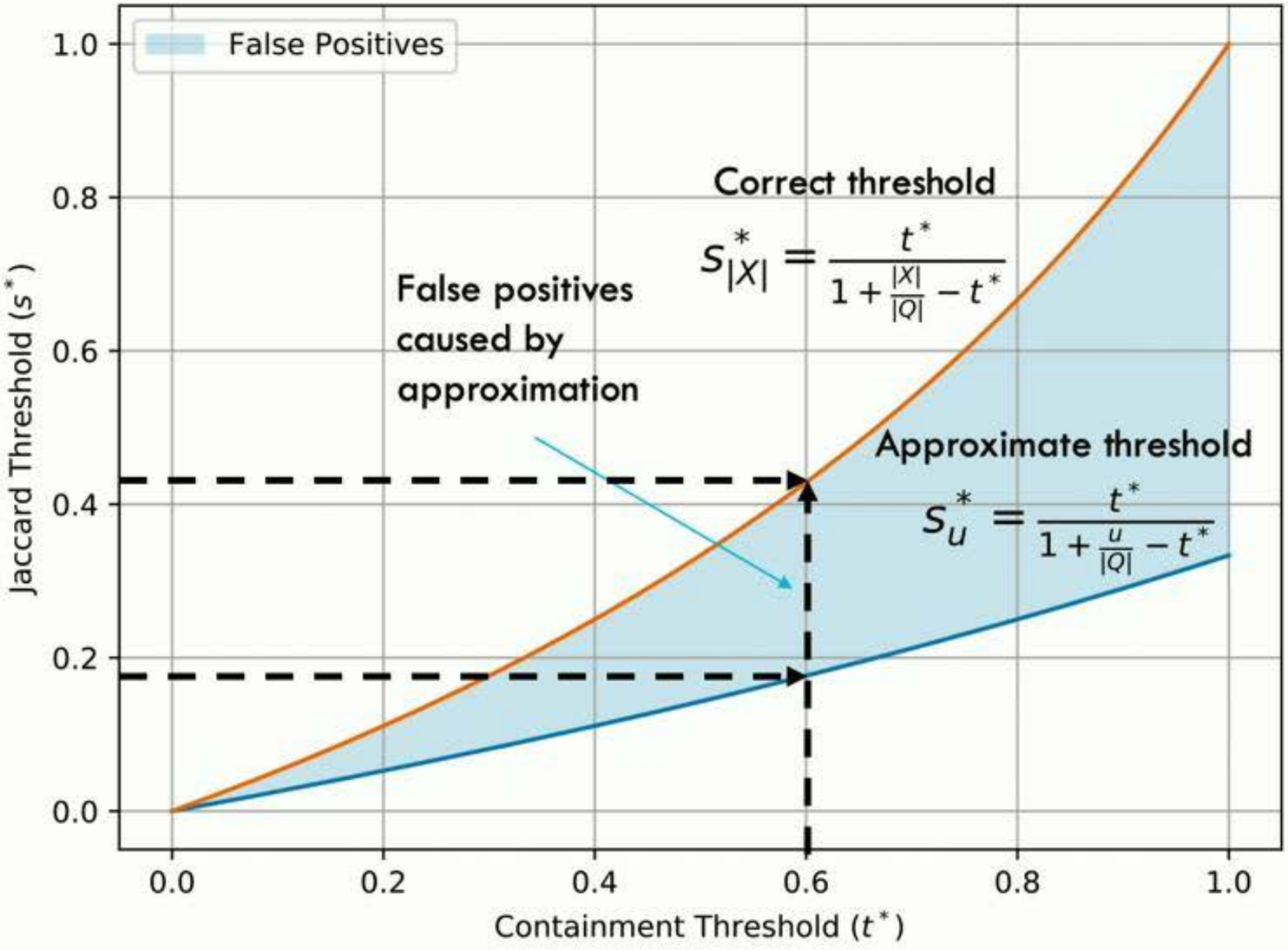


Alternative: use u (the largest set size) to approximate $|X|$

$$s^*_u = \frac{t^*}{1 + \frac{u}{|Q|} - t^*}$$


False Positives
Correct threshold
$s^*_{|X|} = \frac{t^*}{1 + \frac{|X|}{|Q|} - t^*}$
False positives caused by approximation
Approximate threshold
$s^*_u = \frac{t^*}{1 + \frac{u}{|Q|} - t^*}$
Jaccard Threshold (s^*)
Containment Threshold (t^*)
0.0
0.2
0.4
0.6
0.8
1.0


False Positives
Correct threshold
$s^*_{|X|} = \frac{t^*}{1 + \frac{|X|}{|Q|} - t^*}$
False positives caused by approximation
Approximate threshold
$s^*_u = \frac{t^*}{1 + \frac{u}{|Q|} - t^*}$
Jaccard Threshold (s^*)
Containment Threshold (t^*)
0.0
0.2
0.4
0.6
0.8
1.0


False Positives
Correct threshold
$s^*_{|X|} = \frac{t^*}{1 + \frac{|X|}{|Q|} - t^*}$
False positives caused by approximation
Approximate threshold
$s^*_u = \frac{t^*}{1 + \frac{u}{|Q|} - t^*}$
Jaccard Threshold (s^*)
Containment Threshold (t^*)
0.0
0.2
0.4
0.6
0.8
1.0

REDUCE FALSE POSITIVES – BRUTE FORCE

We can verify the exact containment of all sets in the output – “posting processing”

REDUCE FALSE POSITIVES – BRUTE FORCE

We can verify the exact containment of all sets in the output – "posting processing"

https://www.theguardian.com/lifeandstyle/2015/dec/02/eat-drink-bad-for-us-evolution

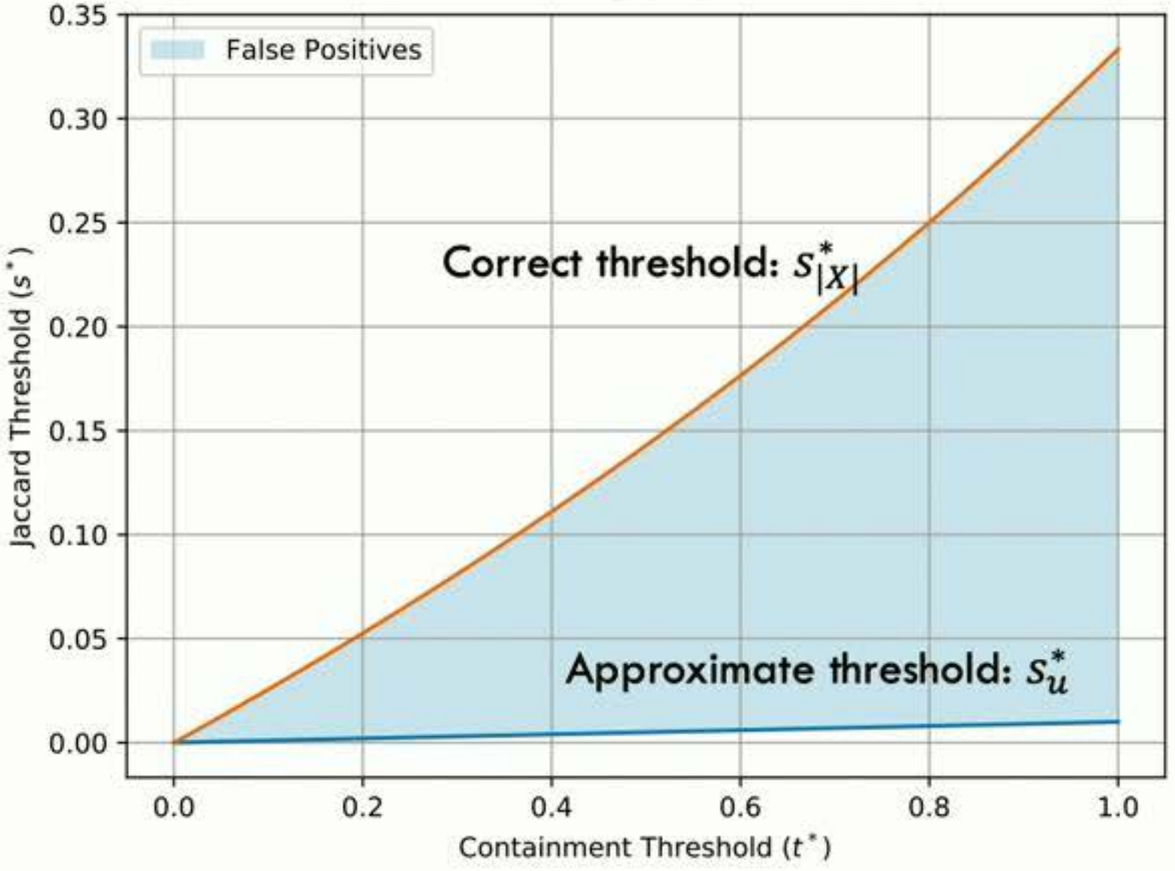
Set sizes in the index:

{3, 3, 3, 3, 4, 4, 4, 4, 99, 99, 99, 100, 100, 100}

Set sizes in the index:

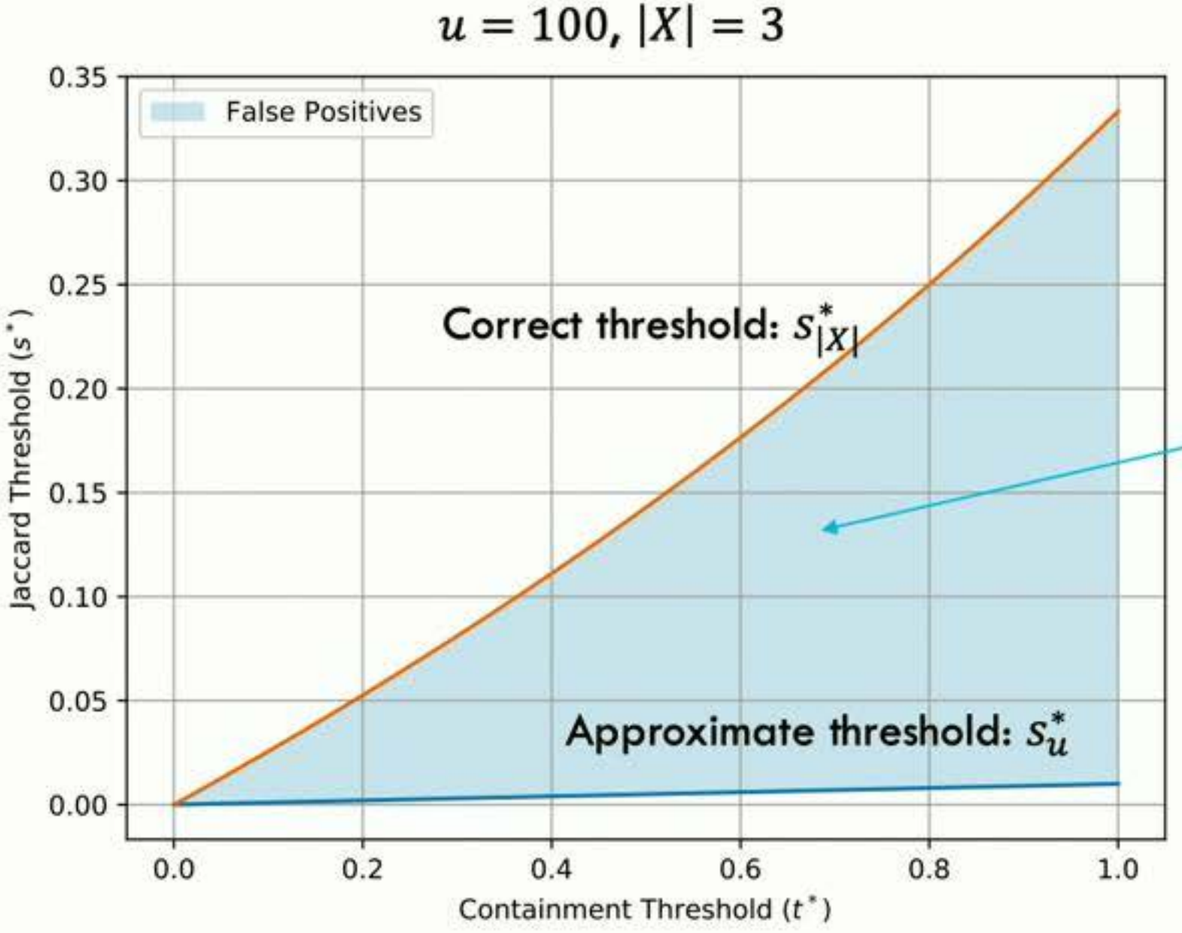
{3, 3, 3, 3, 4, 4, 4, 4, 99, 99, 99, 100, 100, 100}

$u = 100, |X| = 3$



Set sizes in the index:

{3, 3, 3, 3, 4, 4, 4, 4, 99, 99, 99, 100, 100, 100}



Too many false positives to remove by post-processing

Use two indexes:

Set sizes in index 1

{3, 3, 3, 3, 4, 4, 4, 4}

Set sizes in index 2

{99, 99, 99, 100, 100, 100}

Use two indexes:

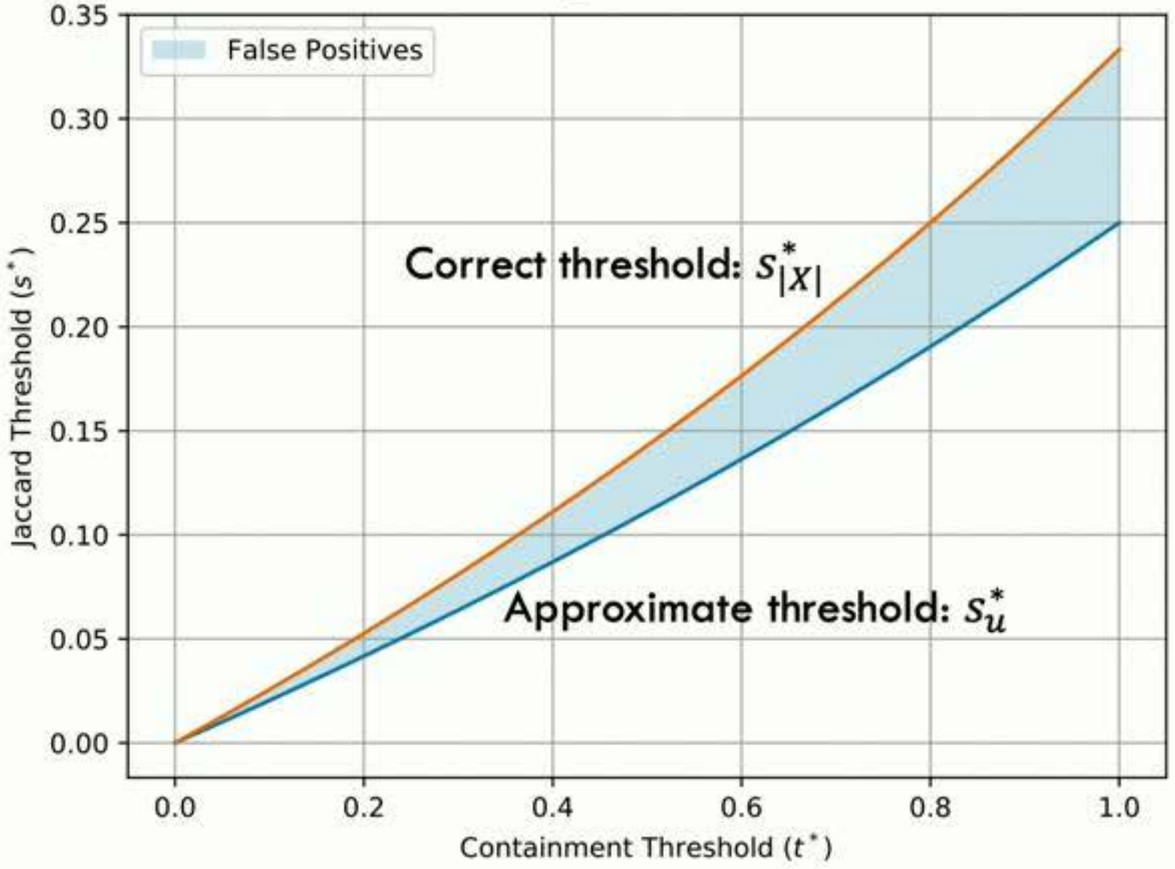
Set sizes in index 1

{3, 3, 3, 3, 4, 4, 4, 4}

Set sizes in index 2

{99, 99, 99, 100, 100, 100}

$u_1 = 4,\ |X| = 3$

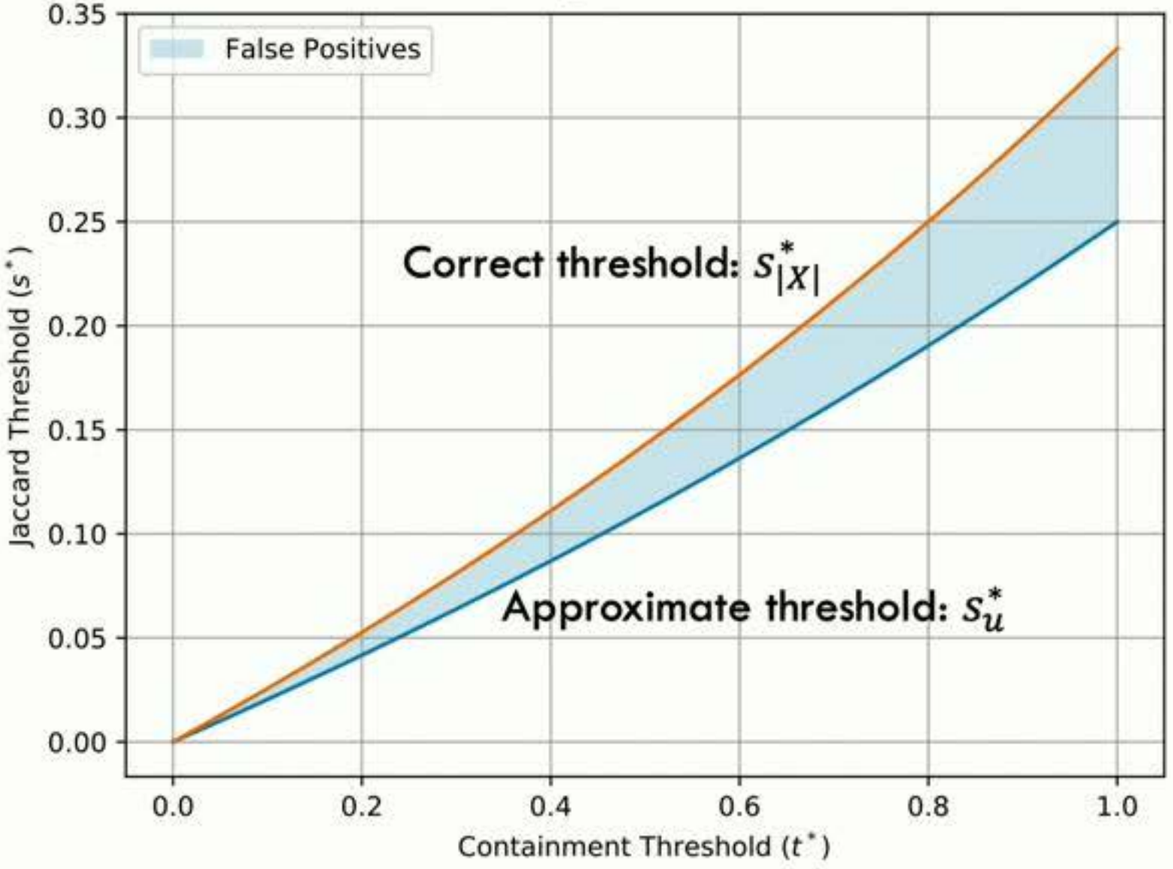


Use two indexes:

