Dissimilarity search: implementing in-memory vector search algorithms for PostgreSQL

Jonathan Katz

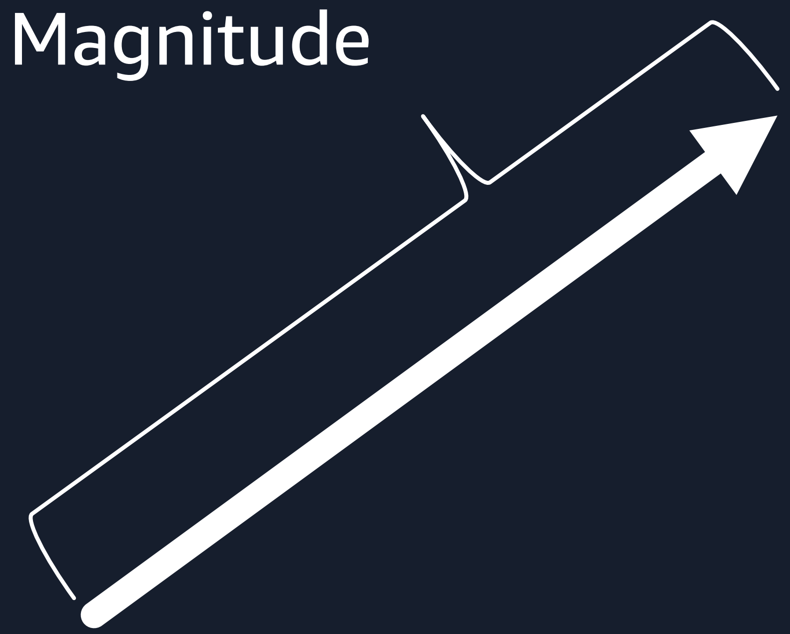
(he/him)
Principal Product Manager – Technical
AWS

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

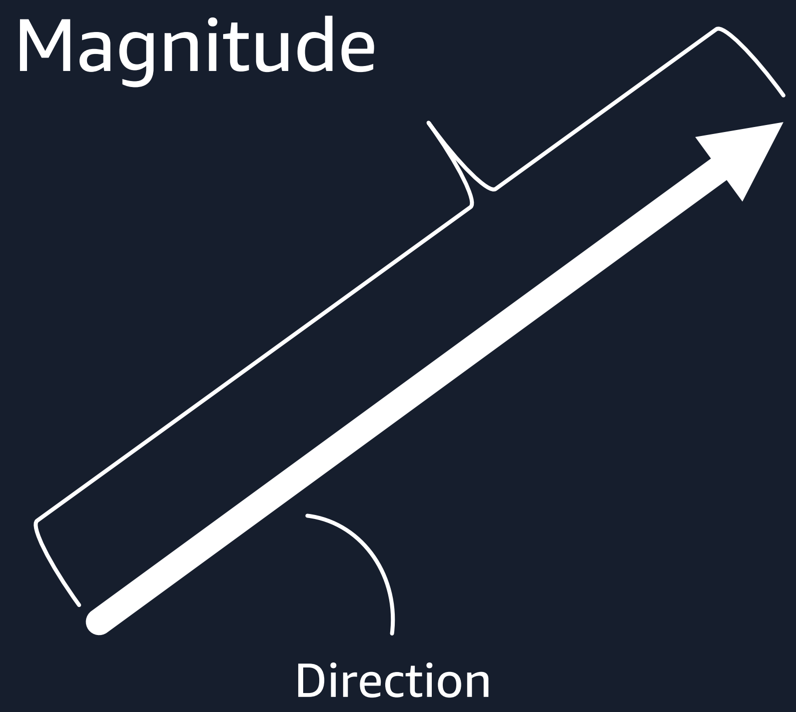
[0.5, 0.5]

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.



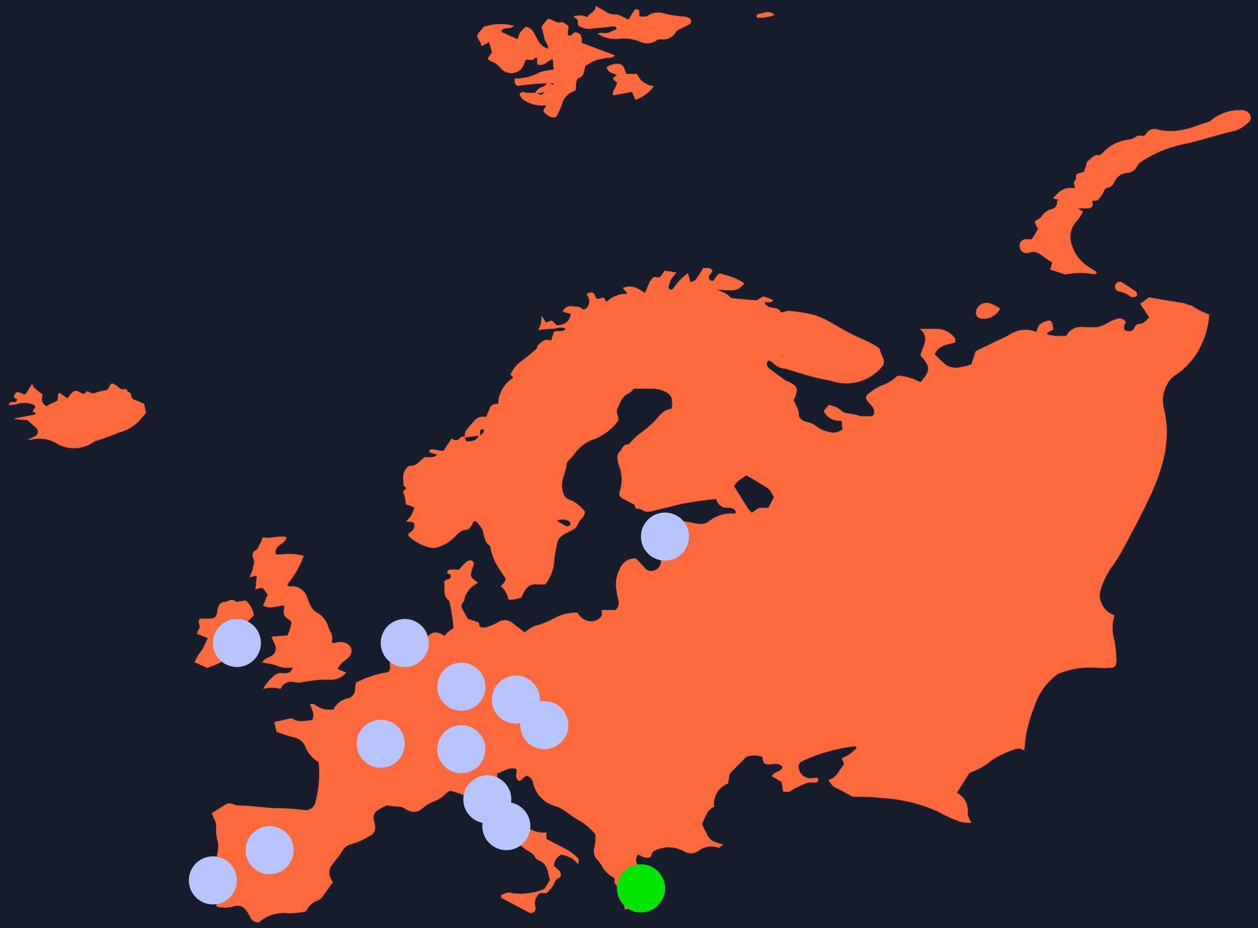
$$\| [0.5, 0.5] \| = \sqrt{(0.5^2 + 0.5^2)} = \mathbf{0.70710}$$

