All Vector Search is Hybrid Search

John Solitario / Product Lead at Rockset

ROCKSET

Skipping to the End…

Step 1 - The Title

All vector search applications are hybrid search applications

Step 2 - Vector Search Infrastructure

All vector search applications are hybrid search applications

Vector search infrastructure is, therefore, actually hybrid search infrastructure

Step 3 - Hybrid Search Infrastructure

All vector search applications are hybrid search applications

Vector search infrastructure is, therefore, actually hybrid search infrastructure

Hybrid search infrastructure is a new kind of thing, combining vector, search, and database infrastructure

Vector Search

Vector Search: Allows us to recast many types of questions…

Show me things I might like

Show me images of cats

Show me books about volcanoes and wizards.

Recommend shows I want to watch

… as …

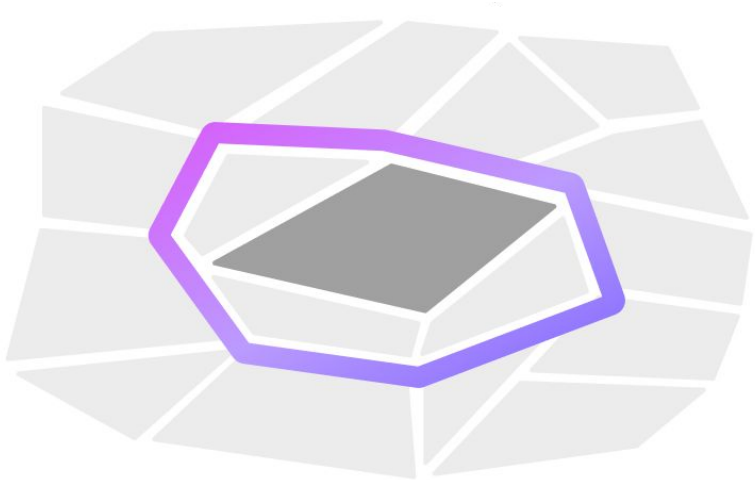
Given an input, **find the closest vectors in the data**

Vector Search: Nearest-Neighbor Algorithms

Exact nearest-neighbor (kNN) is computationally expensive

Approximate Nearest Neighbor (ANN) enables scalability

But “approximation” becomes a new source of pain...



Hybrid Search

Hybrid Search Introduction

Popular but overloaded term often used in the context of AI and vector search

Broadly: Combines 2+ distinct kinds of indexes to increase relevance

Today: **Vector Search** + **Something Else**

True™ Hybrid Search

Combines “dense” and “sparse” embeddings…

Which really means vector search and keyword boosting

Merge results with “rank fusion” or weighted average of scores

Search with Metadata Filtering

Very popular term in vector search

Show me the most relevant documents

… updated in the last 30 days

This is still hybrid search! Weights are just extremal

“matches” = +1000

“doesn’t match” = -1000

Two-Stage Retrieval + Reranking

1. Use an inexpensive index to find 100 candidates
2. Only run expensive ranker on those 100 candidates

Different form of combining two indexes as the first maximizes computational resources of the second

Combine, Mix, Match, and More

Dense / Sparse Search

Metadata Filtering

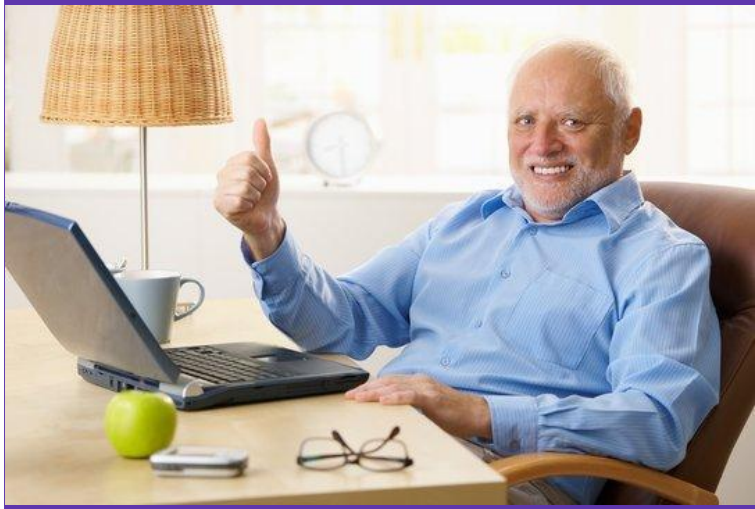
Two-Stage Retrieval

Vector Search Becomes Hybrid Search

The Dream of Vector Search

Train a model to ‘embed’ any query into a vector space of results

With the power of AI, we can run ‘nearest neighbor’ to answer any search, recommendation, or question