Set sizes in index 1

{3, 3, 3, 3, 4, 4, 4, 4}

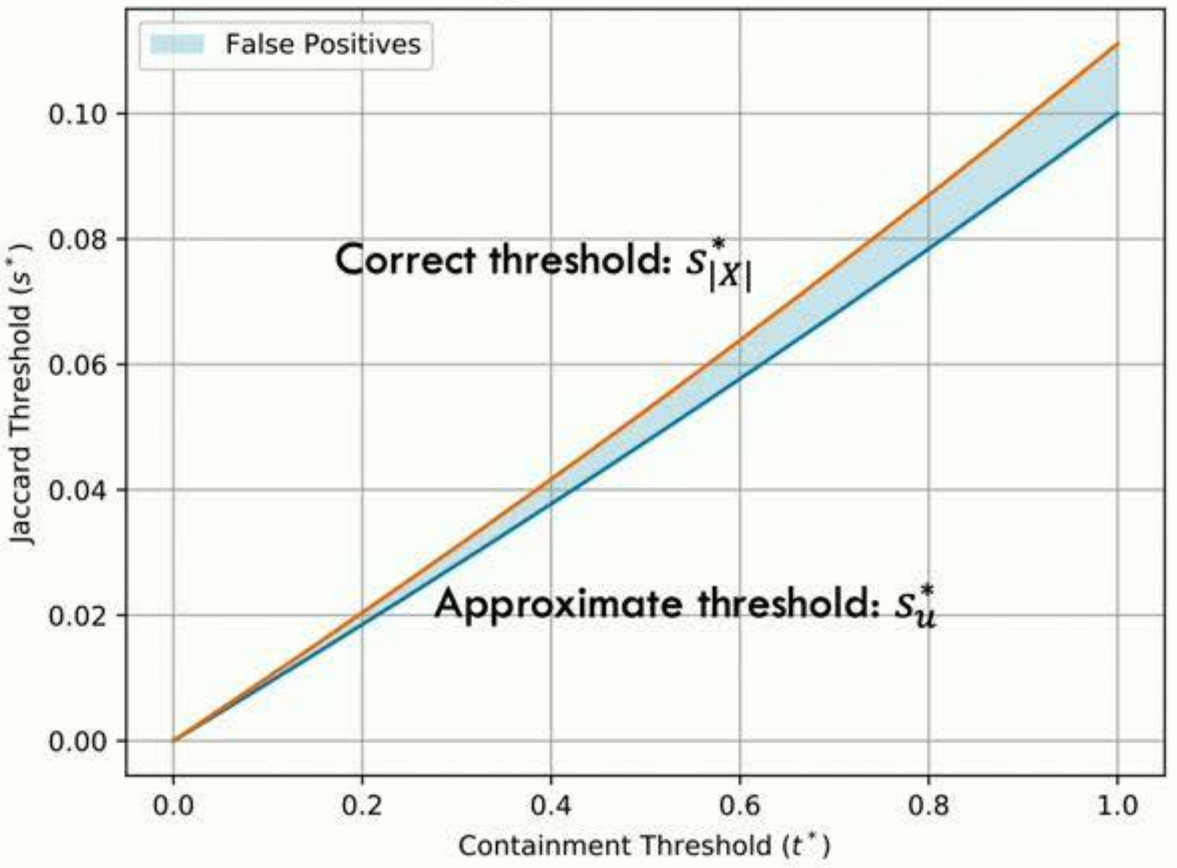
$u_1 = 4, |X| = 3$

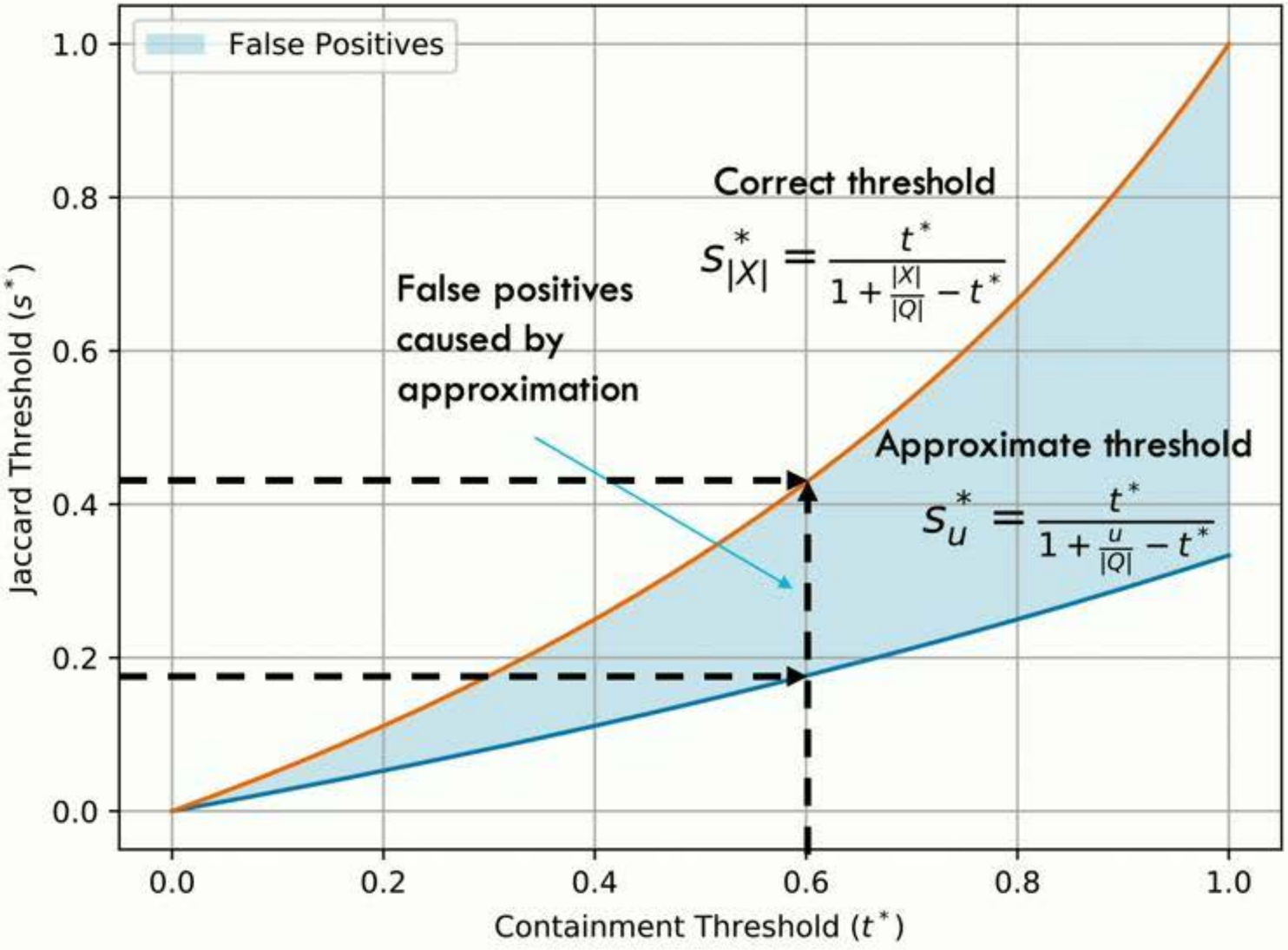


Set sizes in index 2

{99, 99, 99, 100, 100, 100}

$u_2 = 100, |X| = 99$




False Positives
Correct threshold
$s^*_{|X|} = \frac{t^*}{1 + \frac{|X|}{|Q|} - t^*}$
False positives caused by approximation
Approximate threshold
$s^*_u = \frac{t^*}{1 + \frac{u}{|Q|} - t^*}$
Jaccard Threshold (s^*)
Containment Threshold (t^*)
0.0
0.2
0.4
0.6
0.8
1.0

REDUCE FALSE POSITIVES – BRUTE FORCE

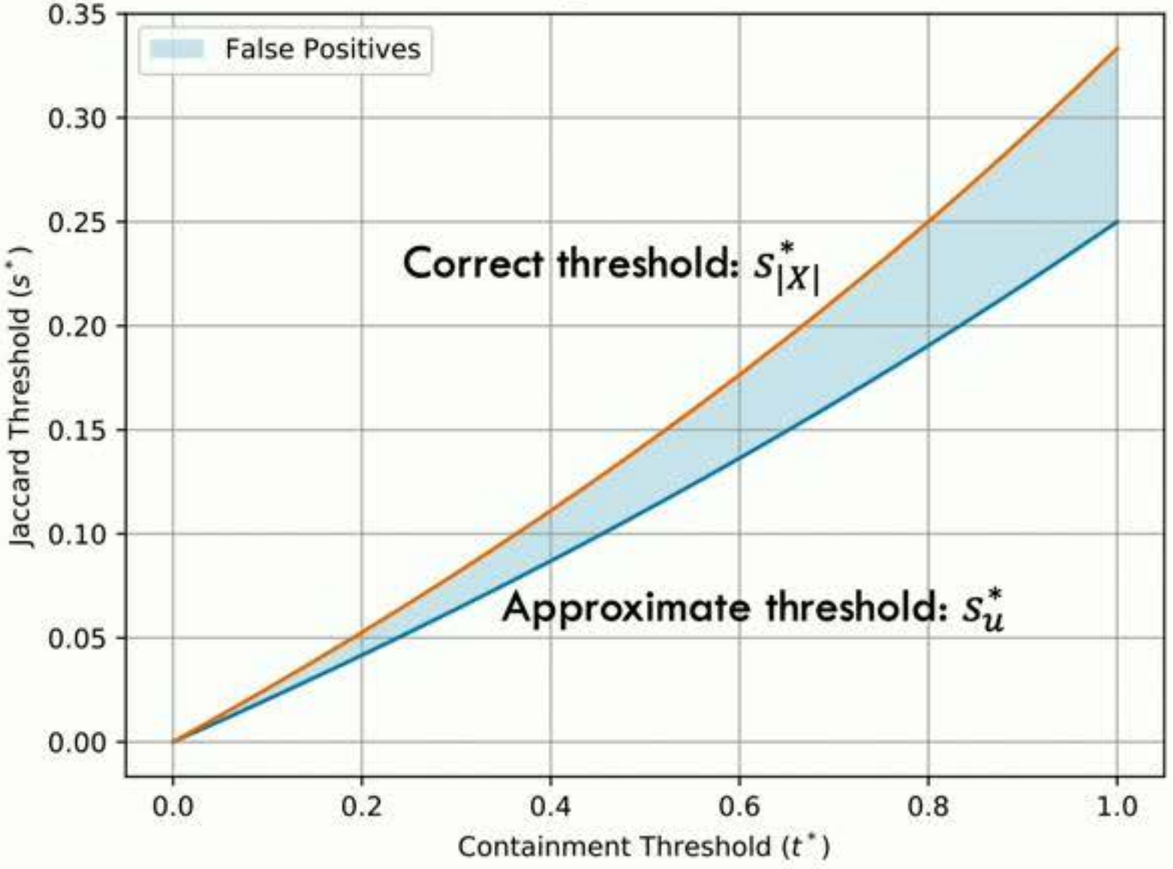
We can verify the exact containment of all sets in the output – “posting processing”

Use two indexes:

Set sizes in index 1

{3, 3, 3, 3, 4, 4, 4, 4}

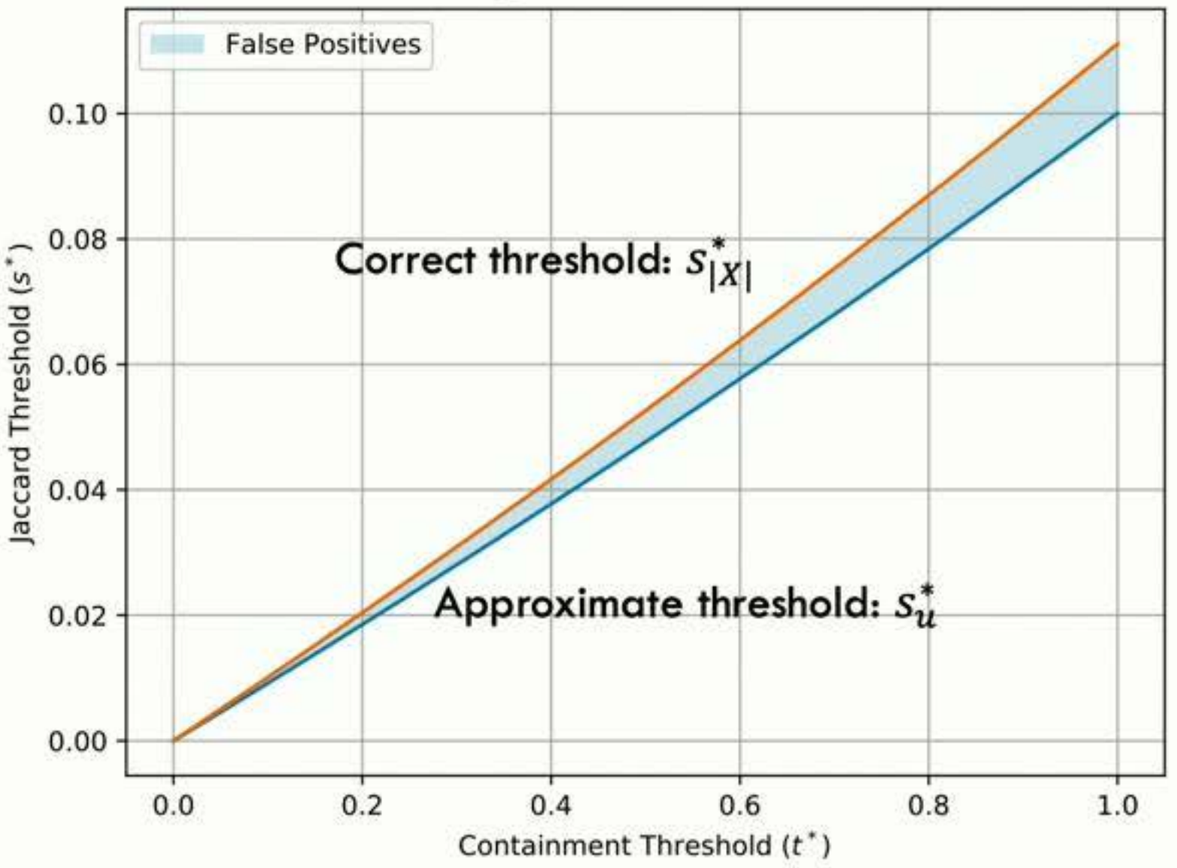
$u_1 = 4, |X| = 3$



Set sizes in index 2

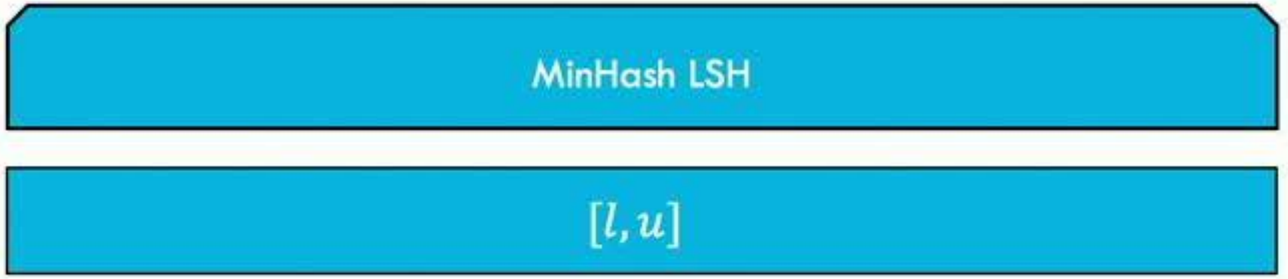
{99, 99, 99, 100, 100, 100}

$u_2 = 100, |X| = 99$



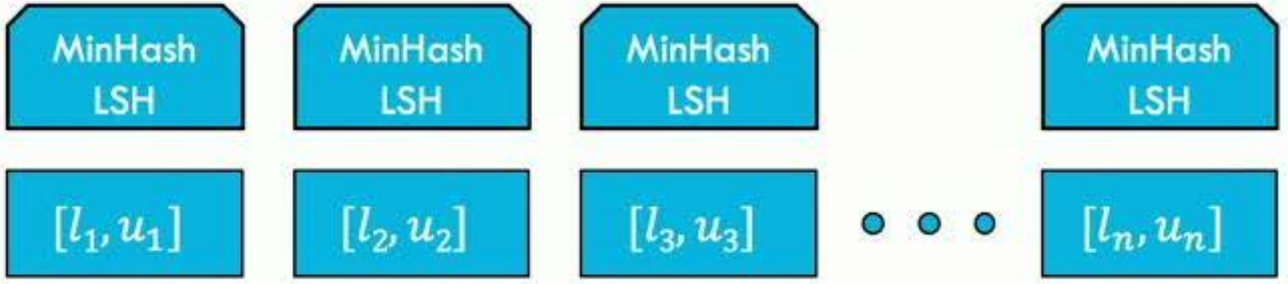
REDUCE FALSE POSITIVES - PARTITIONS

Partition the sets into n disjoint partitions with increasing set sizes:



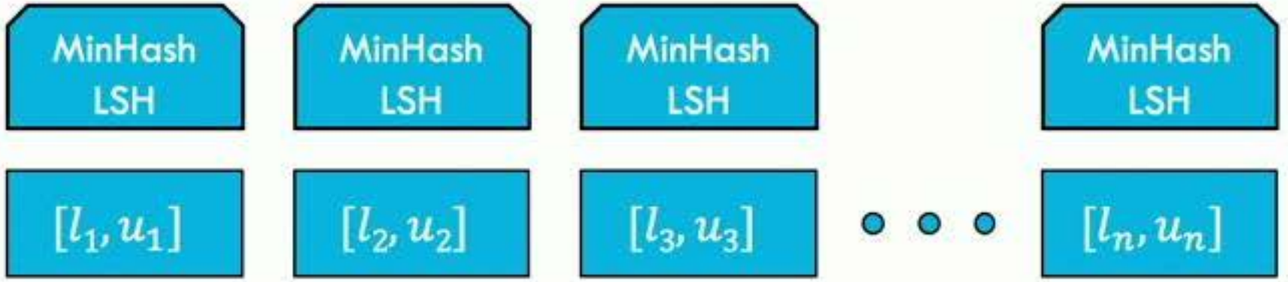
REDUCE FALSE POSITIVES - PARTITIONS

Partition the sets into n disjoint partitions with increasing set sizes:



REDUCE FALSE POSITIVES - PARTITIONS

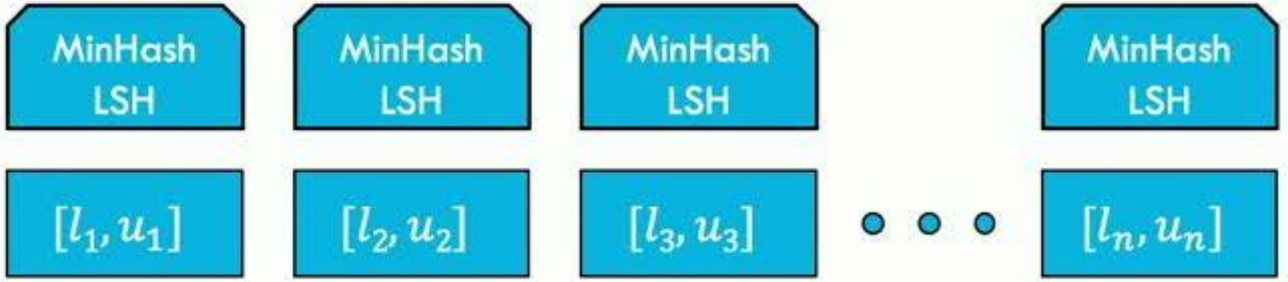
Partition the sets into n disjoint partitions with increasing set sizes:



Reduce false positive at the rate of $\frac{1}{(u-u_i)^2}$ by using u_i instead of u for each partition!