aws
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.



[0.5, 0.5]

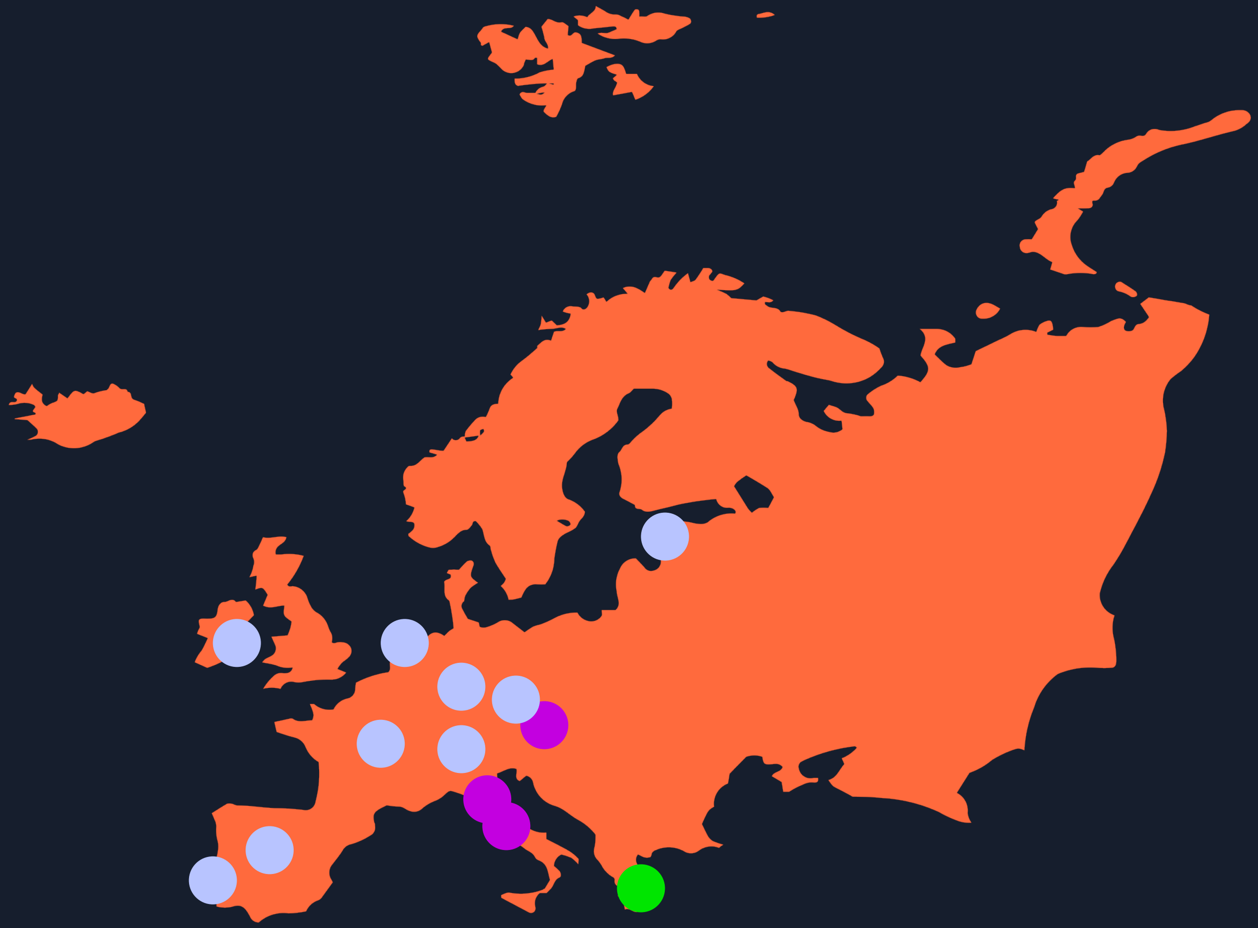
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.