One Big Problem

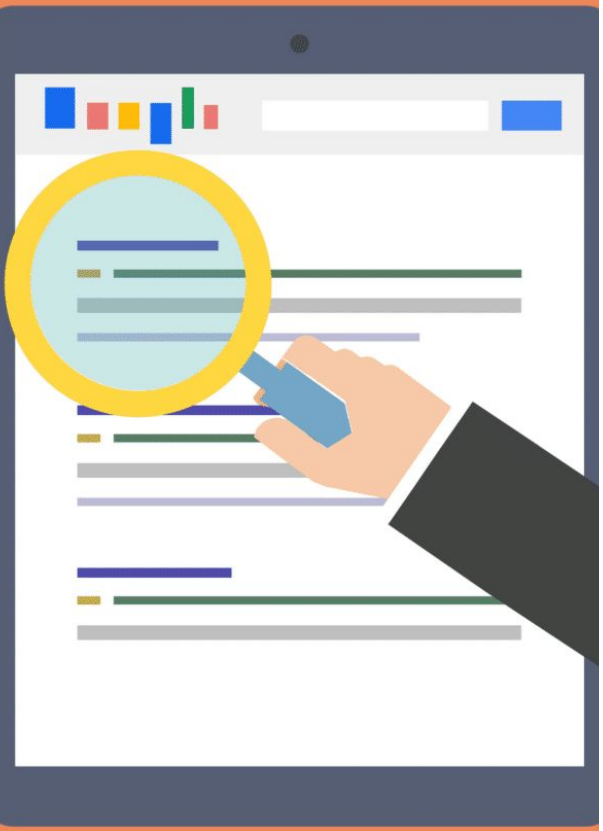
Models are insufficiently advanced

Vector searches are amazing at finding semantic likeness

… but terrible at respecting specific guardrails

ROCKSET

Any product experience powered primarily by vector search will be made better by adding filtering, rules, and other guardrails.



Example 1: In-App Search

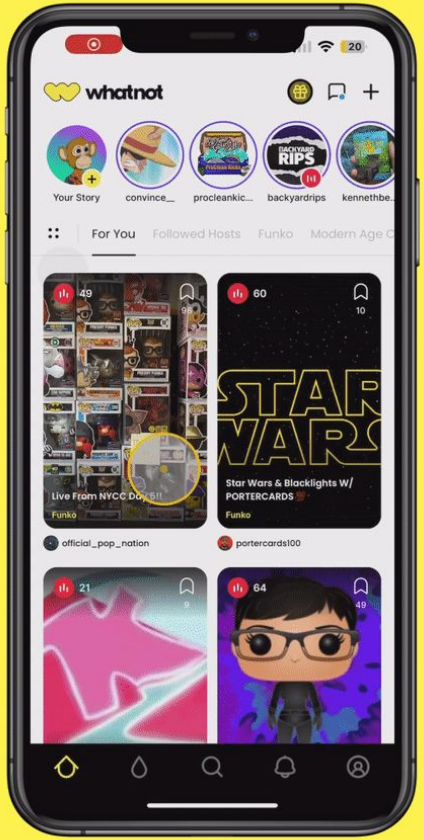
Show me **accounts** where the…

… the company name includes “Geico”

… the address contains “Virginia”

… description states “auto insurance”

… the notes mention “actively engaged”