REDUCE FALSE POSITIVES - PARTITIONS

Partition the sets into n disjoint partitions with increasing set sizes:



Reduce false positive at the rate of $\frac{1}{(u-u_i)^2}$ by using u_i instead of u for each partition!

Infinite partitions? Query time grows linearly with n!

UPPER BOUND NUM. FALSE POSITIVES

Upper bound of number of false positives over a set size interval $[l, u]$

$$N_{l,u}^{FP} \leq \sum_{x \in [l,u]} N_x \cdot \left(1 - \frac{x}{u}\right)$$

Number of sets with size x

$[l, u]$

OPTIMAL PARTITIONING – UNIFORM DIST.

Example: $u = 100$, $l = 1$, $N_{l,u} = 100$, $n = 2$, assume uniform distribution

OPTIMAL PARTITIONING – UNIFORM DIST.

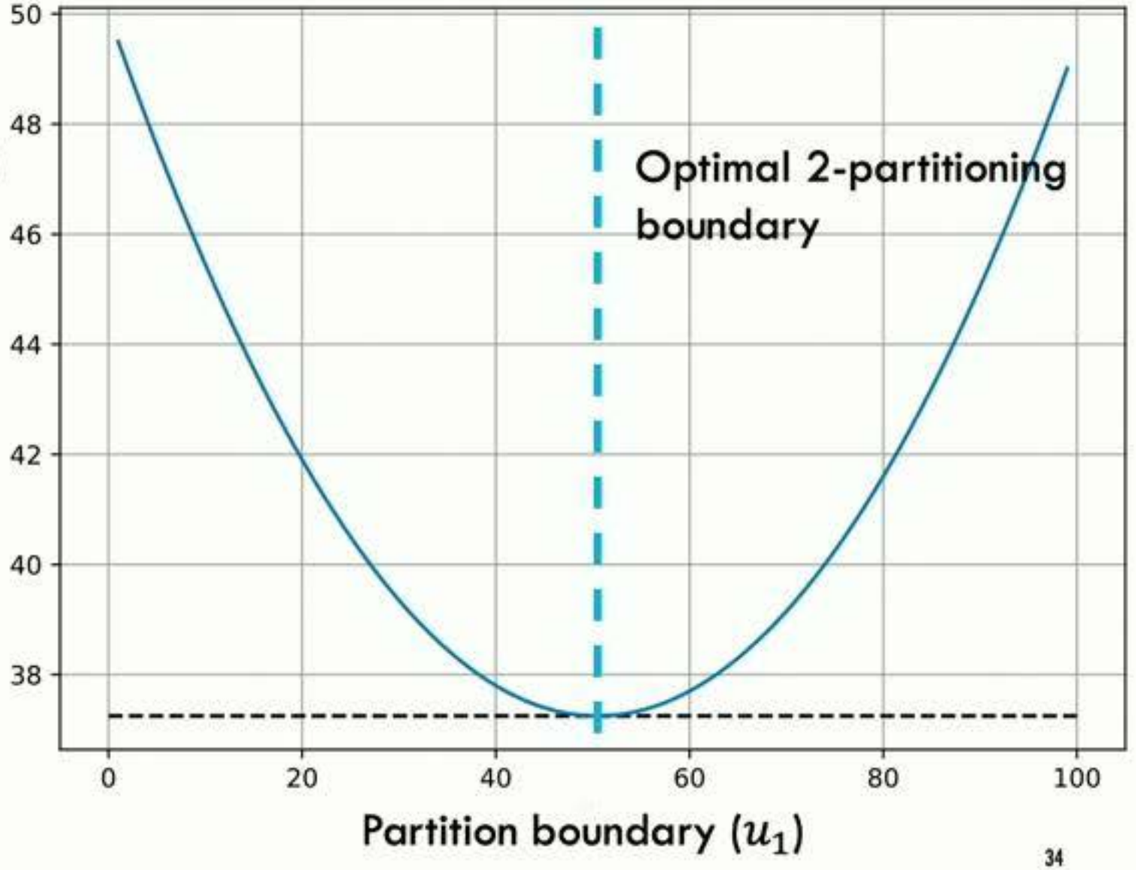
Example: $u = 100$, $l = 1$, $N_{l,u} = 100$, $n = 2$, assume uniform distribution

OPTIMAL PARTITIONING – UNIFORM DIST.

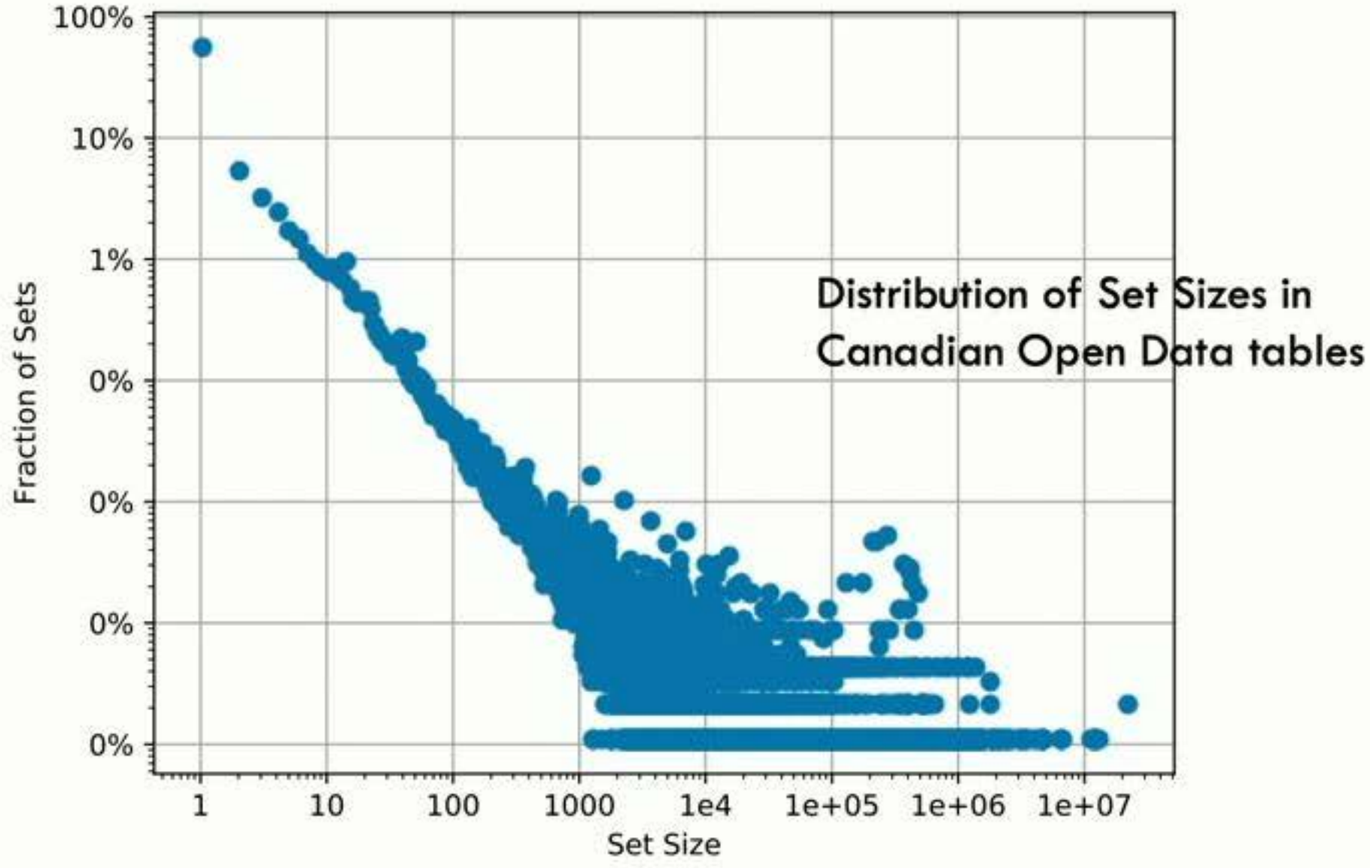
Example: $u = 100$, $l = 1$, $N_{l,u} = 100$, $n = 2$, assume uniform distribution

Number of false positives (upper bound):

$$N^{FP}_{l_1,u_1} + N^{FP}_{l_2,u_2}$$



REAL SIZE DISTRIBUTIONS ARE NOT UNIFORM



OPTIMAL PARTITIONING FOR ANY DISTRIBUTION

Setting $u_1, u_2, \ldots, u_{n-1}$ to minimize the sum of upper bounds of n partitions:

$$\sum_{i=1}^{n} \left(\sum_{x \in [l_i, u_i]} N_x \cdot \left(1 - \frac{x}{u_i} \right) \right)$$

OPTIMAL PARTITIONING FOR ANY DISTRIBUTION

Setting $u_1, u_2, \ldots, u_{n-1}$ to minimize the sum of upper bounds of n partitions:

$$\sum_{i=1}^{n} \left(\sum_{x \in [l_i, u_i]} N_x \cdot \left(1 - \frac{x}{u_i}\right) \right)$$

Dynamic programming to the rescue!

Complexity is $O(|N|^2 \cdot n)$

- N is the domain of all set sizes, n is the number of partitions

OPTIMAL PARTITIONING FOR ANY DISTRIBUTION

Setting $u_1, u_2, \ldots, u_{n-1}$ to minimize the sum of upper bounds of n partitions:

$$\sum_{i=1}^{n} \left(\sum_{x \in [l_i, u_i]} N_x \cdot \left(1 - \frac{x}{u_i}\right) \right)$$

Dynamic programming to the rescue!

Complexity is $O(|N|^2 \cdot n)$

- N is the domain of all set sizes, n is the number of partitions

The complexity is quite manageable in practice as: $|N| \ll |u_n - l_1|$

- $Number\ of\ bank\ accounts \ll |Amount\ in\ Jeff\ Bezos' - Amount\ in\ Eric's|$
- All of Canada, U.S., and U.K. Open Data: $|N| = 14\text{K}$, <10 min in Python on a laptop
- WDC Web Table: $|N| = 4\text{K}$

LSH ENSEMBLE: KEY POINTS

Given a query set Q and containment threshold t^*

- Indexing: DP, MinHash LSH on partitions, $s_u^* = \frac{t^*}{1+\frac{u}{|Q|}-t^*}$
- Querying: probing hash tables; optionally remove false positives
- Also handles arbitrary $|Q|$ and t^*

LSH ENSEMBLE: KEY POINTS

Given a query set Q and containment threshold t^*

- Indexing: DP, MinHash LSH on partitions, $s_u^* = \frac{t^*}{1+\frac{u}{|Q|}-t^*}$
- Querying: probing hash tables; optionally remove false positives
- Also handles arbitrary $|Q|$ and t^*

Parallelize query over partitions

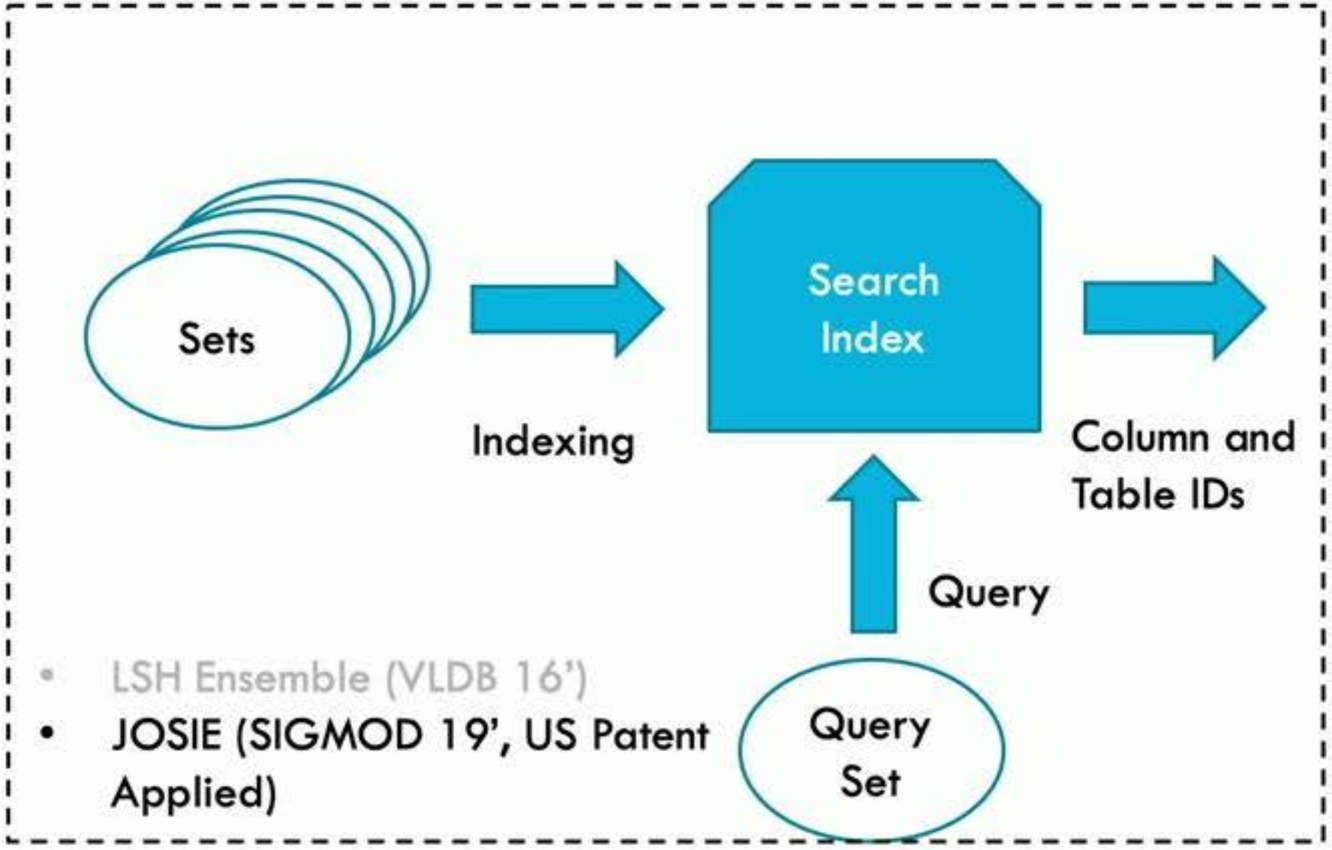
Small query memory footprints

- MinHash is only 1KB (128 hash functions)

Scale to massive number of sets

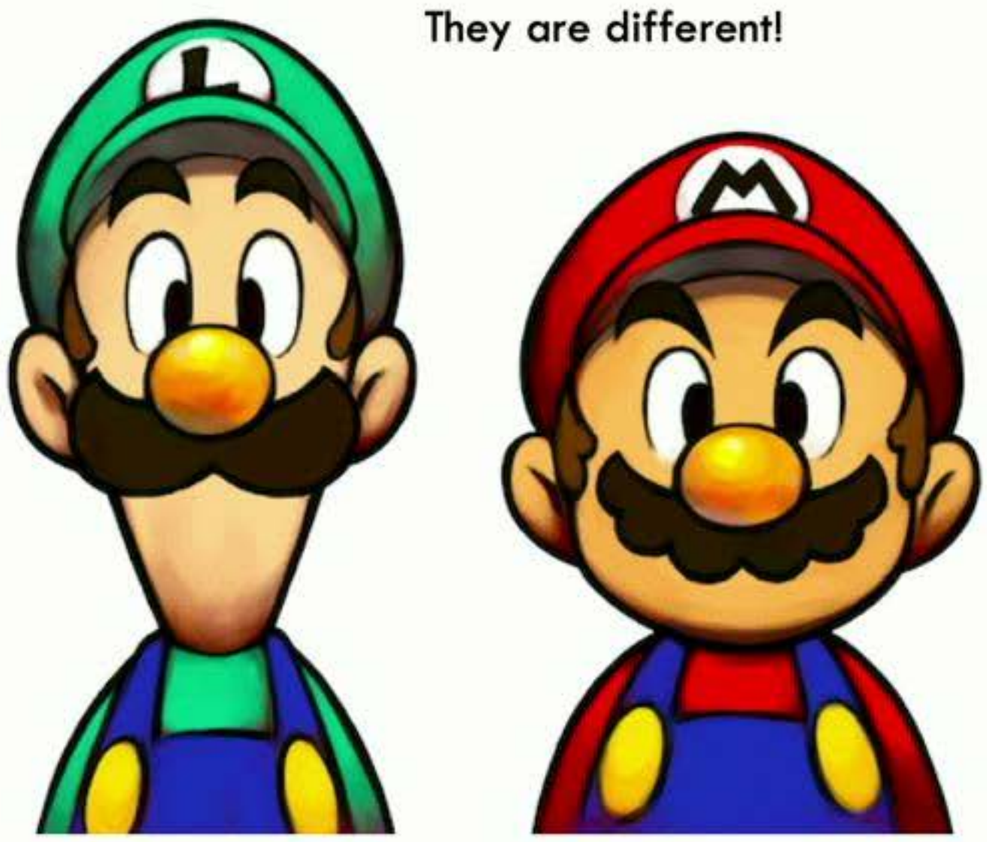
- Tested using 220 Million sets from WDC Web Table (parallel over 5 machines)