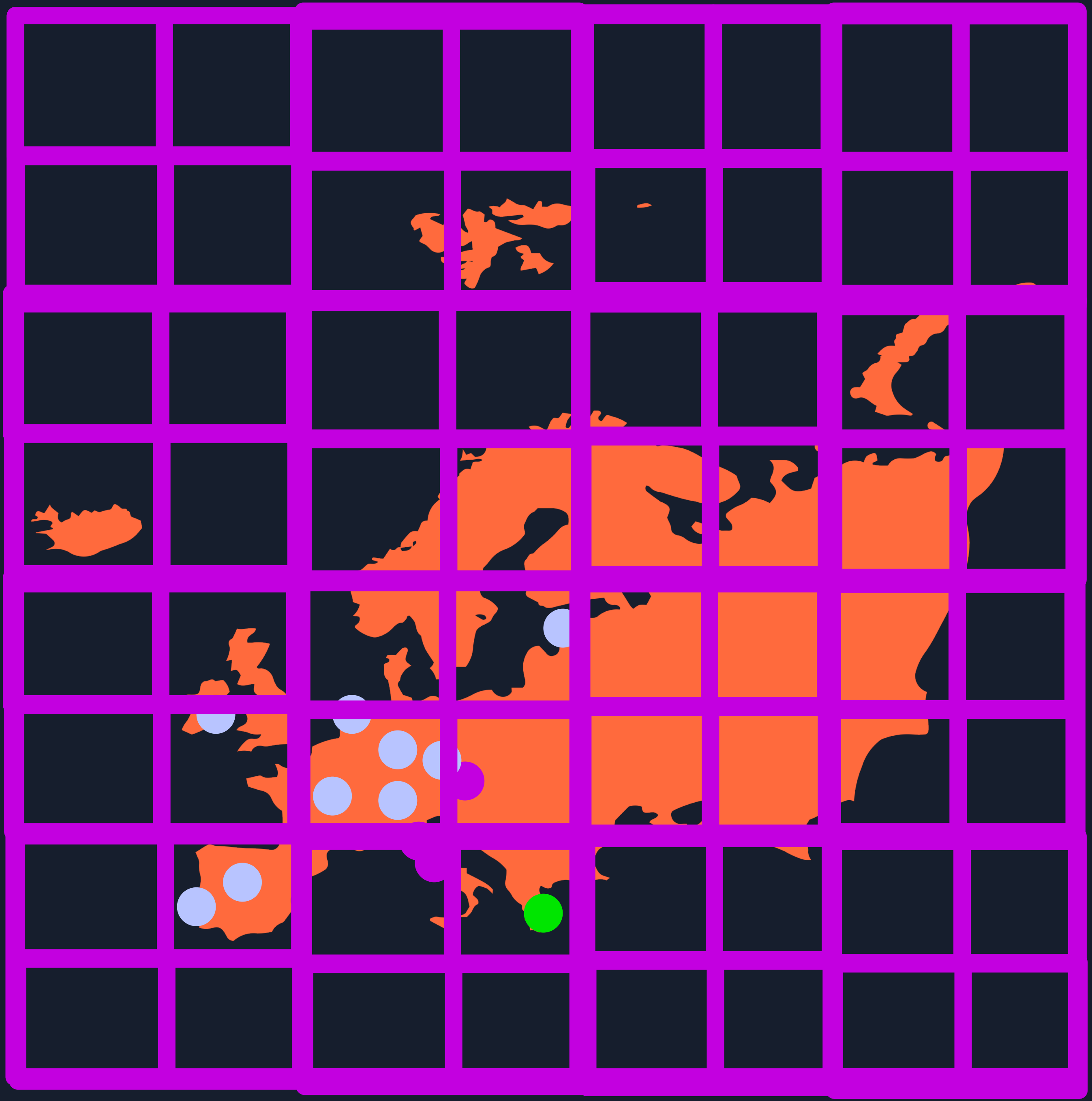
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

```sql
SELECT city_name
FROM conferences
WHERE conference_name LIKE 'PGConf EU%'
ORDER BY
  conference.geocode <-> '(38.0004,23.7195)'::point
LIMIT 3;
```

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Foundations of vector search

- Vectors in "vector space" (search area) must all have the same number of dimensions
  - Each dimension should be comparable to each other
- Distance function defines proximity
  - Distance is always ≥ 0
  - Distance from a vector to itself is 0

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

CUSTOMER

Can you find 5 distributors of green olives near PGConf.EU 2024?

DEVELOPER CREATED AGENT

Yes, here are a list of distributors based on proximity…

Tags

Human: You are an agent who manages orders and returns
executing the set of APIs in order to fulfill user input.

Emphasis (capitalized)

Valid "api" values are GetOrderHistory::GetProductCatalogue,
- DO NOT return an api if all required parameter values are not
- DO NOT replace the placeholders in the api_name with
- Return available parameters in api_inputs ONLY.

Valid "verb" is HTTP verb used in "APIs" e.g. GET, PUT etc

Valid "api_input" as json from "User Input", "Observation" or
- NEVER assume value for any parameter, mark the value as

Convergence criteria

DO NOT go into a loop and return exact same apis with

Format (JSON)

Provide only ONE action per $JSON_BLOB, as shown:

{ "api": $API_NAME, "verb": $HTTP_VERB, "api_input": {

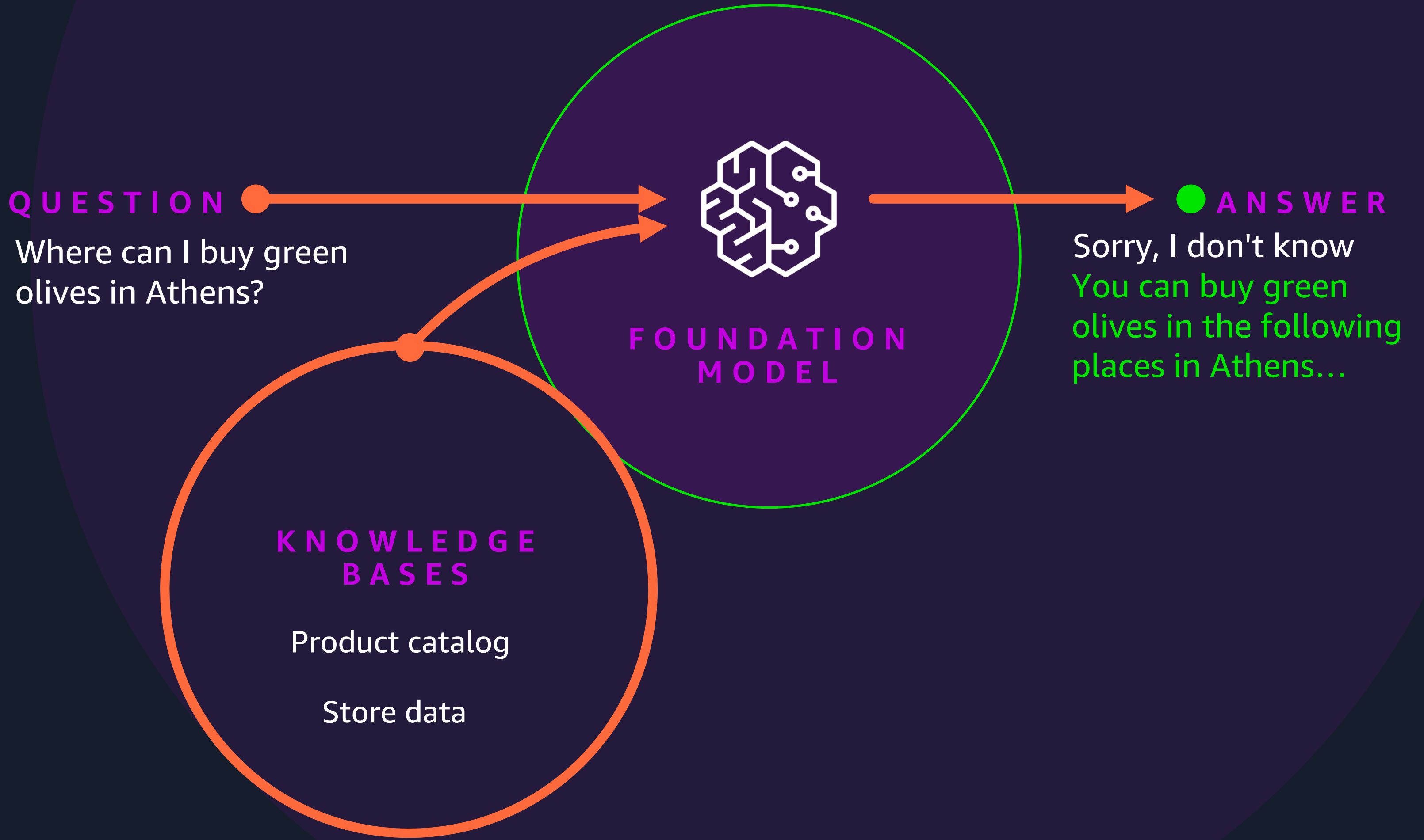
History format

Conversation History: Below is the history of the conversation

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Retrieval-augmented generation (RAG)

Configure foundation model to interact with your data



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

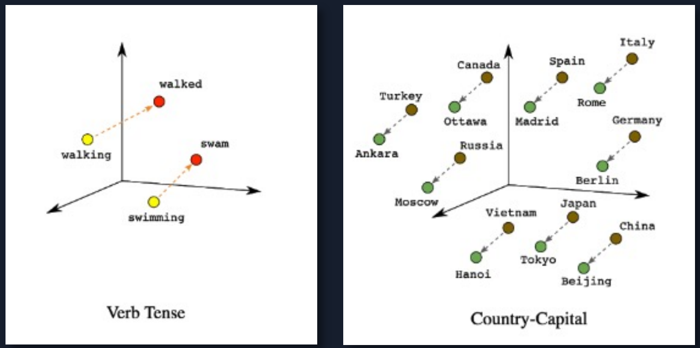
What are embeddings?

Source domain-specific data → Tokenization → Vectorization → Store in vector data store → Perform semantic similarity search → Include semantically similar context in prompt

- Documents
- Audio/video
- Images

Semantic elements:
- Words, phrases
- Paragraphs, documents
- Scenes, song sections
- Faces, detected picture elements
- And more

0.35 0.1 0 0.9 001.0 00 0001.0 0 0…
0.35 0.1 0 0.8 001.0 00 0001.0 0 0…
0.15 0.1 0 0.7 001.0 00 0001.0 0 0…