Example 2: Recommendations

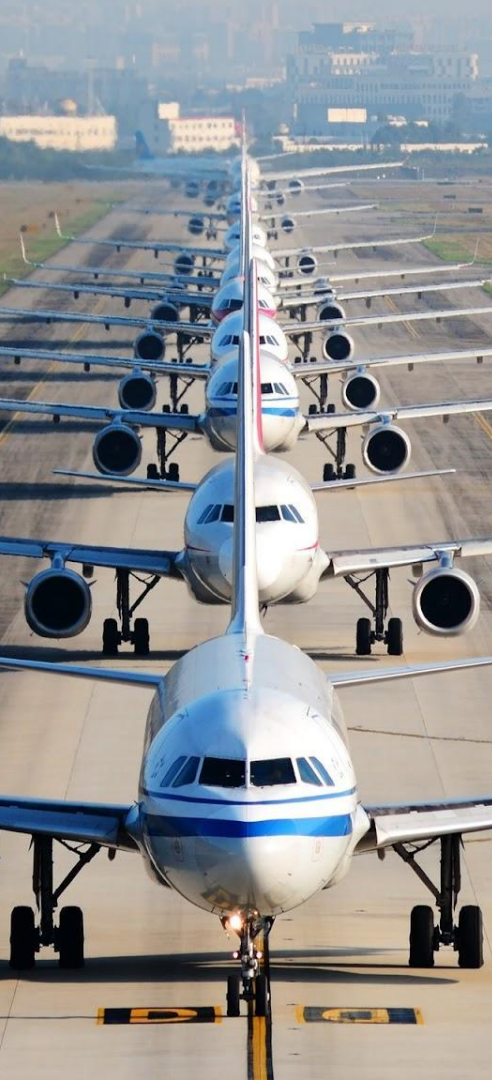
Show me **live streams** that…

… started in the past 5 minutes

… have more than 50 viewers

… are selling sports memorabilia

… were previously followed



Example 3: Retrieval Augmented Generation (RAG)

Return **documents** that…

… are associated with SFO, OAK, or SJC

… contain the phrase “boarding procedures”

… have been updated in the past 30 days

… include permissions for flight attendants

Vector Search Infra Becomes Hybrid Search Infra

Hybrid Search Infrastructure

New Problem - we're no longer talking about a single, clever vector index

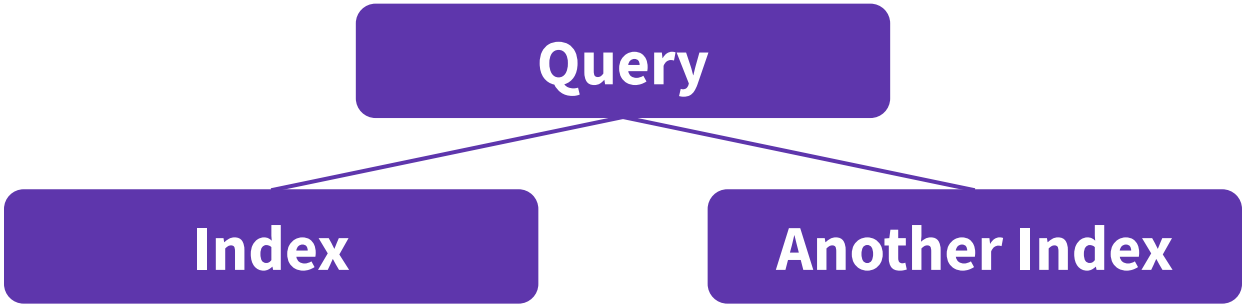
Vector search needs to cooperate with "other" indexes

Vector search infrastructure starts to look a lot like…
search and database infrastructure

Hybrid Search Infrastructure

Search and database infrastructure have been optimizing “multi-index” queries for a long time

Great ideas from both fields apply, but ANN indexes are especially difficult to integrate