A JOINABLE TABLE SEARCH SYSTEM



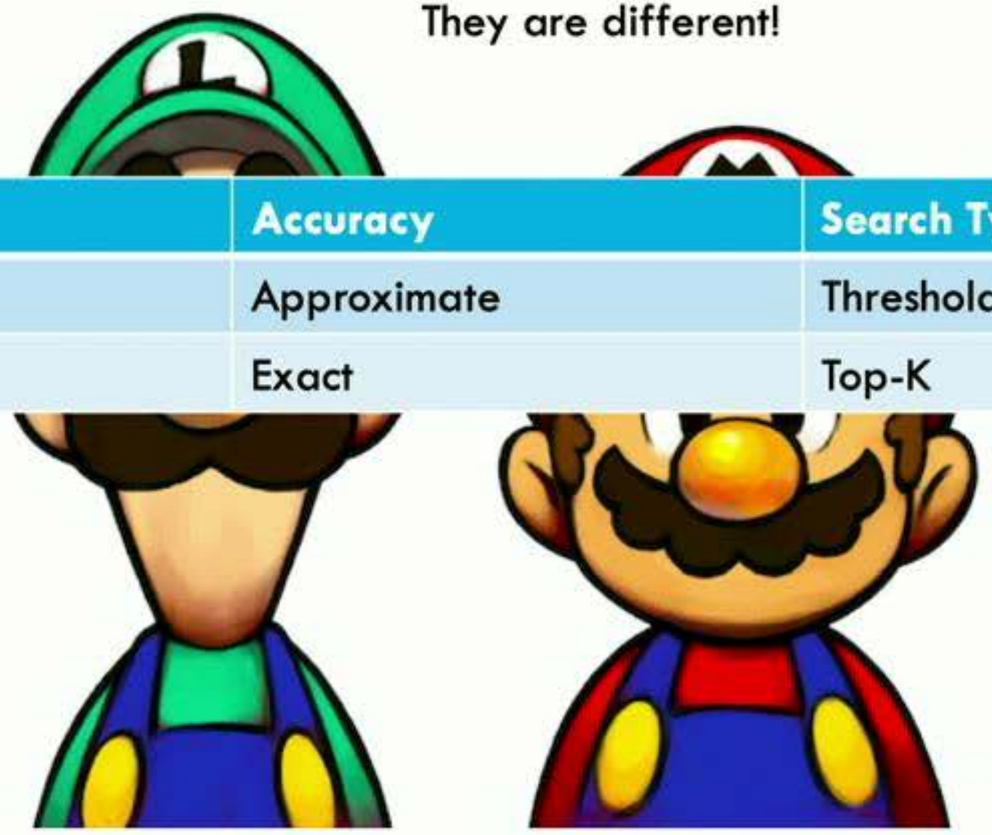
- LSH Ensemble (VLDB 16')
- JOSIE (SIGMOD 19', US Patent Applied)

JOSIE VS. LSH ENSEMBLE



JOSIE VS. LSH ENSEMBLE

They are different!



Algorithm	Accuracy	Search Type
LSH Ensemble	Approximate	Threshold
JOSIE	Exact	Top-K

WHEN TO USE TOP-K SEARCH?

Threshold search: user must specify a containment threshold

- User may not know a good threshold
- User may not understand containment

Top-K problem: just return the best k results

- No knowledge of relevance measure is required
- We showed that for small k (< 20), our exact top-k algorithm can be faster than LSH Ensemble with decreasing threshold hack!

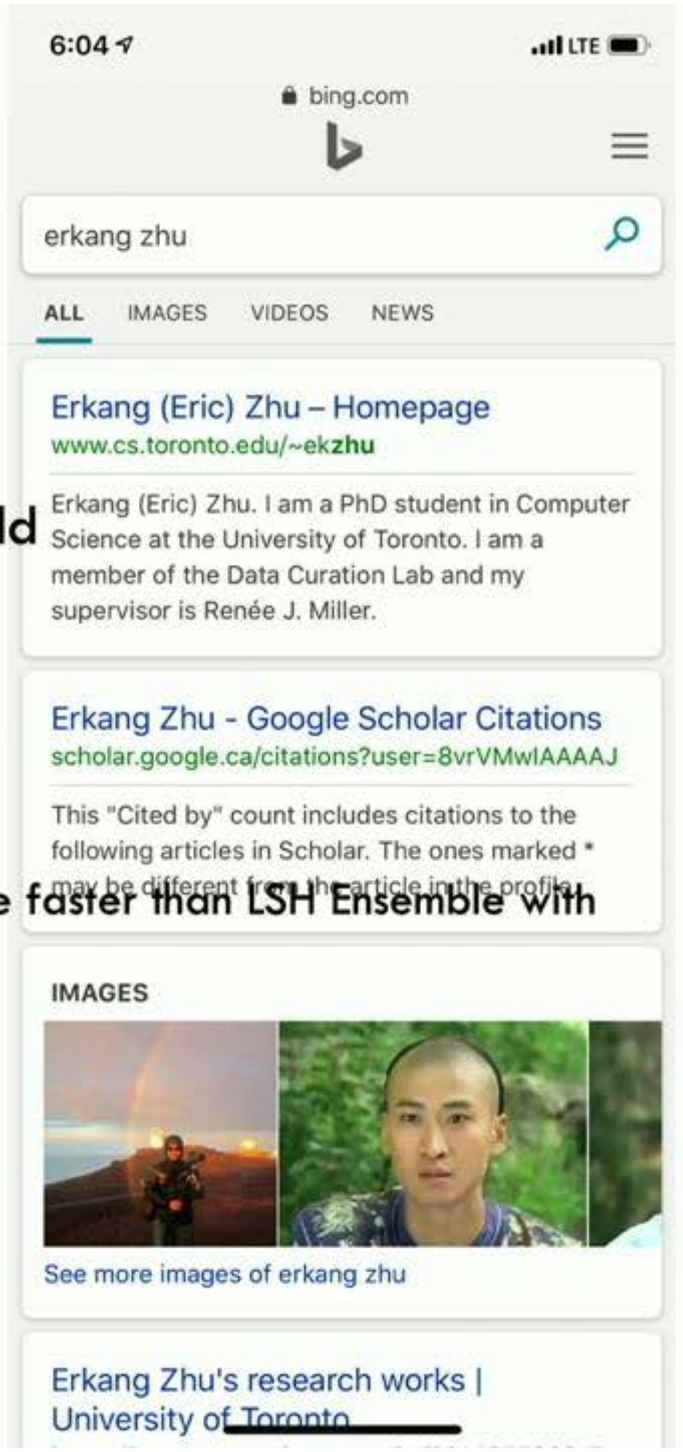
WHEN TO USE TOP-K SEARCH?

Threshold search: user must specify a containment threshold

- User may not know a good threshold
- User may not understand containment

Top-K problem: just return the best k results

- No knowledge of relevance measure is required
- We showed that for small k (< 20), our exact top-k algorithm can be faster than LSH Ensemble with decreasing threshold hack!



WHEN TO USE TOP-K SEARCH?

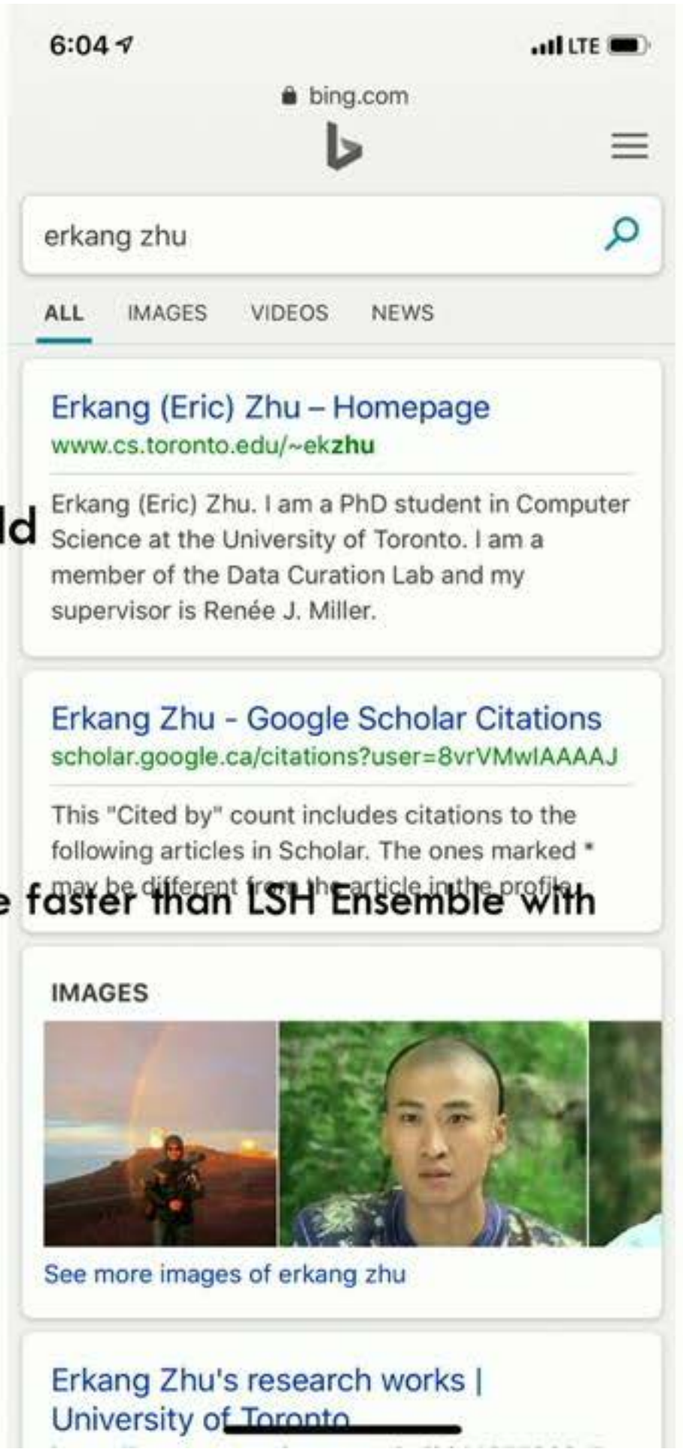
Threshold search: user must specify a containment threshold

- User may not know a good threshold
- User may not understand containment

Top-K problem: just return the best k results

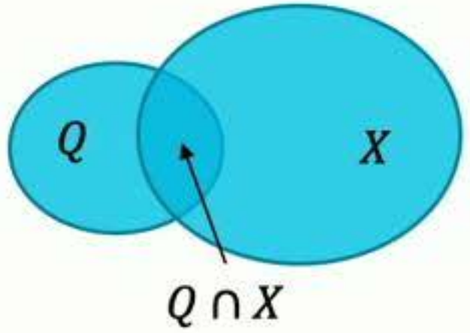
- No knowledge of relevance measure is required
- We showed that for small k (< 20), our exact top-k algorithm can be faster than LSH Ensemble with decreasing threshold hack!

Use top-k for less sophisticated users and small k

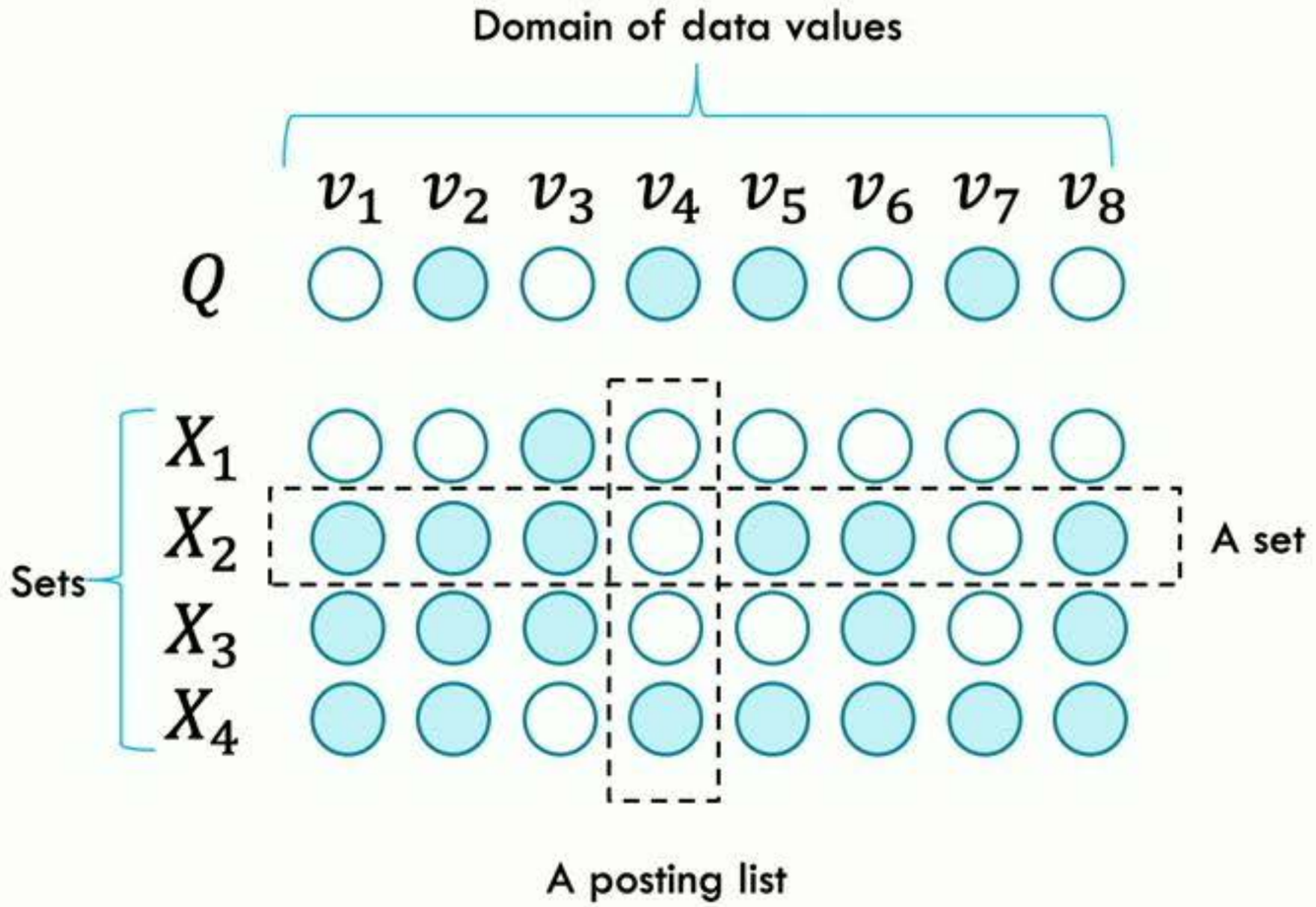


TOP-K OVERLAP SET SIMILARITY SEARCH

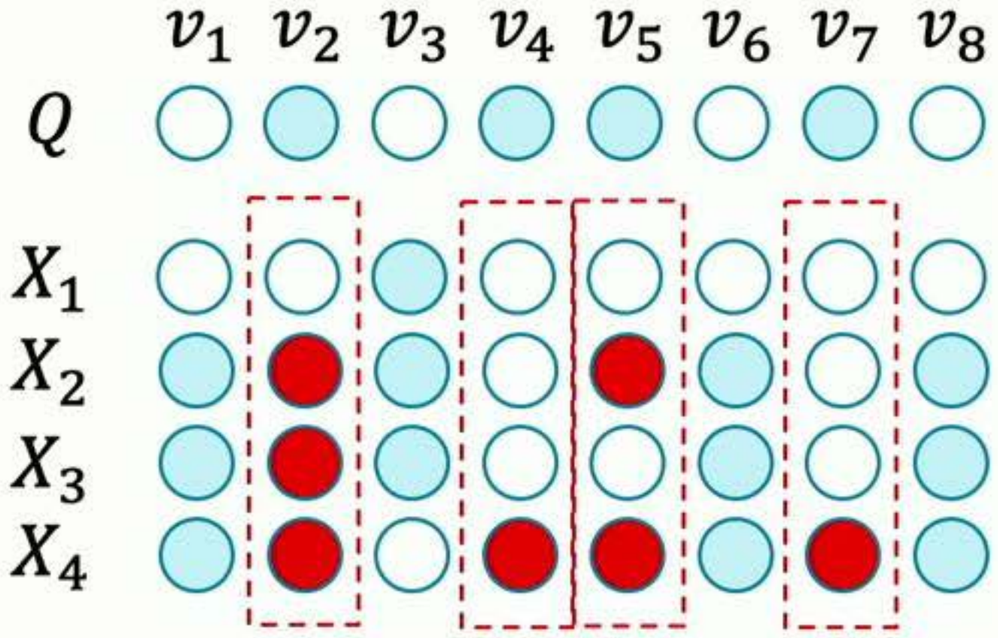
Overlap as the relevance measure: $|Q \cap X|$, equivalent to containment