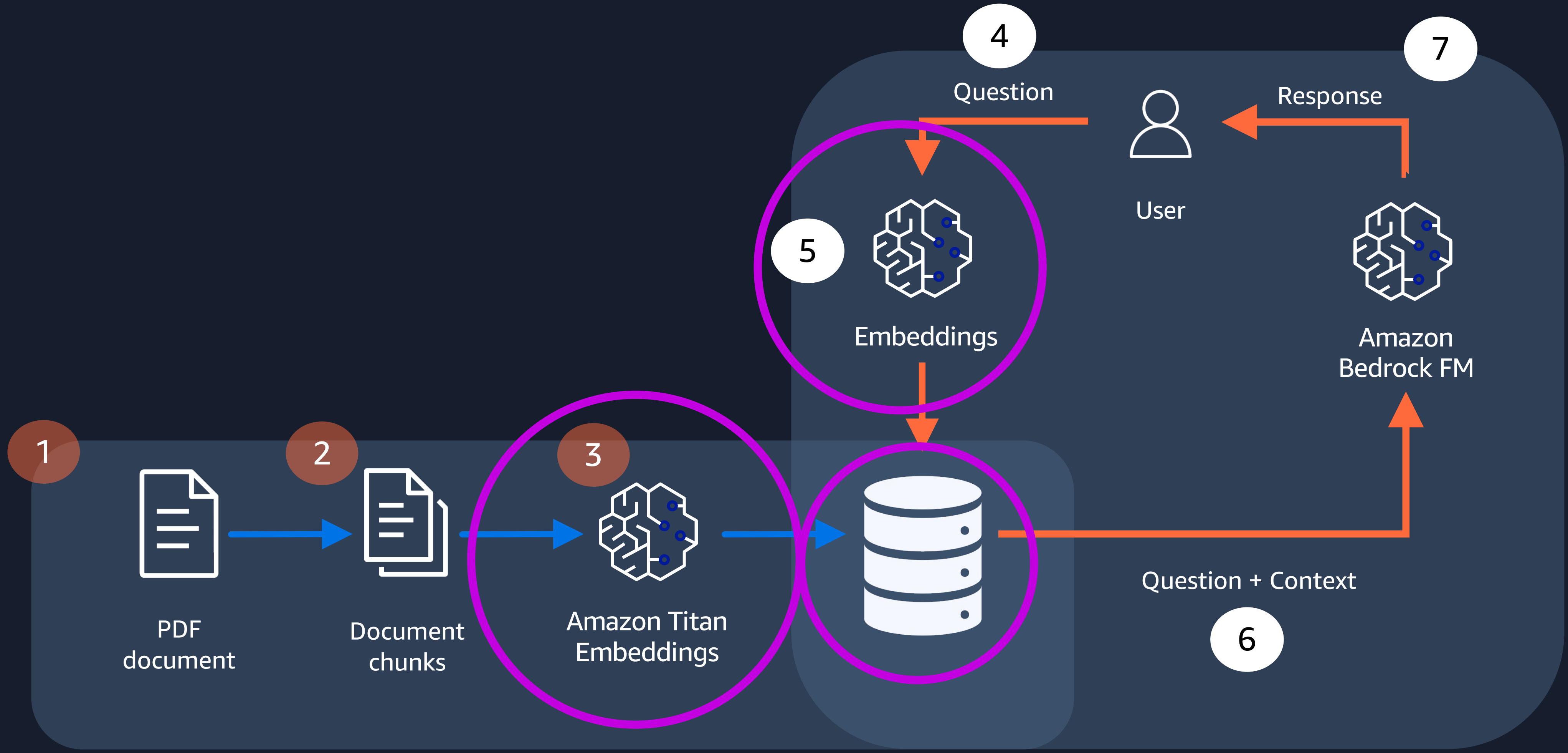


3D simplified representation. Embeddings can have thousands of dimensions. Source: https://daleonai.com/embeddings-explained

Embeddings: When vector elements are semantic, used in generative AI

aws
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

How embeddings are used



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Challenges with larger vectors

- Generation time
- Size
- ~~Compression~~
- Query time

1536 dimensions

4-byte floats

6152B => 6KiB

1,000,000 => 5.7GB

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

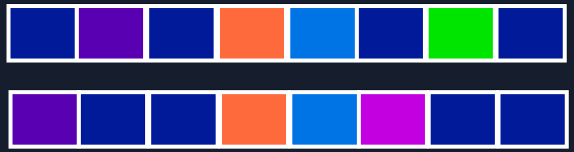
Approximate nearest neighbor (ANN)

- Find similar vectors without searching all of them
- Faster than exact nearest neighbor
- “Recall” – % of expected results