A Motivating Example

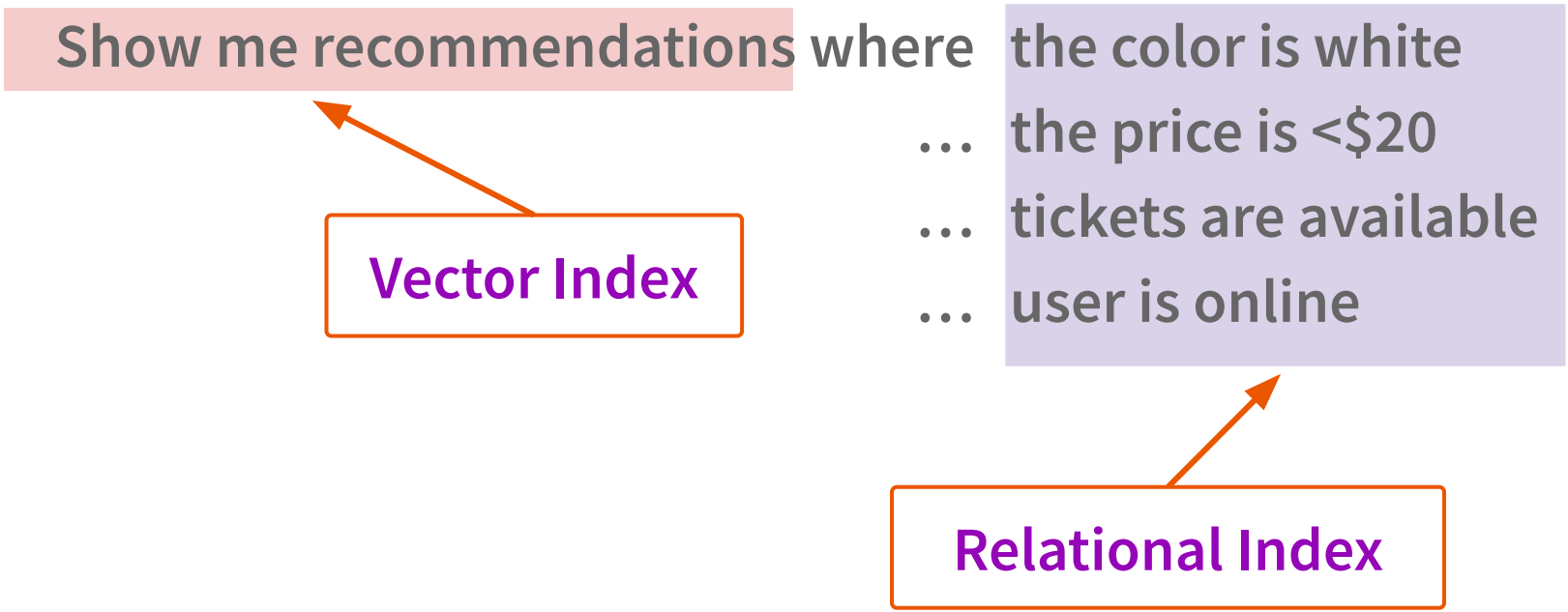
Show me recommendations where the color is white