INVERTED INDEX – A MATRIX PERSPECTIVE



BASELINES FOR FIND TOP-1

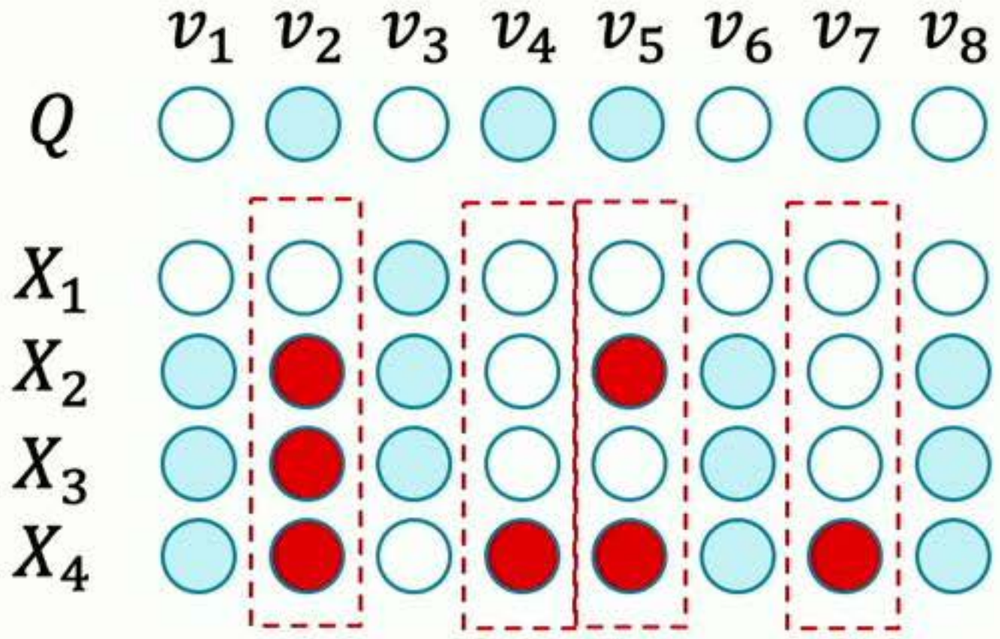


Posting list union[1]: reading all posting lists costs 7 values

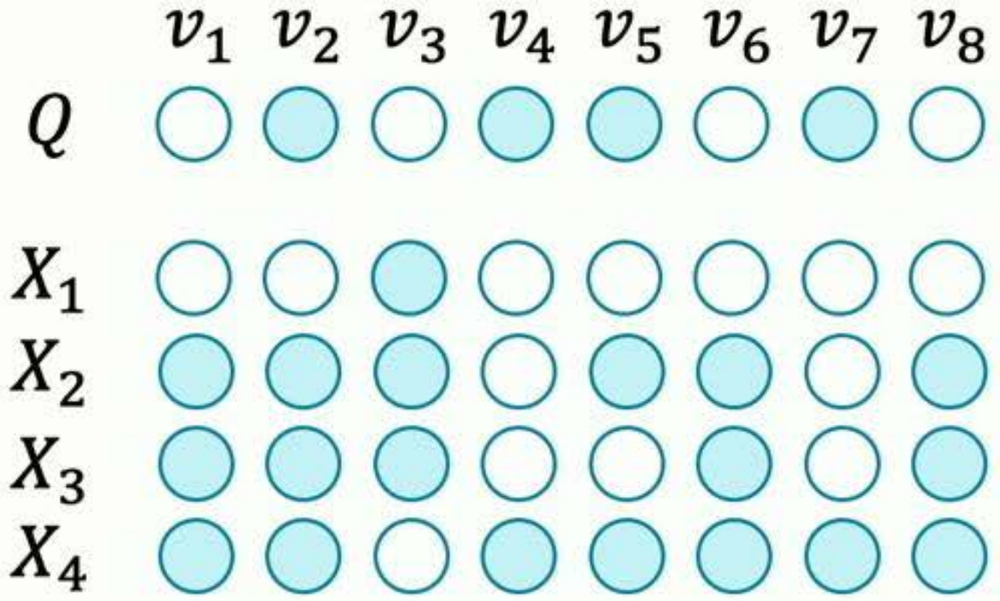
[1] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schu ̈tze. Introduction to Information Retrieval. Cambridge University Press, 2008

[2] Chuan Xiao, Wei Wang, Xuemin Lin, and Haichuan Shang. Top-k set similarity joins. In ICDE, pages 916–927, 2009.

BASELINES FOR FIND TOP-1



Posting list union[1]: reading all posting lists costs 7 values

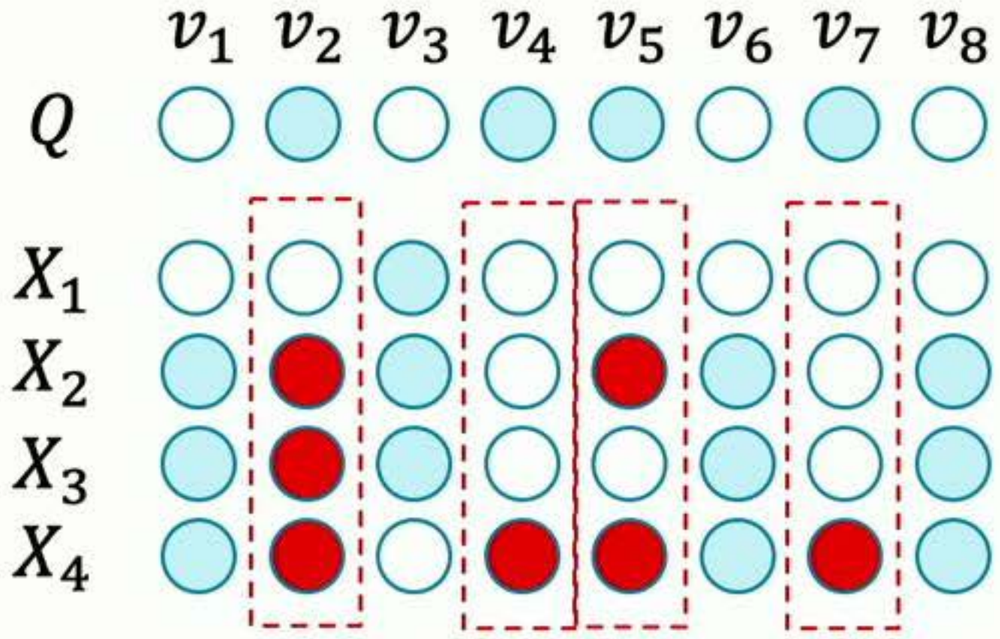


Prefix filtering[2]: reading candidate sets

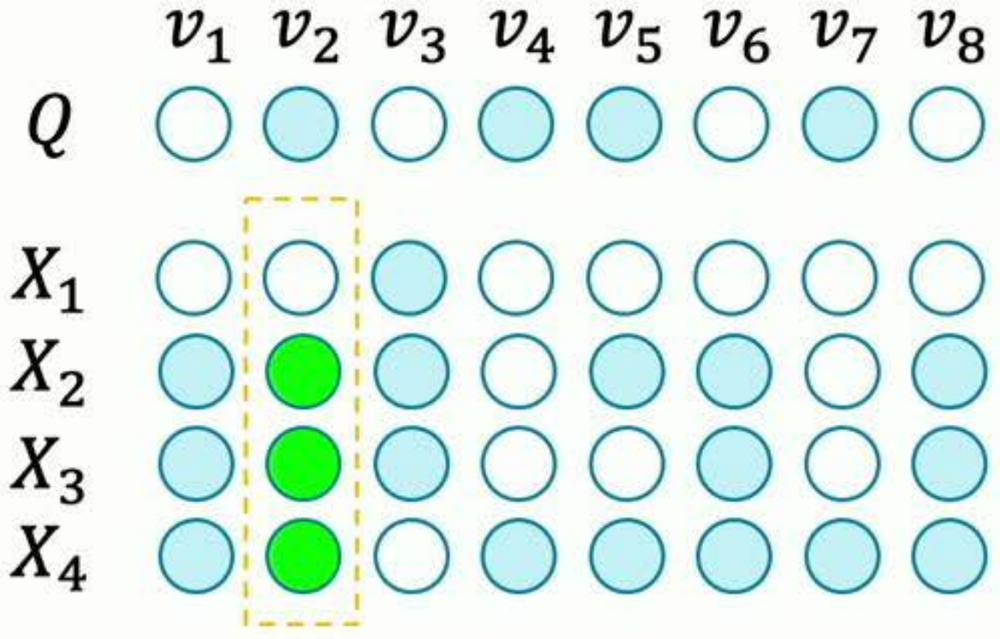
[1] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schu¨tze. Introduction to Information Retrieval. Cambridge University Press, 2008

[2] Chuan Xiao, Wei Wang, Xuemin Lin, and Haichuan Shang. Top-k set similarity joins. In ICDE, pages 916–927, 2009.

BASELINES FOR FIND TOP-1



Posting list union[1]: reading all posting lists costs 7 values

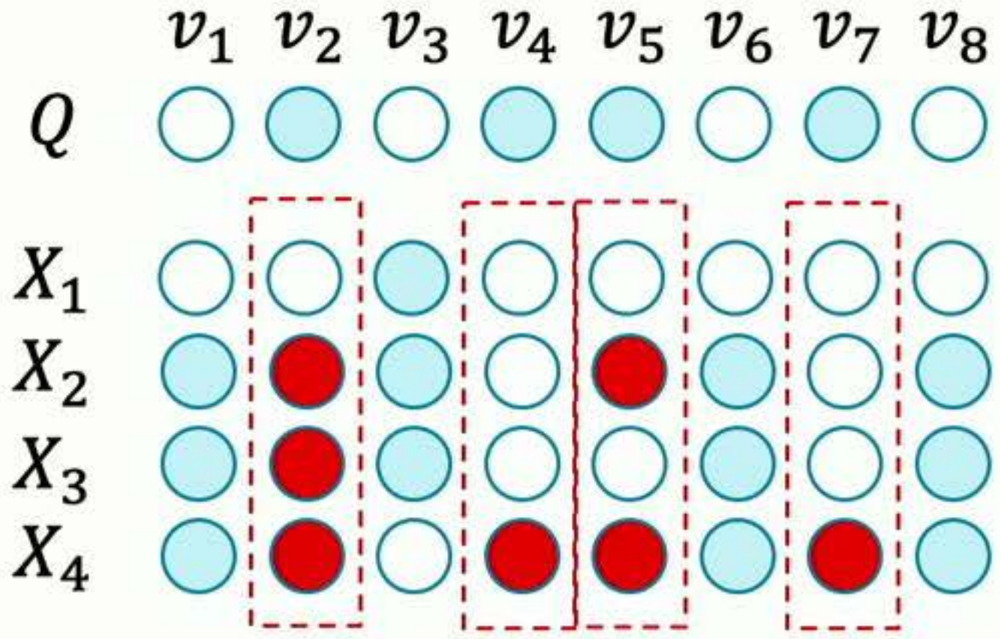


Prefix filtering[2]: reading candidate sets

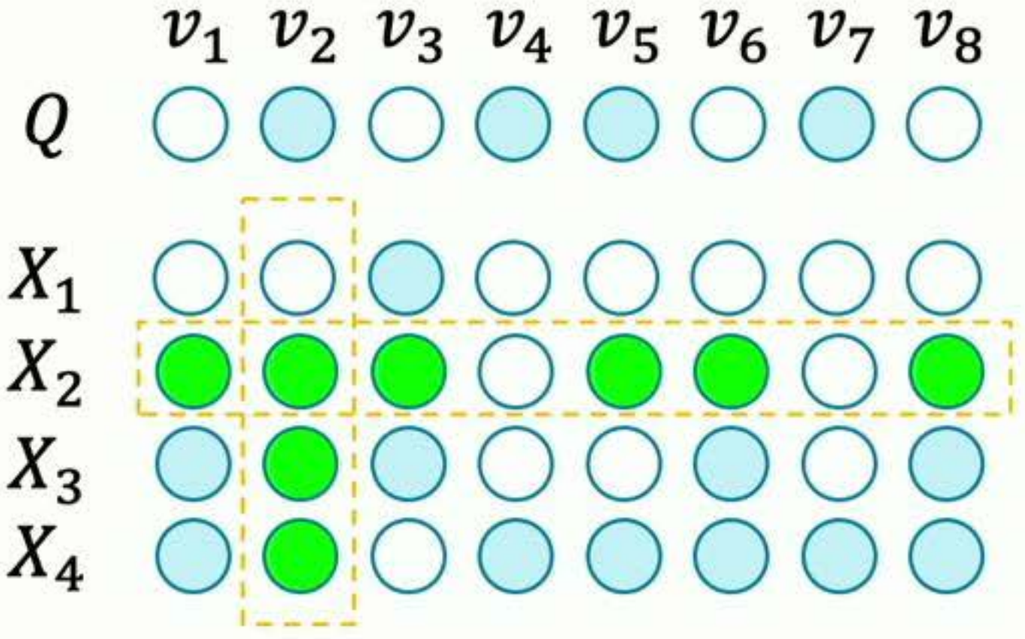
[1] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schu ̈tze. Introduction to Information Retrieval. Cambridge University Press, 2008

[2] Chuan Xiao, Wei Wang, Xuemin Lin, and Haichuan Shang. Top-k set similarity joins. In ICDE, pages 916–927, 2009.

BASELINES FOR FIND TOP-1



Posting list union[1]: reading all posting lists costs 7 values

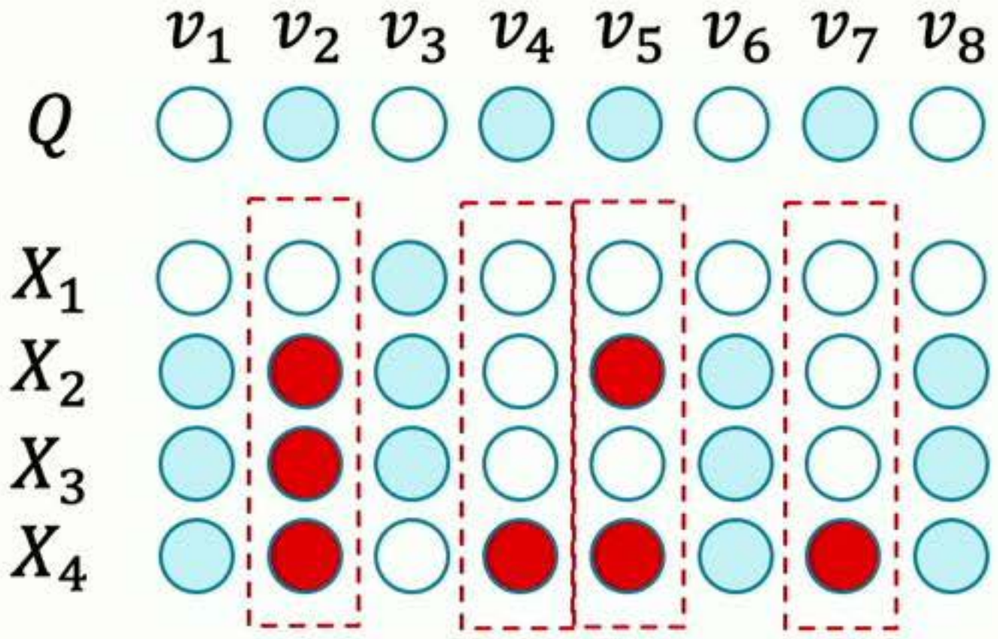


Prefix filtering[2]: reading candidate sets

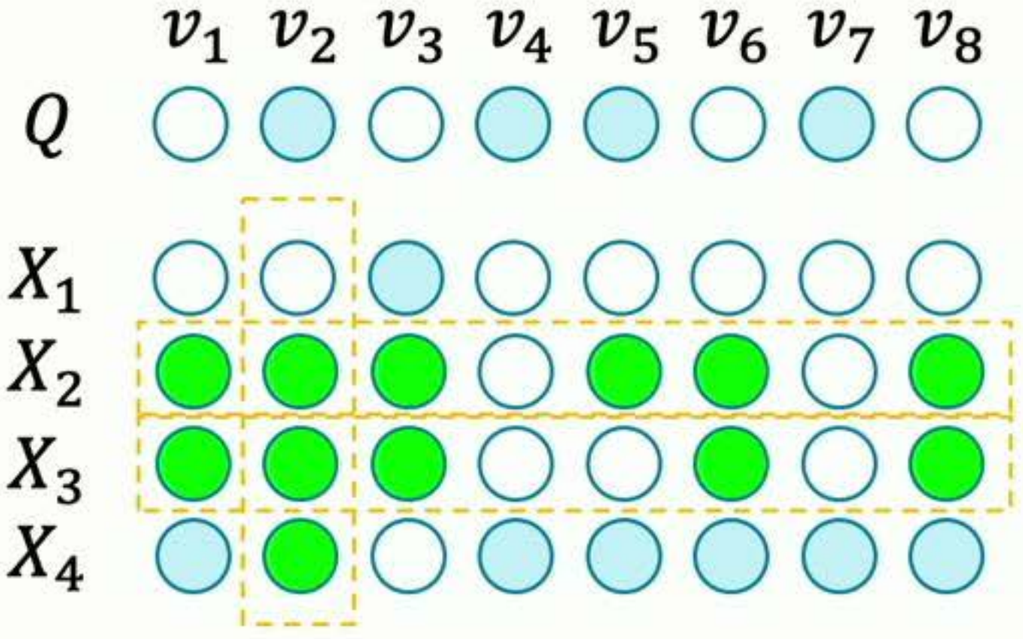
[1] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schu¨tze. Introduction to Information Retrieval. Cambridge University Press, 2008

[2] Chuan Xiao, Wei Wang, Xuemin Lin, and Haichuan Shang. Top-k set similarity joins. In ICDE, pages 916–927, 2009.