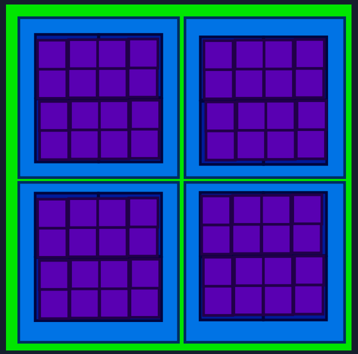
Recall: 80%

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

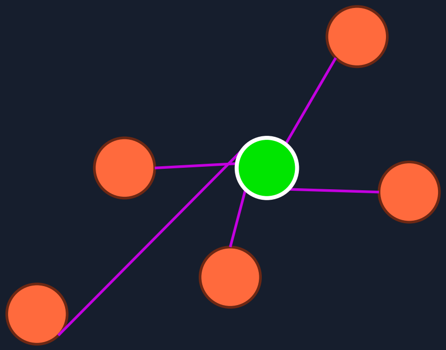
ANN indexing algorithm types



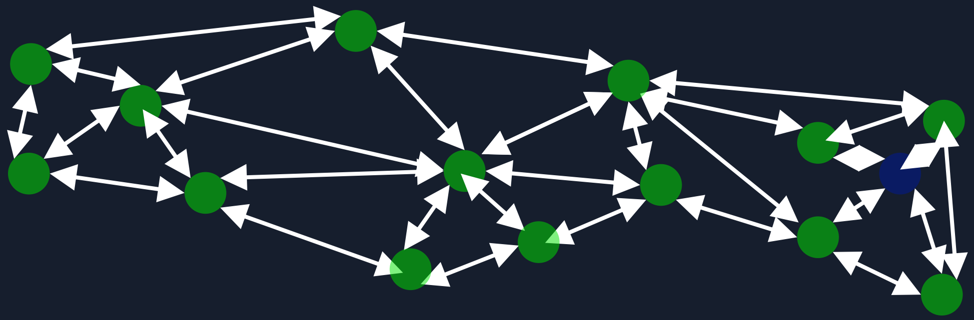
Hash



Tree



Cluster



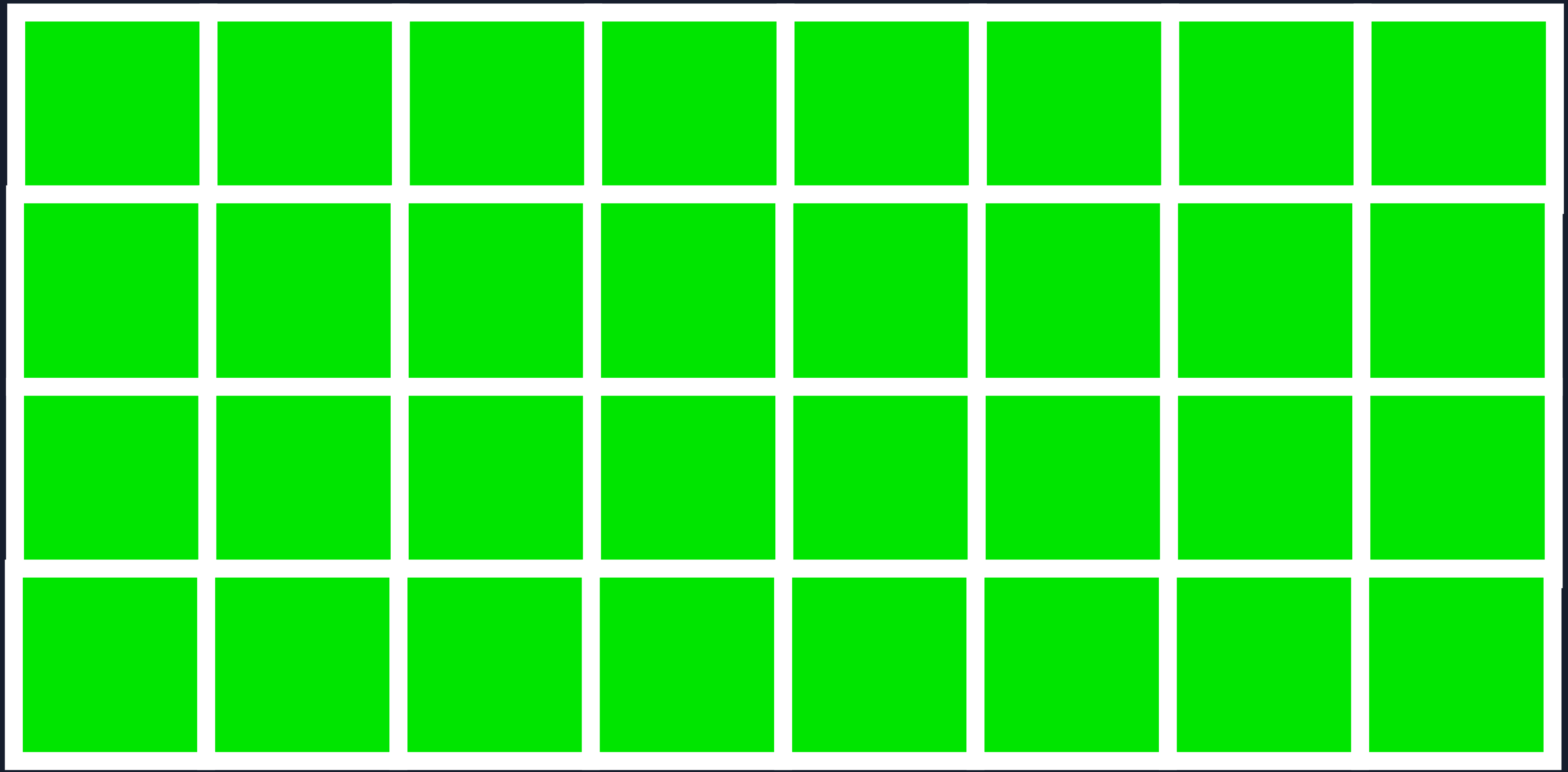
Graph

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Index layout in memory

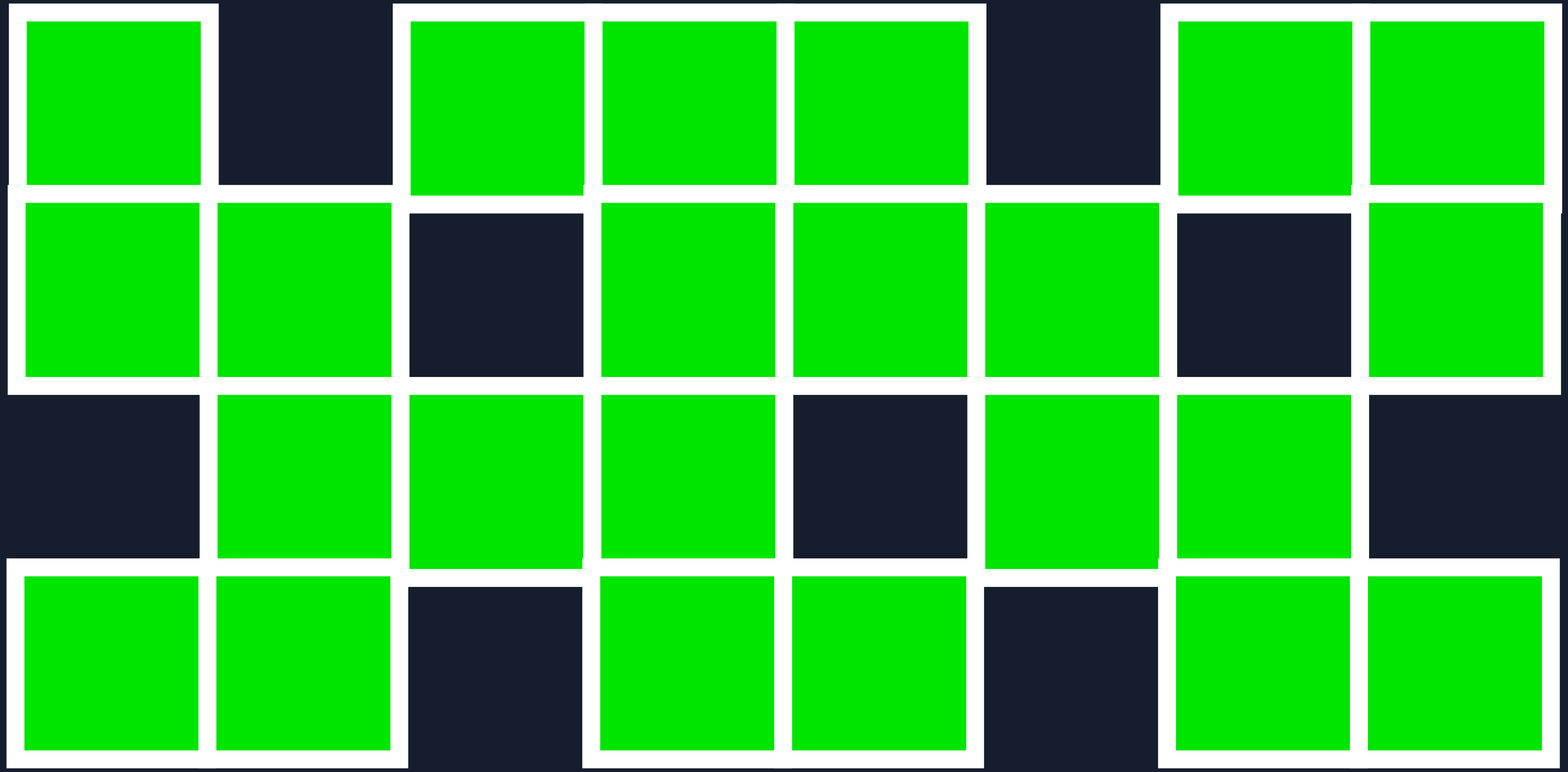
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Index layout in a database