… the price is <$20

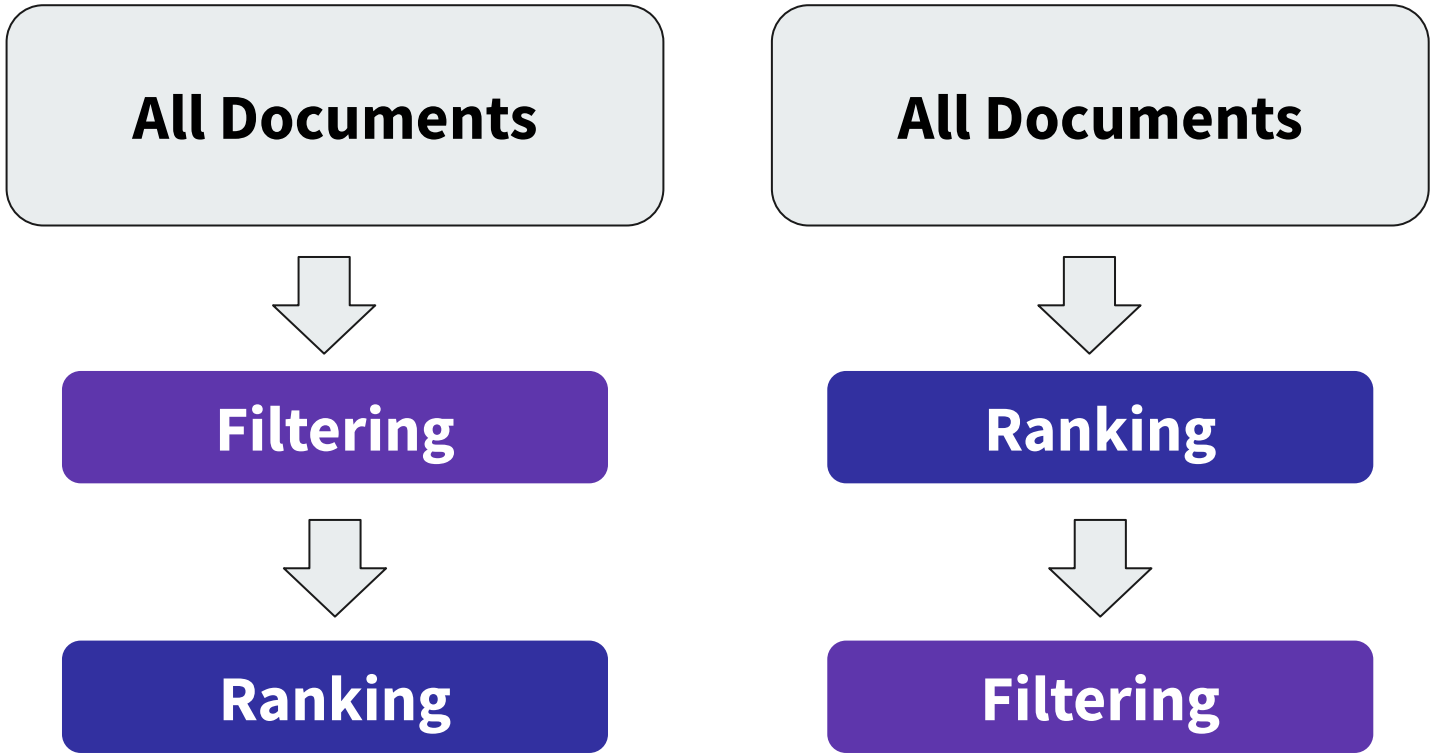
… tickets are available

… user is online

Filtering on Metadata



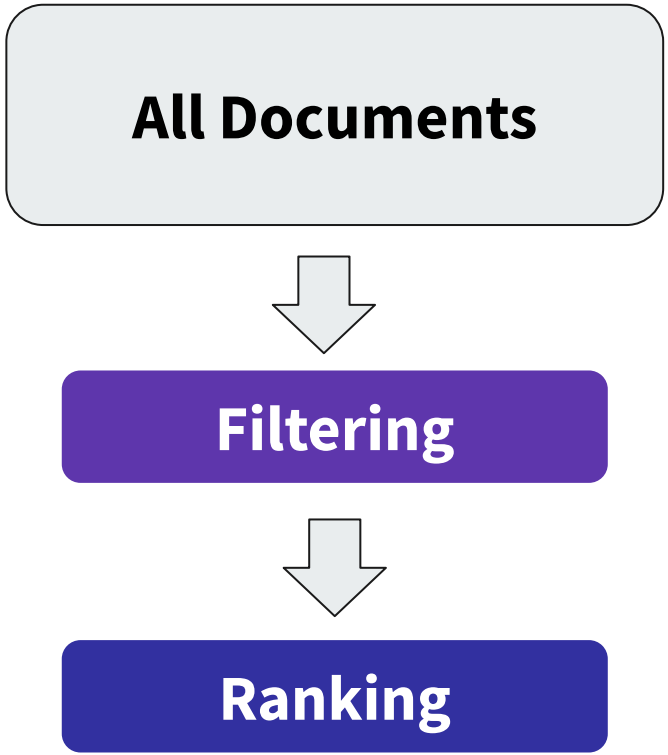
Two Obvious But Not Great Approaches



Pre-Filtering Approach

Apply the filter **before** the vector search

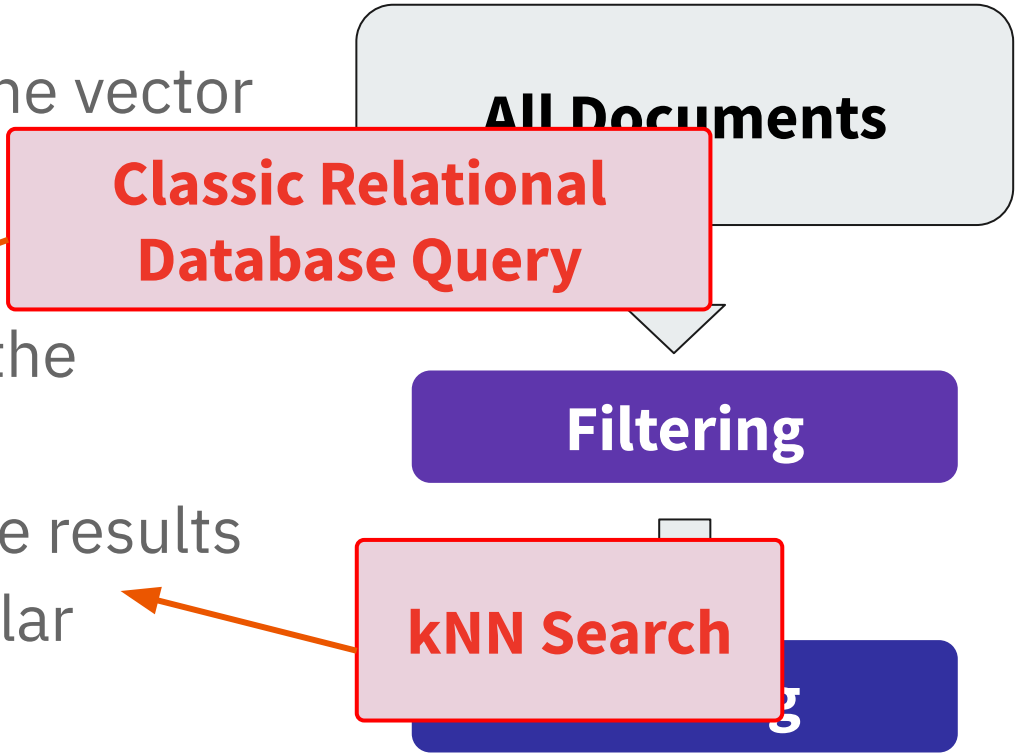
1. Find vectors that fit the metadata filter
2. Exhaustively scan the results to find the most similar