BASELINES FOR FIND TOP-1



Posting list union[1]: reading all posting lists costs 7 values

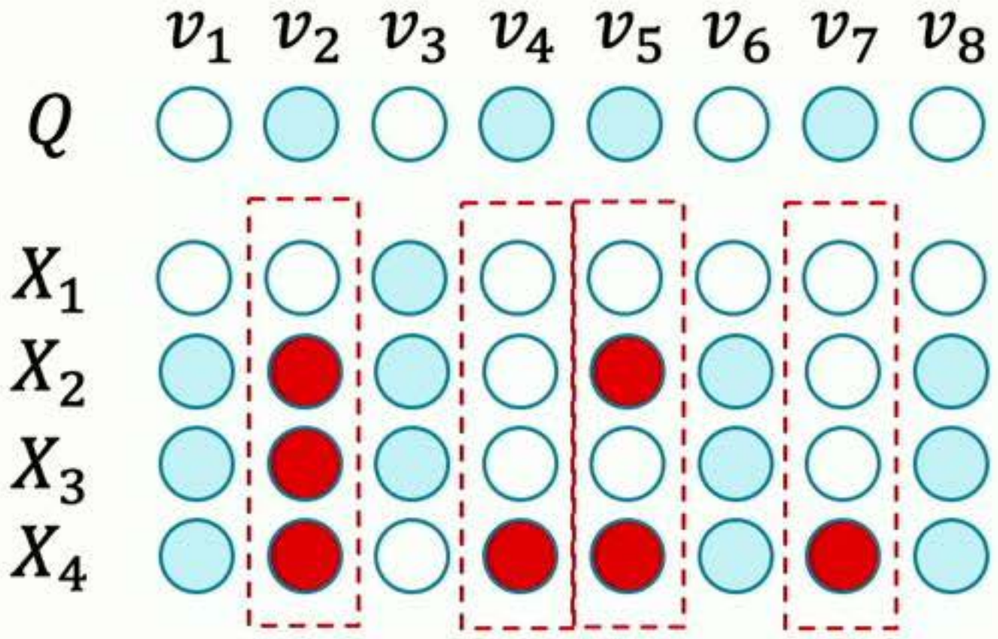


Prefix filtering[2]: reading candidate sets

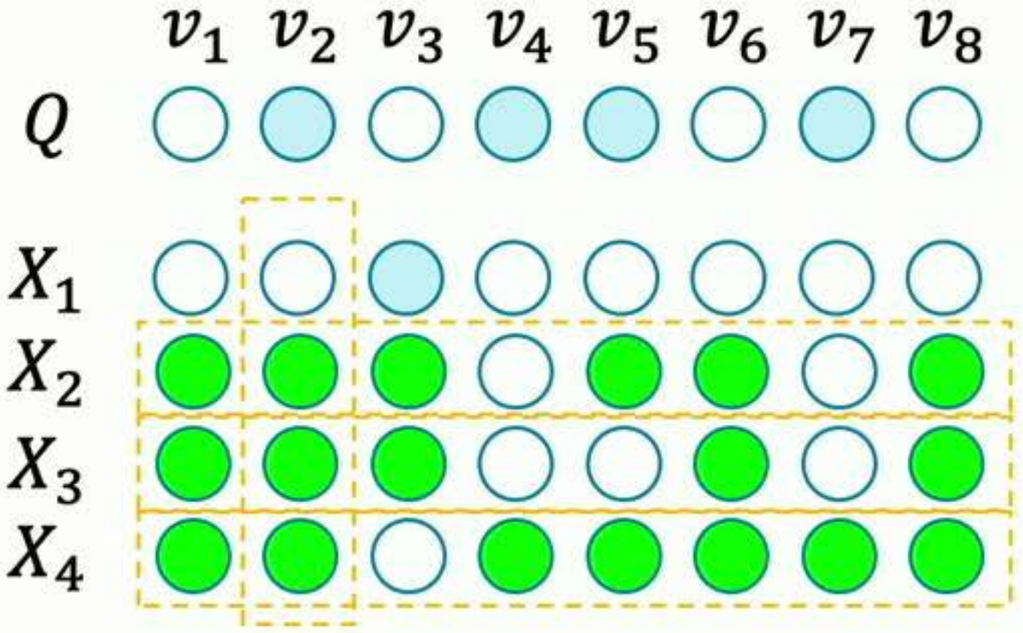
[1] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schu ̈tze. Introduction to Information Retrieval. Cambridge University Press, 2008

[2] Chuan Xiao, Wei Wang, Xuemin Lin, and Haichuan Shang. Top-k set similarity joins. In ICDE, pages 916–927, 2009.

BASELINES FOR FIND TOP-1



Posting list union[1]: reading all posting lists costs 7 values

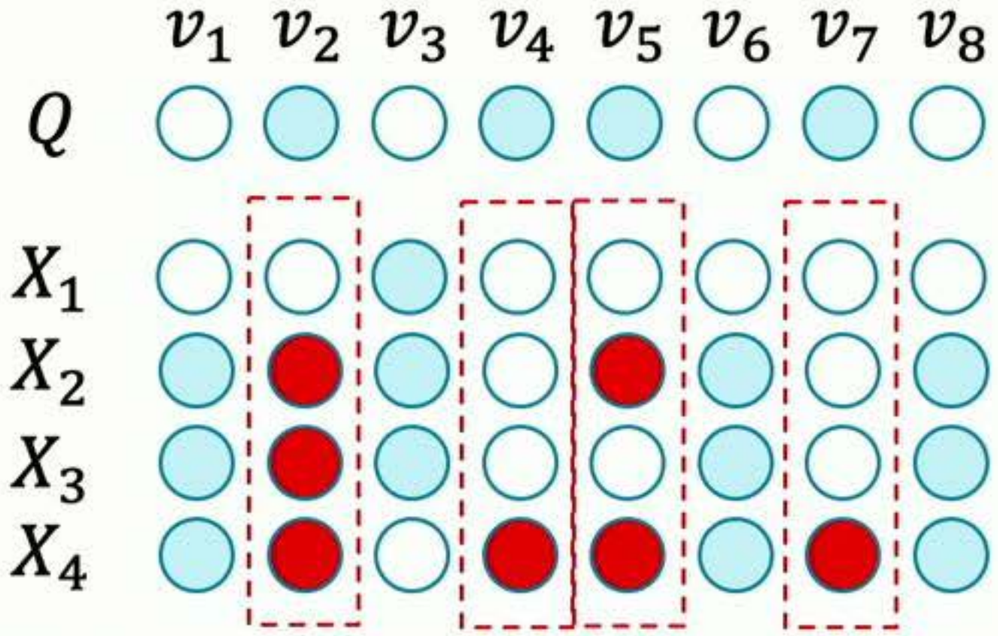


Prefix filtering[2]: reading candidate sets

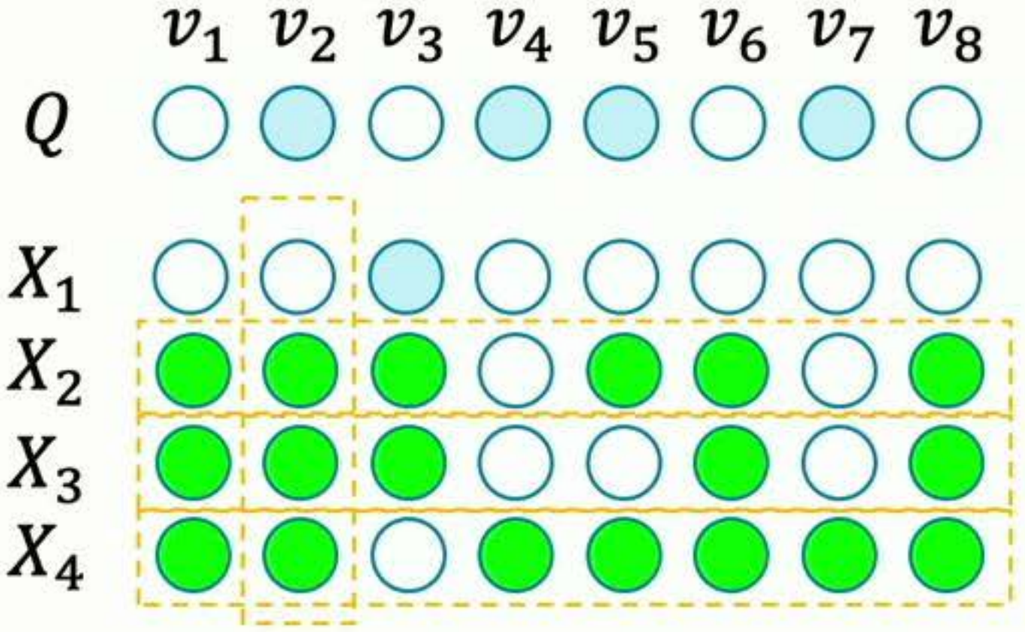
[1] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schu¨tze. Introduction to Information Retrieval. Cambridge University Press, 2008

[2] Chuan Xiao, Wei Wang, Xuemin Lin, and Haichuan Shang. Top-k set similarity joins. In ICDE, pages 916–927, 2009.

BASELINES FOR FIND TOP-1



Posting list union[1]: reading all posting lists costs 7 values

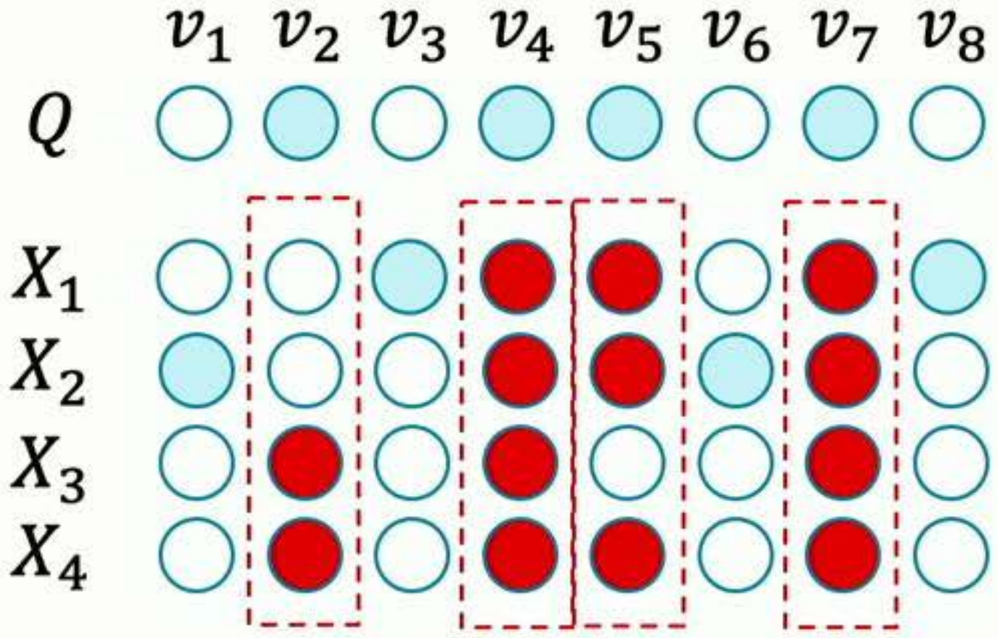


Prefix filtering[2]: reading candidate sets costs 18 values

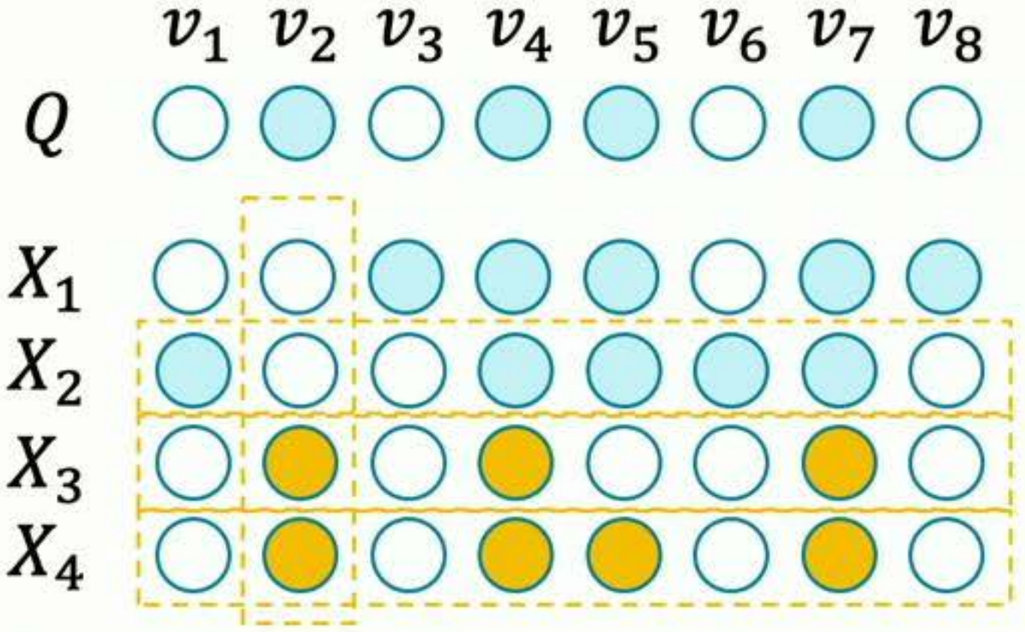
[1] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schu¨tze. Introduction to Information Retrieval. Cambridge University Press, 2008

[2] Chuan Xiao, Wei Wang, Xuemin Lin, and Haichuan Shang. Top-k set similarity joins. In ICDE, pages 916–927, 2009.

BASELINES FOR FIND TOP-1



Posting list union[1]: reading all posting lists costs 13 values



Prefix filtering[2]: reading candidate sets costs 7 values

[1] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schu¨tze. Introduction to Information Retrieval. Cambridge University Press, 2008

[2] Chuan Xiao, Wei Wang, Xuemin Lin, and Haichuan Shang. Top-k set similarity joins. In ICDE, pages 916–927, 2009.

COST MATTERS IN DATA LAKES

Dataset	# of Sets	Max Size	Avg. Size	# of Uniq. Values
Open Data*	745K	22M	1,540	562M
WDC Web Table	163M	17K	10	184M
AOL (Query Logs)	10M	245	3	3.9M
ERON (Emails)	517K	3,162	135	1.1M
DBLP (Bibliographies)	100K	1,625	86	6,864

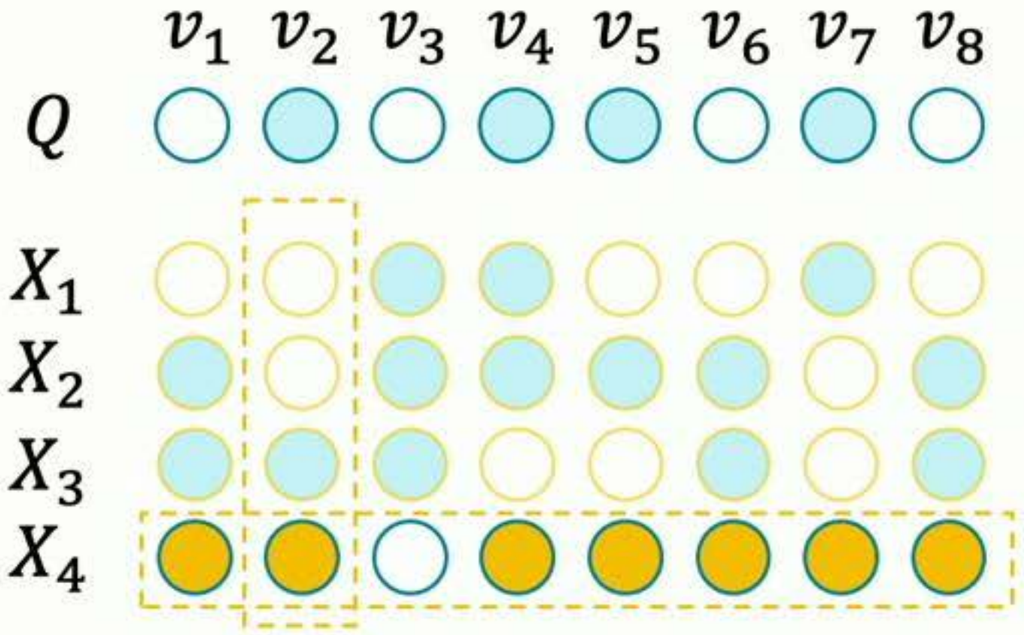
Data Lakes

*215,393 Open Data tables from Canadian, US, and UK Open Data Portal

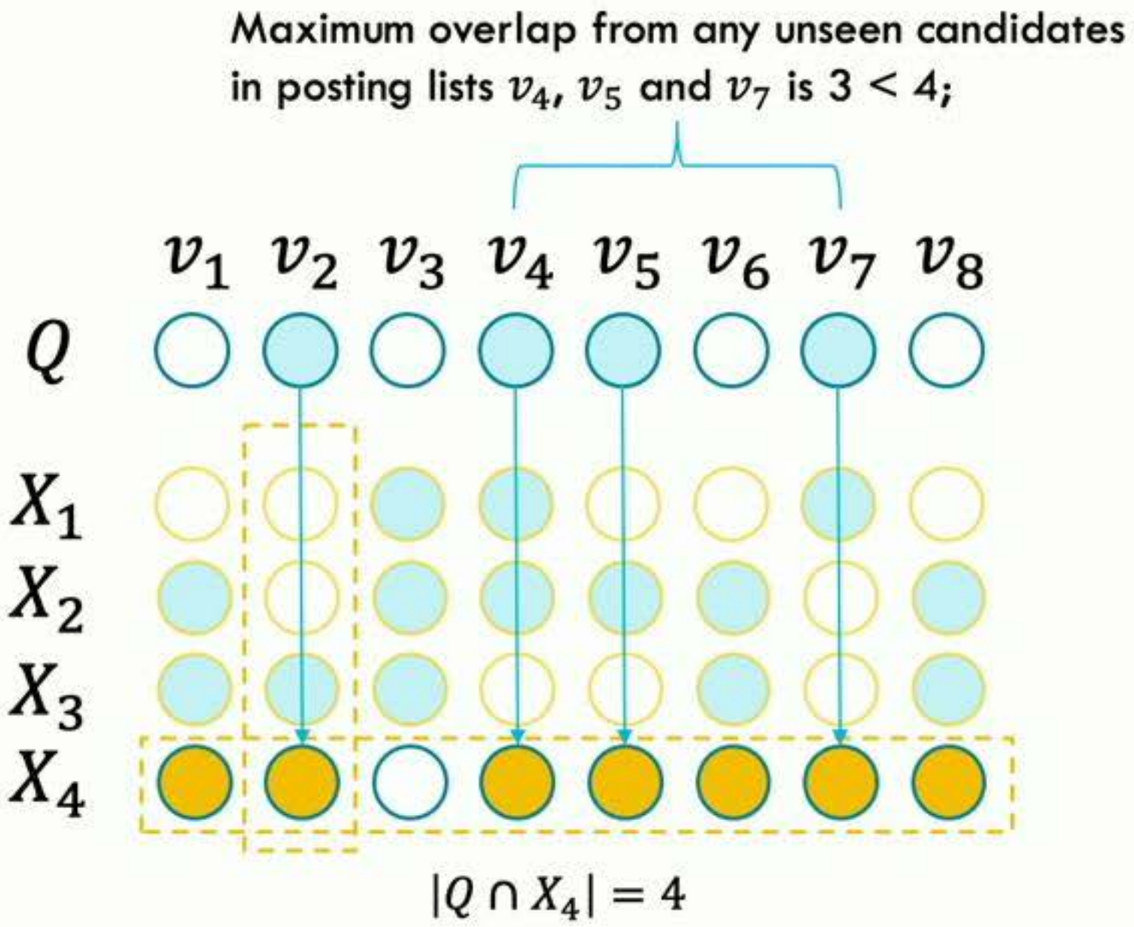
Read Bottleneck – Large number of sets and data values makes index access expensive; large posting lists and sets are expensive to read