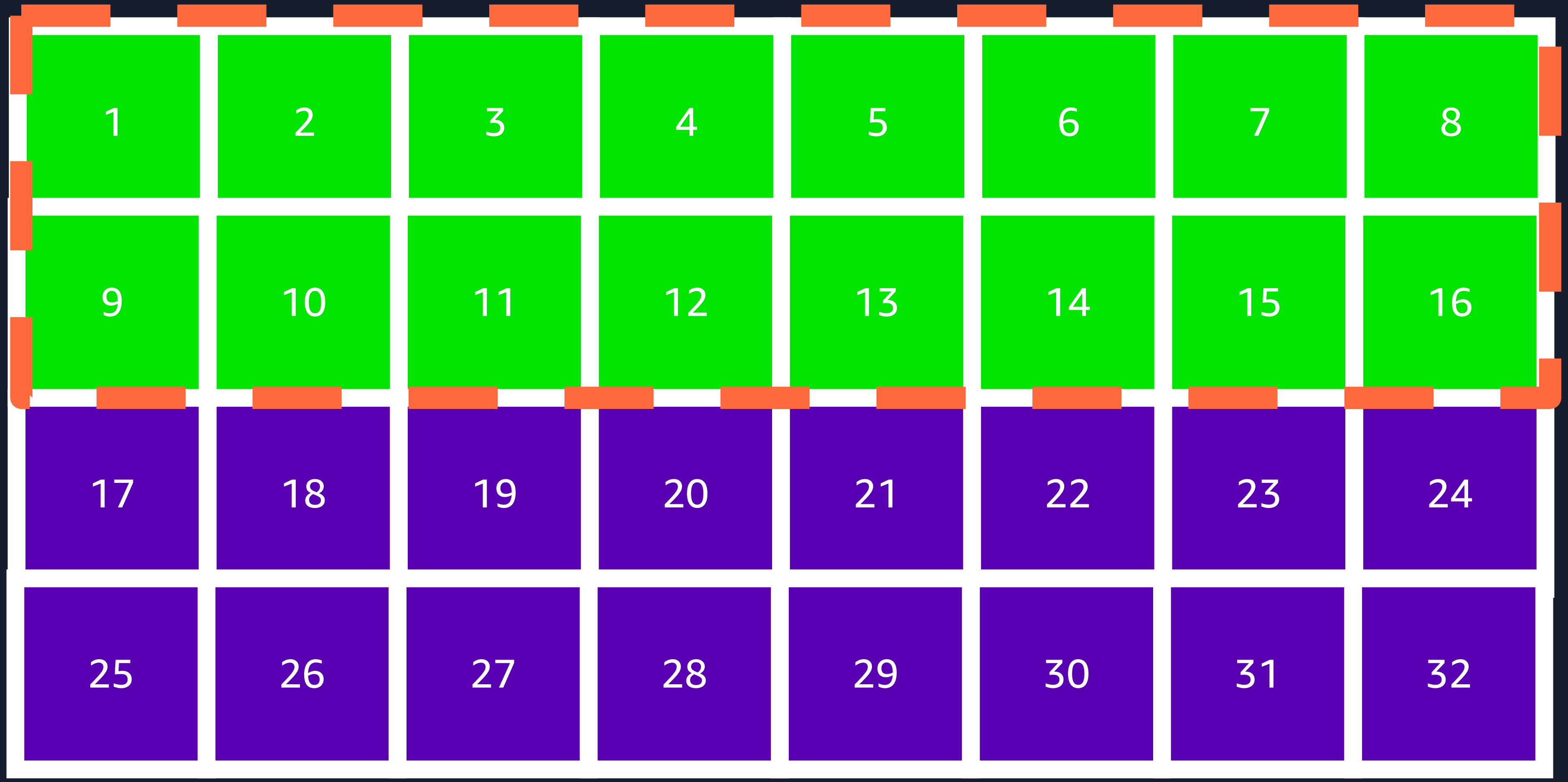
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Index layout in a database



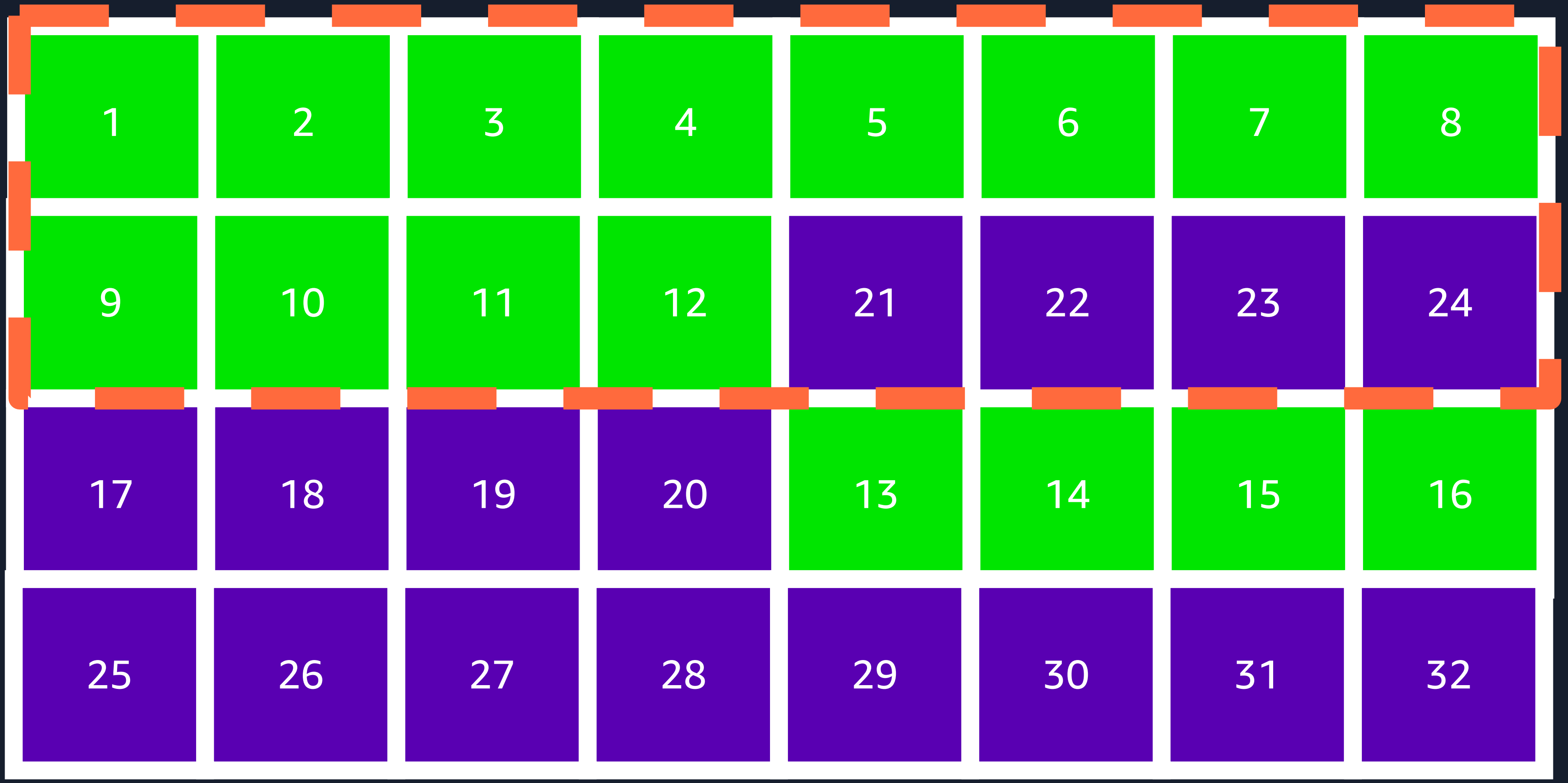
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Index size exceeds available memory