Pre-Filtering Approach

Apply the filter **before** the vector search

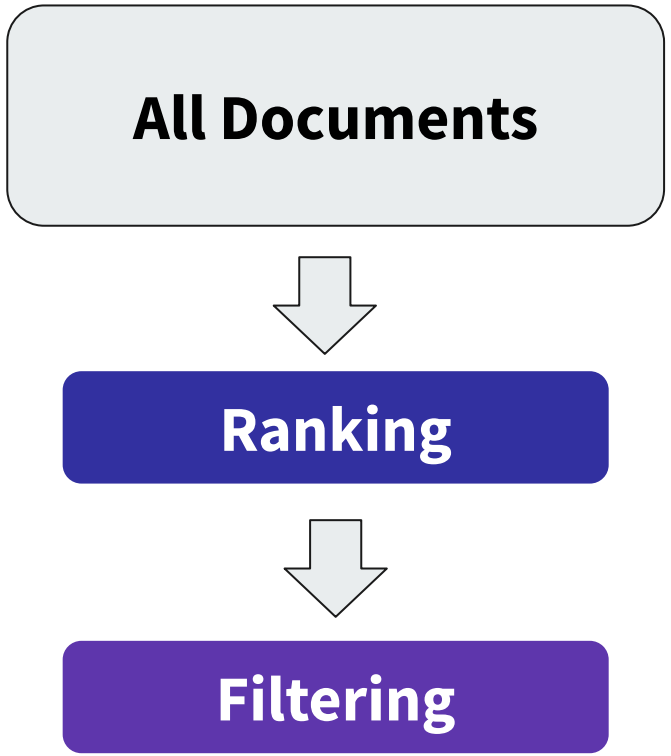
1. Find vectors that fit the metadata filter
2. Exhaustively scan the results to find the most similar



Post-Filtering Approach

Apply filtering **after** the vector search.

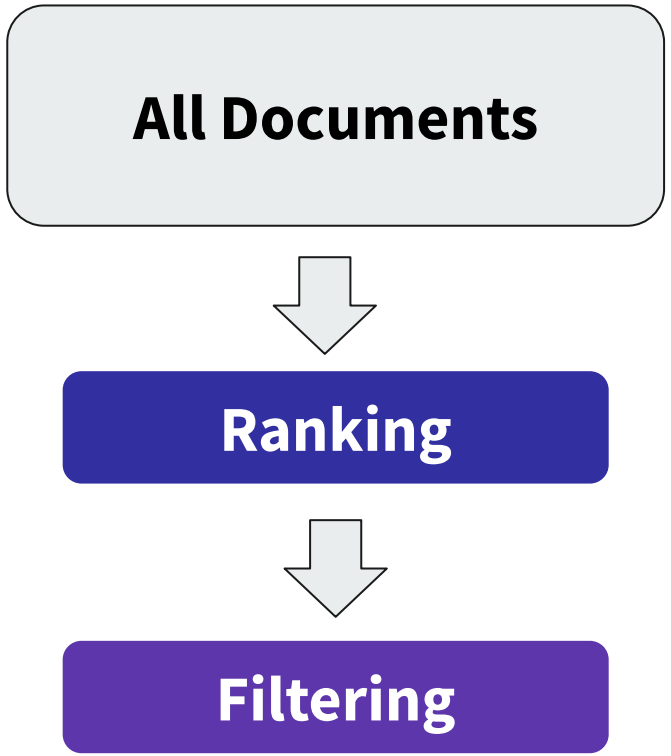
1. Find similar vectors using vector index
2. Filter out candidates that don’t match the criteria