An efficient algorithm must reduce read cost

READING SET ELIMINATES POSTING LISTS

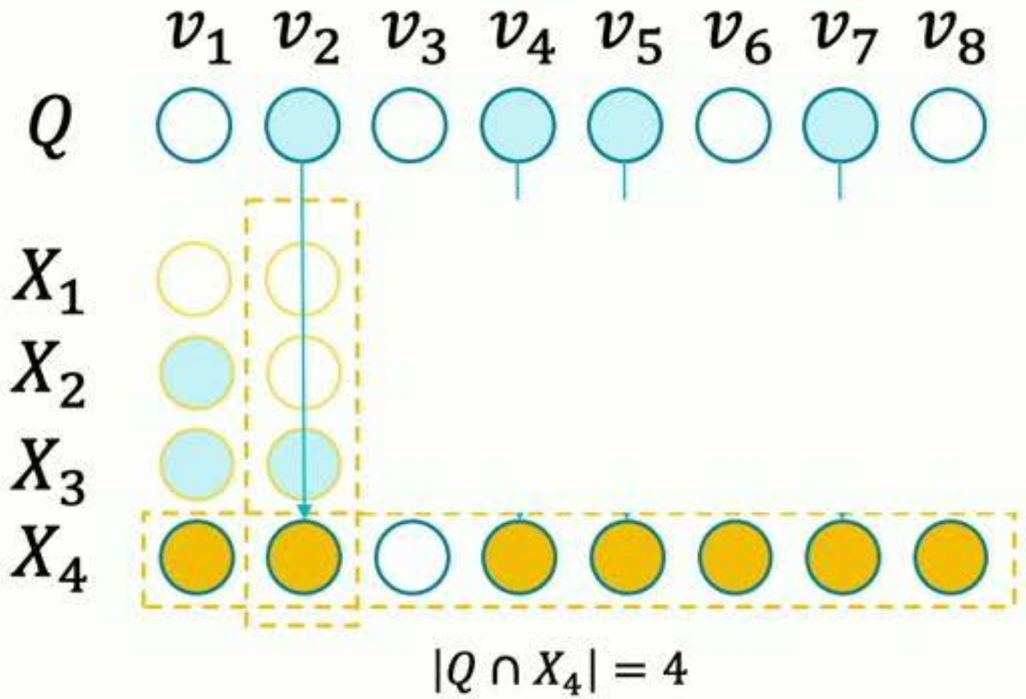


READING SET ELIMINATES POSTING LISTS



READING SET ELIMINATES POSTING LISTS

Maximum overlap from any unseen candidates in posting lists v_4, v_5 and v_7 is $3 < 4$;

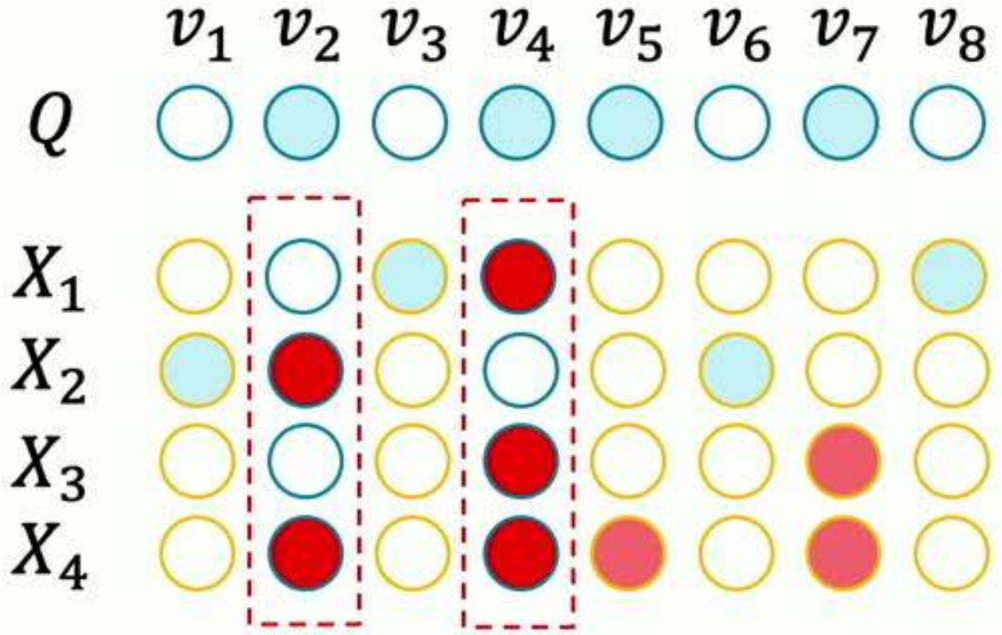


$|Q \cap X_4| = 4$

X_4 is the top-1; **eliminate** posting lists v_4, v_5 and v_7

READING POSTING LISTS ELIMINATE SETS

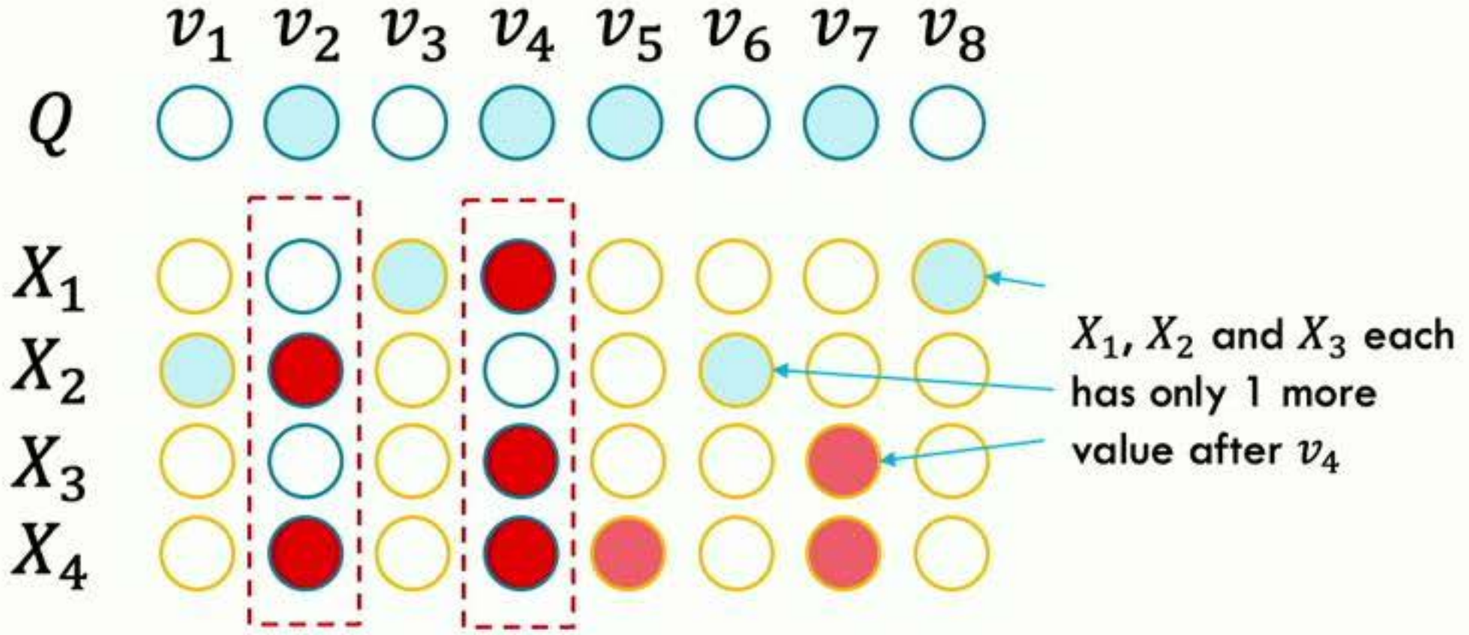
Current candidate sets are:
$\{X_1, X_2, X_3, X_4\}$



Assume the current top-1
has 4 overlaps

READING POSTING LISTS ELIMINATE SETS

Current candidate sets are:
$\{X_1, X_2, X_3, X_4\}$

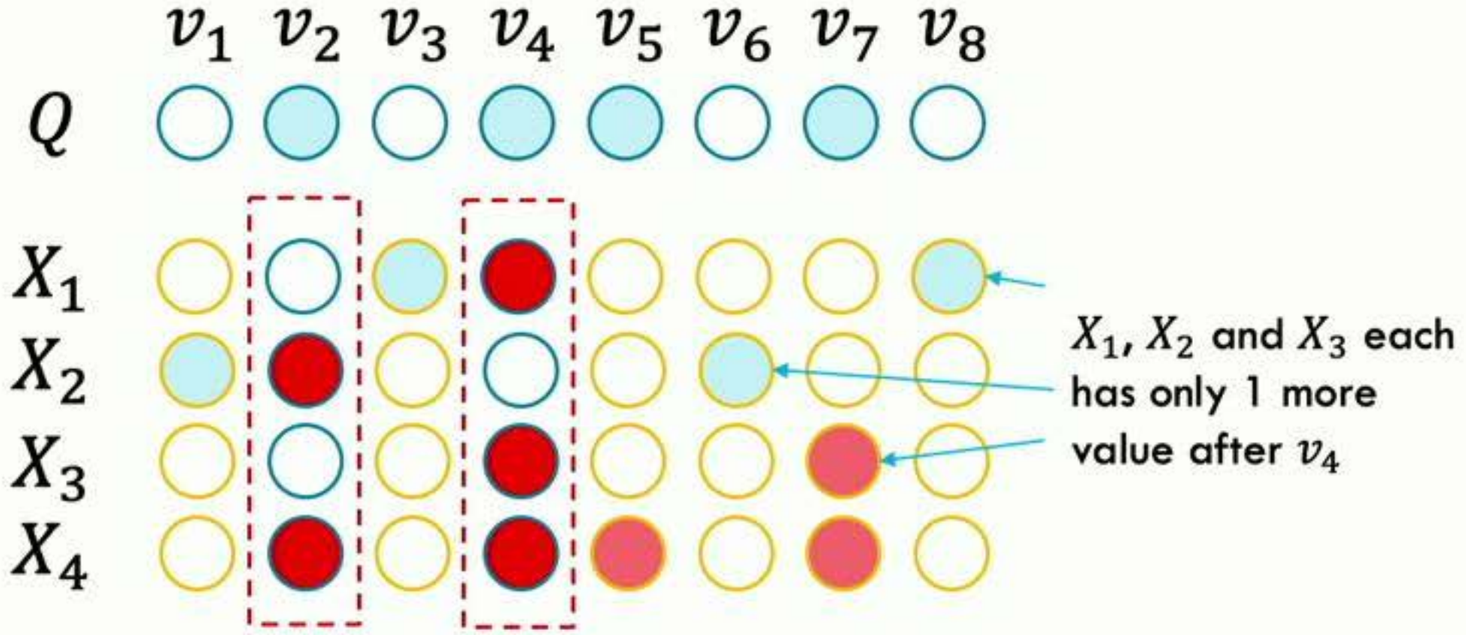


Assume the current top-1 has 4 overlaps

READING POSTING LISTS ELIMINATE SETS

Current candidate sets are:
$\{X_1, X_2, X_3, X_4\}$

X_1, X_2 and X_3 can have at most 2 overlaps, less than the current top-1; **eliminate**

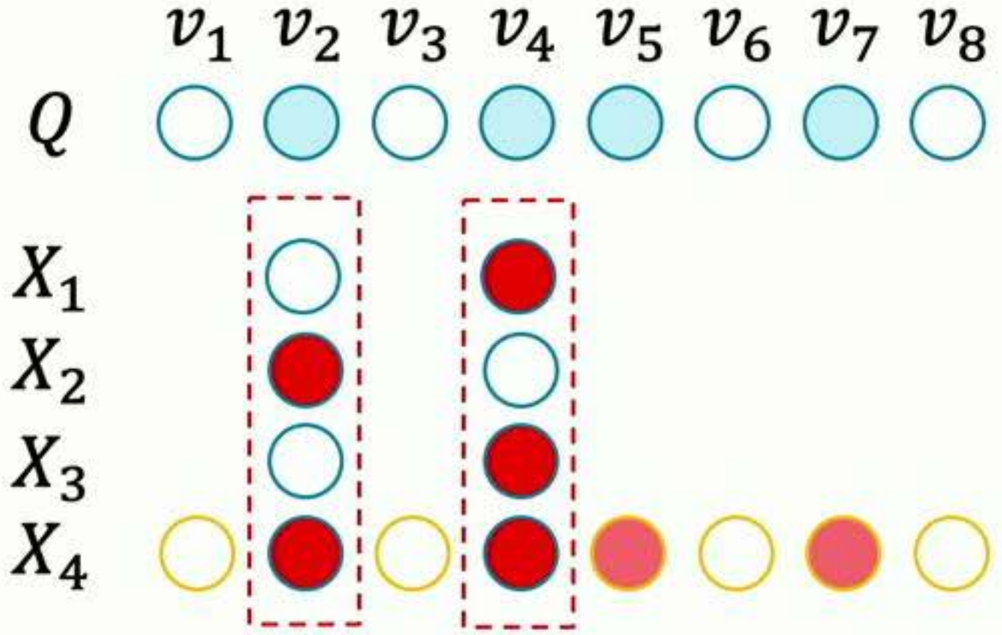


Assume the current top-1 has 4 overlaps

READING POSTING LISTS ELIMINATE SETS

Current candidate sets are:
$\{$ $,X_4\}$

X_1, X_2 and X_3 can have at most 2 overlaps, less than the current top-1; **eliminate**



Assume the current top-1 has 4 overlaps

X_1, X_2 and X_3 each has only 1 more value after v_4

JOSIE - JOIN SEARCH VIA INTERSECTION EST.[1]

Measure "work" as total read cost of remaining posting lists and candidates

Measure "progress" as reduction in work

- Reading posting lists reveals intersecting values – reducing work in reading candidates
- Reading candidates improves prefix filter – reducing work in reading posting lists

Measure "price" as the read cost associated with progress

[1] Erkang Zhu, Dong Deng, Fatemeh Nargesian, Renée J. Miller, "JOSIE: Overlap Set Similarity Search for Finding Joinable Tables in Data Lakes", SIGMOD 2019

JOSIE - JOIN SEARCH VIA INTERSECTION EST.[1]

Measure "work" as total read cost of remaining posting lists and candidates

Measure "progress" as reduction in work

- Reading posting lists reveals intersecting values – reducing work in reading candidates
- Reading candidates improves prefix filter – reducing work in reading posting lists

Measure "price" as the read cost associated with progress

```
while work > 0:
    estimate progress
    choose reading posting lists or candidates
        that maximize (progress - price)
```

[1] Erkang Zhu, Dong Deng, Fatemeh Nargesian, Renée J. Miller, "JOSIE: Overlap Set Similarity Search for Finding Joinable Tables in Data Lakes", SIGMOD 2019

JOSIE - JOIN SEARCH VIA INTERSECTION EST.[1]

Measure "work" as total read cost of remaining posting lists and candidates

Measure "progress" as reduction in work

- Reading posting lists reveals intersecting values – reducing work in reading candidates
- Reading candidates improves prefix filter – reducing work in reading posting lists

Measure "price" as the read cost associated with progress

```
while work > 0:
    estimate progress
    choose reading posting lists or candidates
        that maximize (progress - price)
```

Cost model

[1] Erkang Zhu, Dong Deng, Fatemeh Nargesian, Renée J. Miller, "JOSIE: Overlap Set Similarity Search for Finding Joinable Tables in Data Lakes", SIGMOD 2019

JOSIE - JOIN SEARCH VIA INTERSECTION EST.[1]

Measure "work" as total read cost of remaining posting lists and candidates

Measure "progress" as reduction in work

- Reading posting lists reveals intersecting values – reducing work in reading candidates
- Reading candidates improves prefix filter – reducing work in reading posting lists

Measure "price" as the read cost associated with progress

```
while work > 0:
    estimate progress
    choose reading posting lists or candidates
        that maximize (progress - price)
```

Cost model

[1] Erkang Zhu, Dong Deng, Fatemeh Nargesian, Renée J. Miller, "JOSIE: Overlap Set Similarity Search for Finding Joinable Tables in Data Lakes", SIGMOD 2019

JOSIE - PERFORMANCE

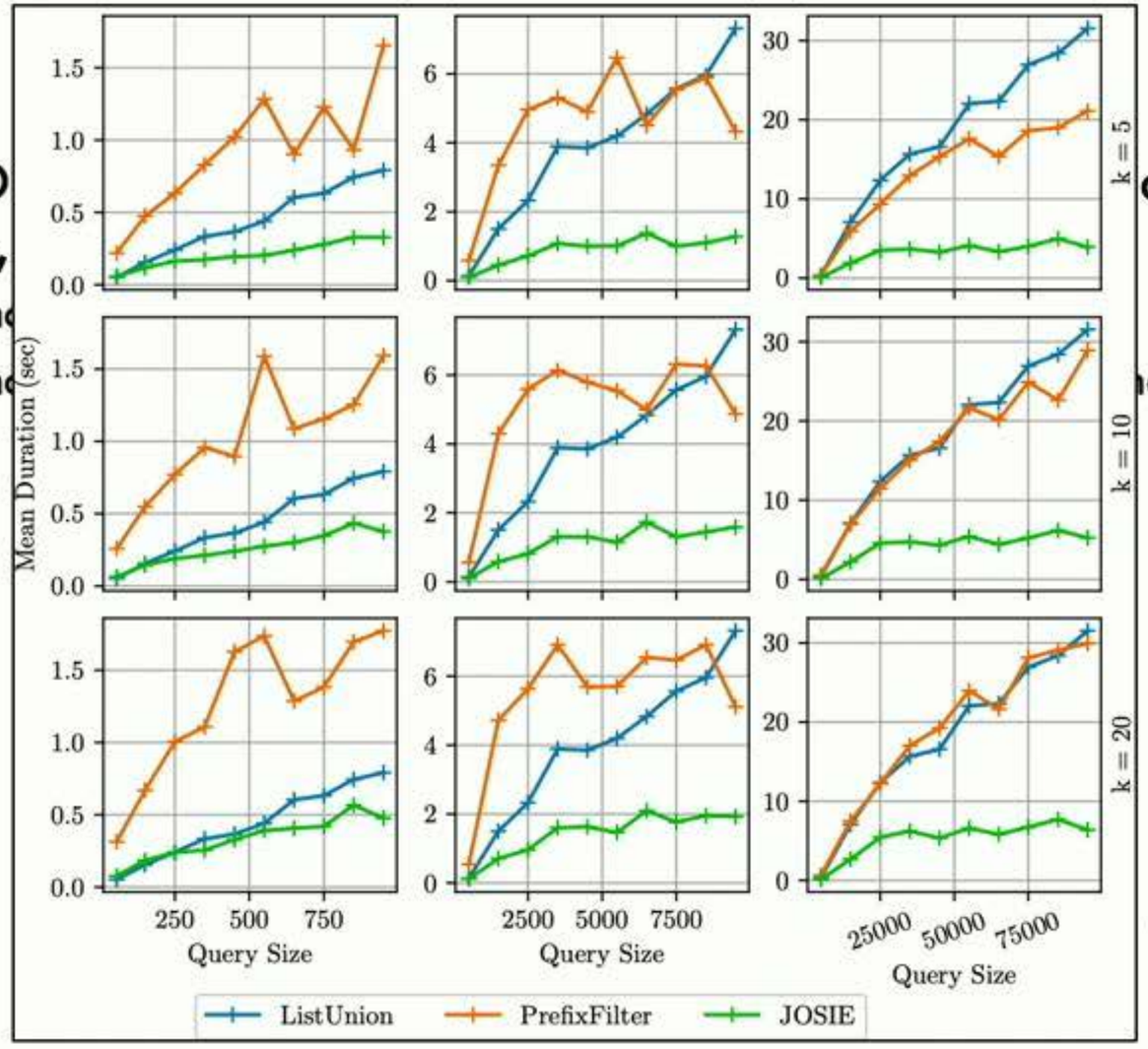
Evaluated on an Open Data lake (745K sets, 562M posting lists) and WDC Web Tables (163M sets, 184M posting lists)

- Up to 5× performance over baseline algorithms
- Up to 2× performance over LSH Ensemble (approximate), small queries (<5K) and $k < 20$

JOSIE - PERFORMANCE

Evaluated on an O … d WDC Web Tables (163M sets,

- Up to 5× performanc…
- Up to 2× performanc… … d $k < 20$



On Open Data Benchmark

OTHER WORK

Auto-Join: Join Table by Leveraging Transformation (VLDB 17')

- Collaboration with Yeye He and Surajit Chaudhuri @ MSR

Learning transformation **without** user input

Company	Headquarter	Ticker
Alphabet Inc.	Mountain View	GOOGL (NASDAQ)
Apple Inc.	Palo Alto	AAPL (NASDAQ)
Alibaba	Hangzhou	BABA (NYSE)

Exchange	Ticker	Open	Close	Market Cap
NASDAQ	GOOGL	1,032.50	1,042.32	766 B
NASDAQ	AAPL	171.10	175.12	892 B
NYSE	BABA	179.37	174.13	427 B

OTHER WORK

Table Union Search on Open Data[1] (VLDB 17')

- Given a query table, find tables that can be "stacked" with the query table and aligning columns
- Top-K and Approximate

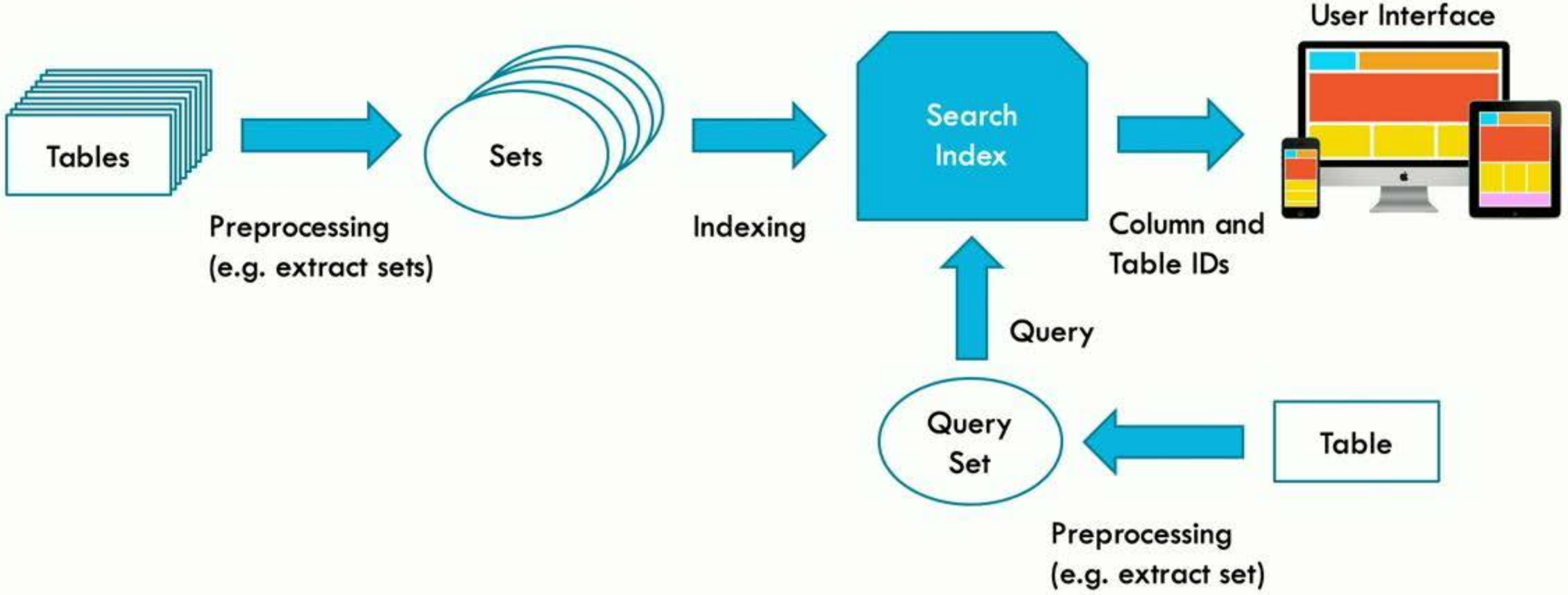
Optimizing Organizations for Navigating Data Lakes[2] (VLDB 19' Revision)

- Generate a navigational DAG of topics for browsing the data lakes effectively
- Complementary to search

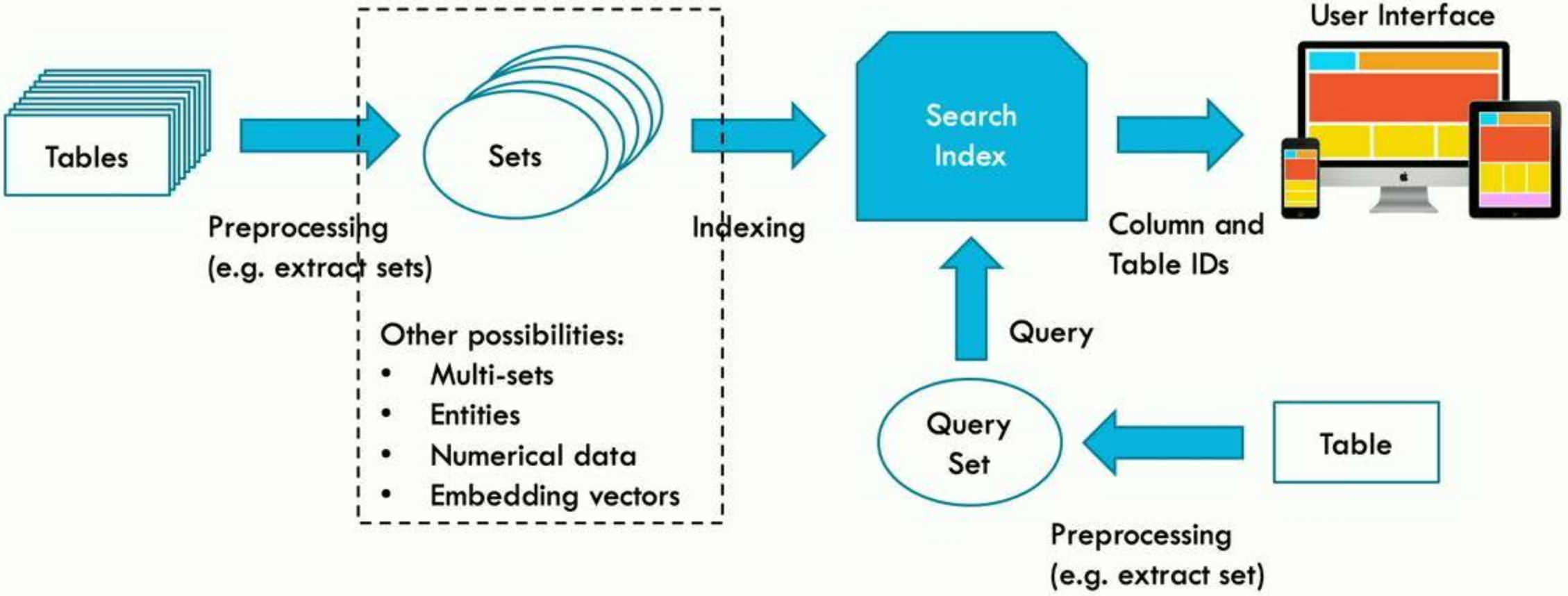
[1] Fatemeh Nargesian, **Erkang Zhu**, Ken Pu, Renée J. Miller, "Table Union Search on Open Data", VLDB 2017

[2] Fatemeh Nargesian, Ken Pu, **Erkang Zhu**, Bahar Ghadiri Bashardoost, Renée J. Miller, "Optimizing Organizations for Navigating Data Lakes" VLDB 2019 (Revision), arXiv:1812.07024

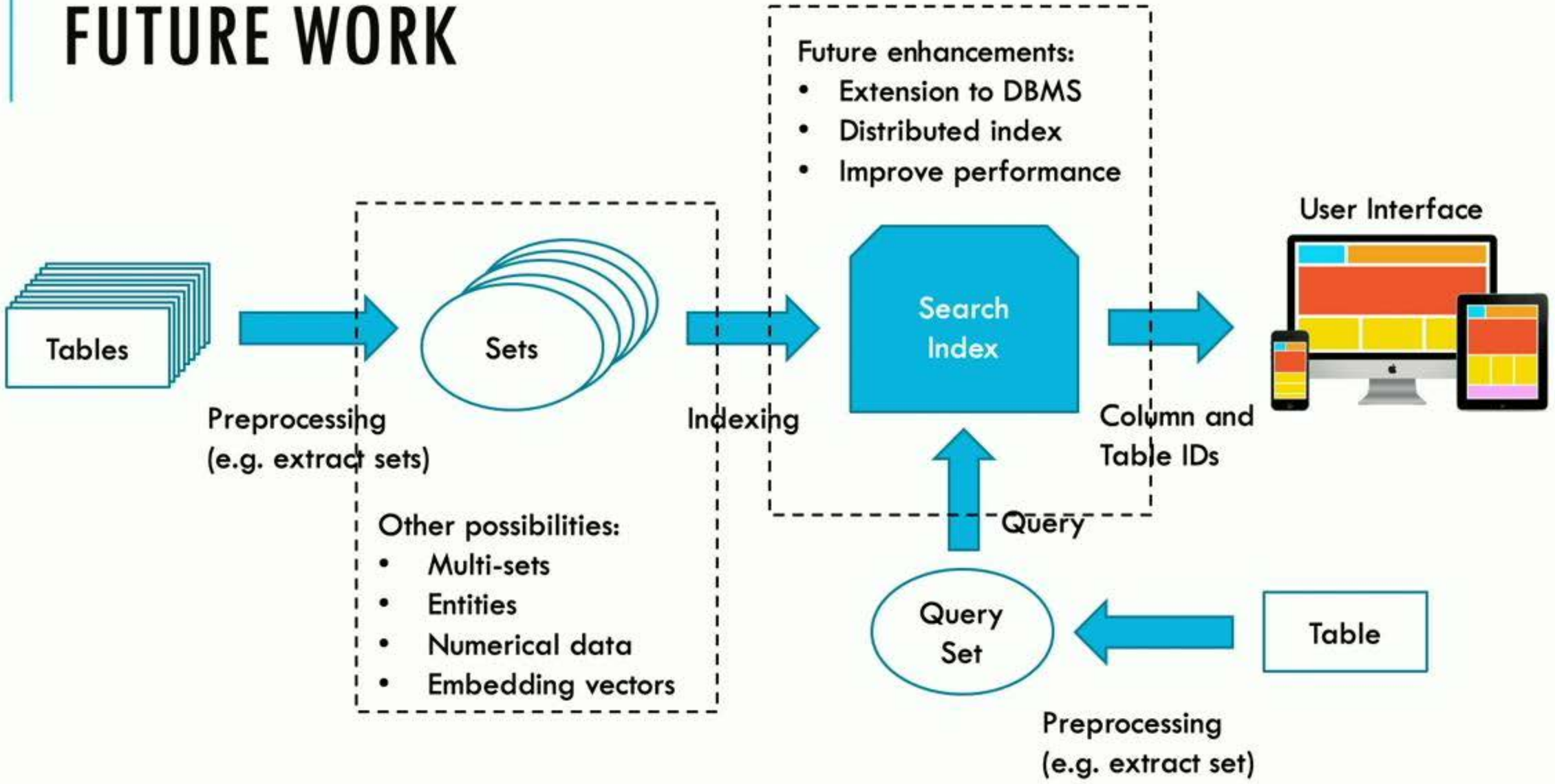
FUTURE WORK


Tables
Preprocessing
(e.g. extract sets)
Sets
Indexing
Search
Index
User Interface
Column and
Table IDs
Query
Query
Set
Table
Preprocessing
(e.g. extract set)

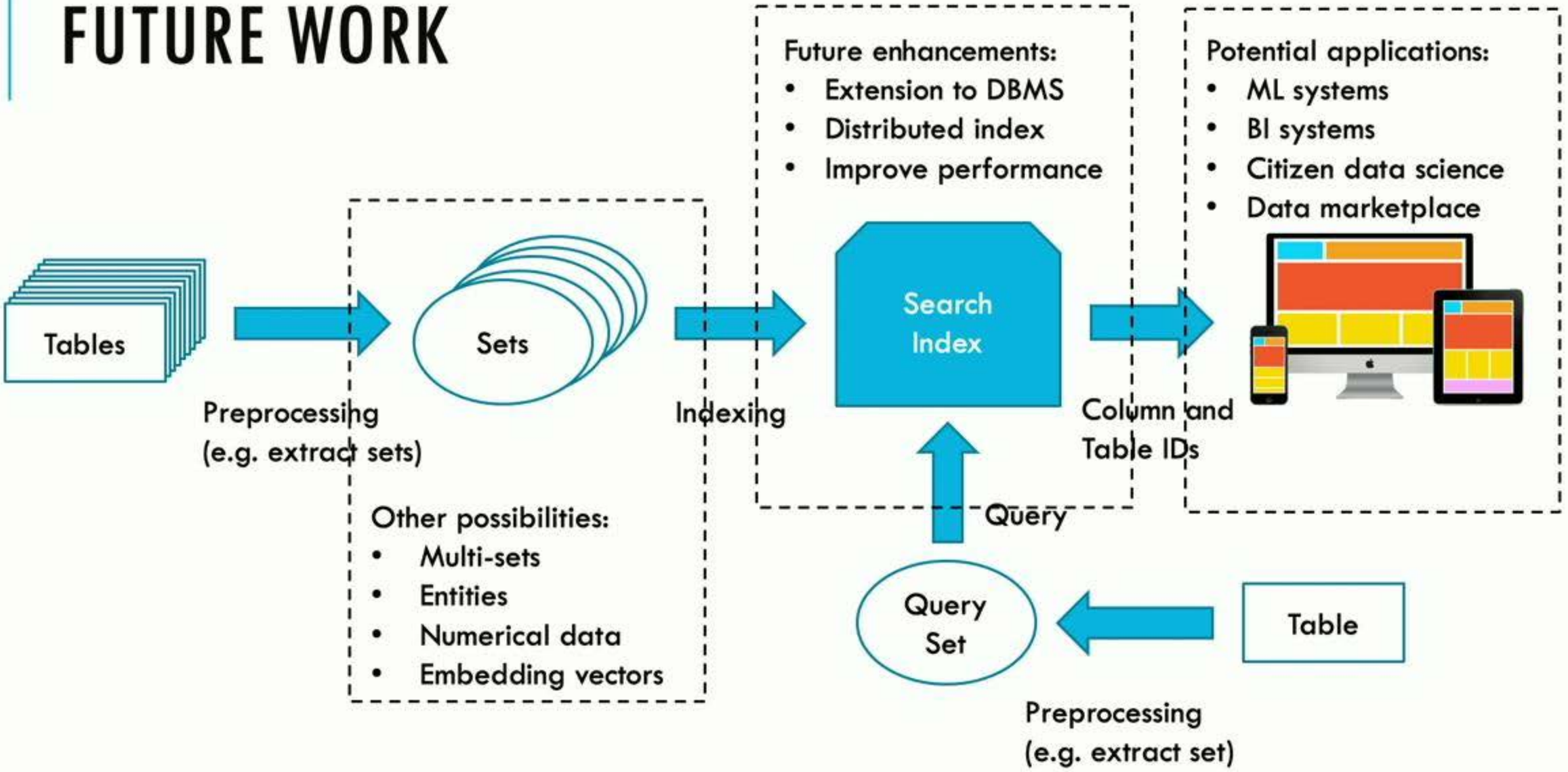
FUTURE WORK


Tables
Preprocessing
(e.g. extract sets)
Sets
Other possibilities:
• Multi-sets
• Entities
• Numerical data
• Embedding vectors
Indexing
Search
Index
Query
Query
Set
Preprocessing
(e.g. extract set)
Table
Column and
Table IDs
User Interface

FUTURE WORK


Future enhancements:
• Extension to DBMS
• Distributed index
• Improve performance
User Interface
Tables
Sets
Search Index
Preprocessing
(e.g. extract sets)
Indexing
Column and Table IDs
Other possibilities:
• Multi-sets
• Entities
• Numerical data
• Embedding vectors
Query
Query Set
Table
Preprocessing
(e.g. extract set)

FUTURE WORK


Future enhancements:
• Extension to DBMS
• Distributed index
• Improve performance
Potential applications:
• ML systems
• BI systems
• Citizen data science
• Data marketplace
Tables
Preprocessing
(e.g. extract sets)
Sets
Indexing
Search
Index
Column and
Table IDs
Other possibilities:
• Multi-sets
• Entities
• Numerical data
• Embedding vectors
Query
Query
Set
Table
Preprocessing
(e.g. extract set)