Post-Filtering Approach

Apply filtering **after** the vector search.

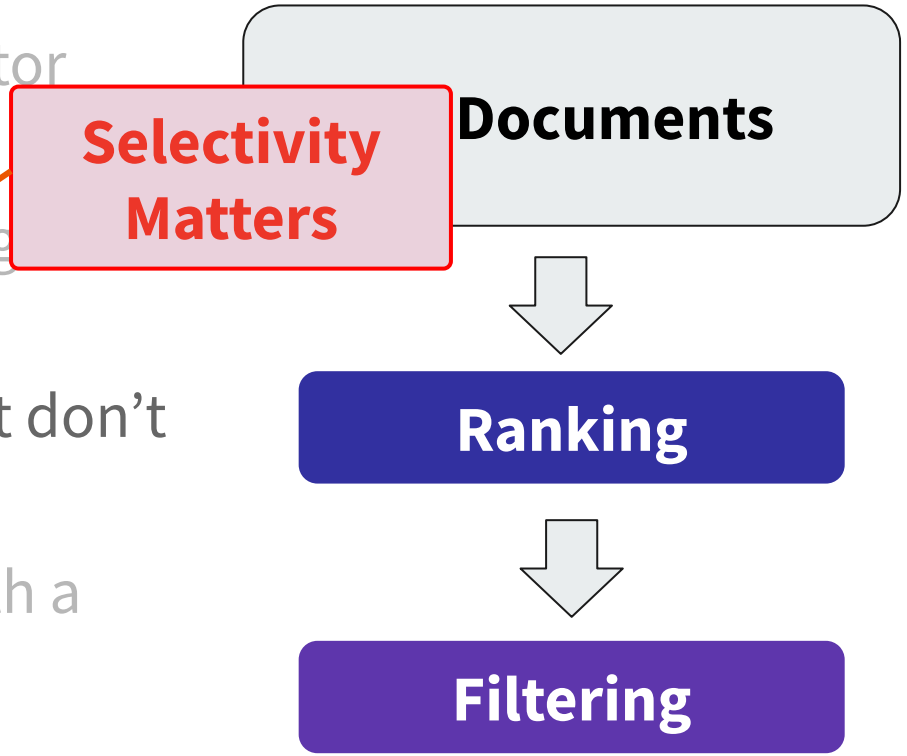
1. Find similar vectors using vector index
2. Filter out candidates that don't match the criteria
3. If too few, execute (1) with a larger threshold



Post-Filtering Approach

Apply the filter **after** the vector search.

1. Find similar vectors using index
2. Filter out candidates that don't match the criteria
3. If too few, execute (1) with a larger threshold



ROCKSET

Two Obvious But Not Great Approaches

Pre-Filtering - works great for selective filters

Post-Filtering - works great for non-selective filters

Completely Different Problem

Imagine a Database

Row Store

id	name	status	etc
0	john	active	…
1	julie	active	…
2	louis	inactive	…

Inverted Index

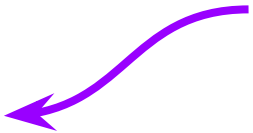
```
name:
  john
  …

status:
  active
 …
```

ROCKSET

Inverted Index

Row Store

id	name	status	etc
0	john	active	…
1	julie	active	…
2	louis	inactive	…



Inverted Index

```
name:
  john
  …

status:
  active
  …
```

ROCKSET

Fundamental Idea of Indexes

Per row accessed - scanning is much faster.

Inverted index lets us look at **far fewer** rows

This implies, though, a tipping point

Fundamental Idea of Indexes

Per row accessed - scanning is much faster.

Inverted index lets us look at **far fewer** rows

This implies, though, a tipping point

```
SELECT … where name    = “john”   AND …
SELECT … where status  = “active” AND …
```

Tipping Point

```
SELECT … where name   = “john”   AND …
SELECT … where status = “ACTIVE” AND …
```

~1 Row

~All Rows

Best Query Strategy

```
SELECT … where name   = “john”   AND …
SELECT … where status = “ACTIVE” AND …
```

Inverted Index

Scan

Selectivity

```
SELECT … where name   = “john”   AND …
SELECT … where status = “ACTIVE” AND …
```

Selective vs **Non-Selective**

So…

Deep in the query optimizer

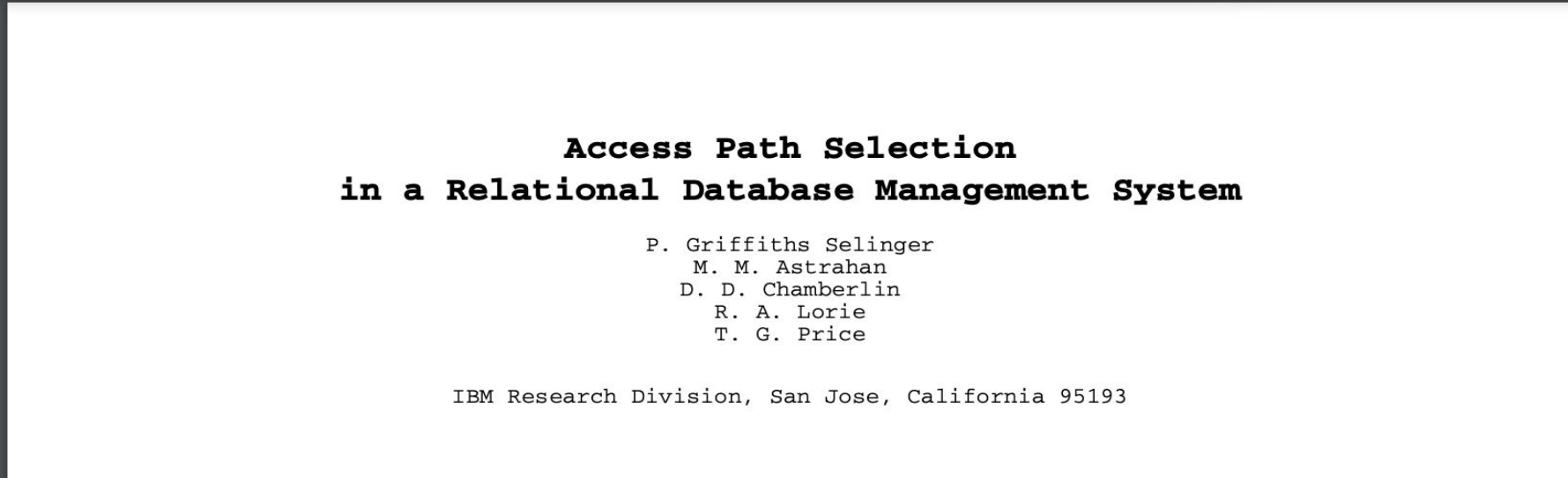
There’s two ways to do things

The right way is based on the selectivity of the filter

Sounds familiar…

Fairly Famous Paper

[1] P. G. Selinger, M. M. Astrahan, D. D. Chamberlin, R. A. Lorie, and T. G. Price, “Access path selection in a relational database management system,” in Proceedings of the 1979 ACM SIGMOD International Conference on Management of Data, ser. SIGMOD ’79. New York, NY, USA: Association for Computing Machinery, 1979, p. 23–34.



Access Path Selection
in a Relational Database Management System

P. Griffiths Selinger
M. M. Astrahan
D. D. Chamberlin
R. A. Lorie
T. G. Price

IBM Research Division, San Jose, California 95193

Selectivity Estimates for Query Optimization

40-year-old concept, which means…

We've made 40 years of progress on how to measure it, where to apply it, and so on

~2 years ago, vector metadata filtering was in 1979

Reread the last 40 years of database and search advances for ANN indexes

What does this all mean?

Vector indexes are joining the large body of research

We have a huge amount of work to do and expect rapid development… and merging these infrastructures

Vector search, search, and databases coming together

All Vector Search is Hybrid Search

ROCKSET

Thank You

John Solitario / **solitario@rockset.com**