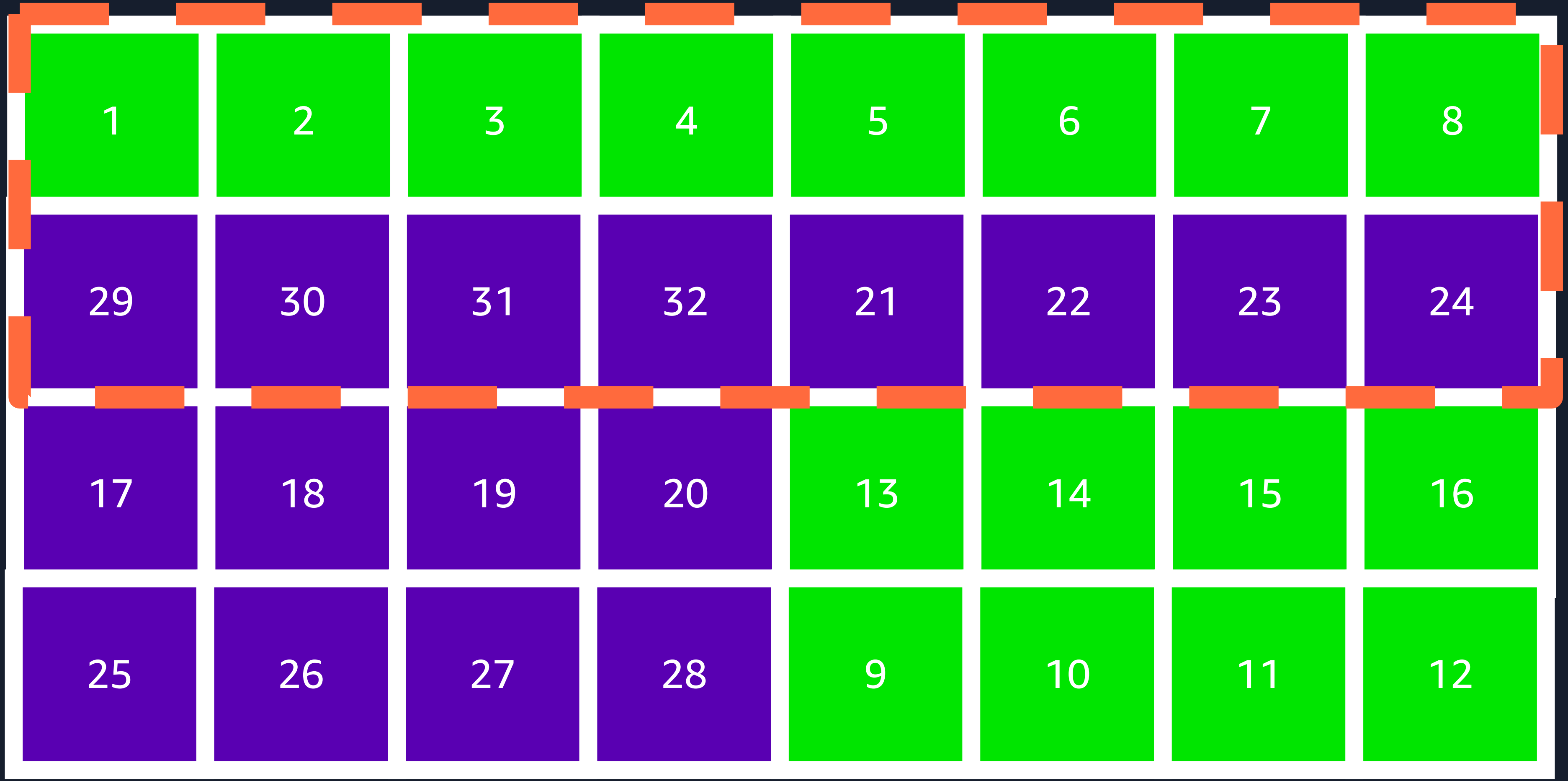
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Index size exceeds available memory



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Index size exceeds available memory



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Where memory and storage diverge

- Continuous allocations vs. pages
- Data layout on disk
- Percentage of index in memory
- Hardware acceleration strategies (CPU vs. GPU)

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Vector search design principles for PostgreSQL

- Take shortcuts, where applicable
- Design for 8KiB blocks (page size)
- Leverage PostgreSQL infrastructure
- Understand your tradeoffs

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

What is pgvector?

Adds support for storage, indexing, searching, metadata with choice of distance

`vector` data type

Co-locate with embeddings

Exact nearest neighbor (K-NN)
Approximate nearest neighbor (ANN)

Supports HNSW & IVFFlat indexing, with options for scalar and binary quantization

Distance operations include
Cosine, Euclidean/L2, Manhattan/L1, Dot product, Hamming, Jaccard

github.com/pgvector/pgvector

aws © 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Example pgvector query

```
SET hnsw.ef_search TO 60;

SELECT id, text_chunk
FROM documents
ORDER BY
    embedding <=> '[0.003421, -0.23053, 0.402153, …]'::vector
LIMIT 10
```

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

What do we need to define?

1. Data type
2. Distance functions and operators
3. Indexing strategy

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

What do we need to define?

1. Data type
2. Distance functions and operators
3. Indexing strategy

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Data type

```
typedef struct Vector
{
        int32           vl_len_;        /* varlena header (do not touch directly!) */
        int16           dim;            /* number of dimensions */
        int16           unused;         /* reserved for future use, always zero */
        float           x[FLEXIBLE_ARRAY_MEMBER];
} Vector;
```

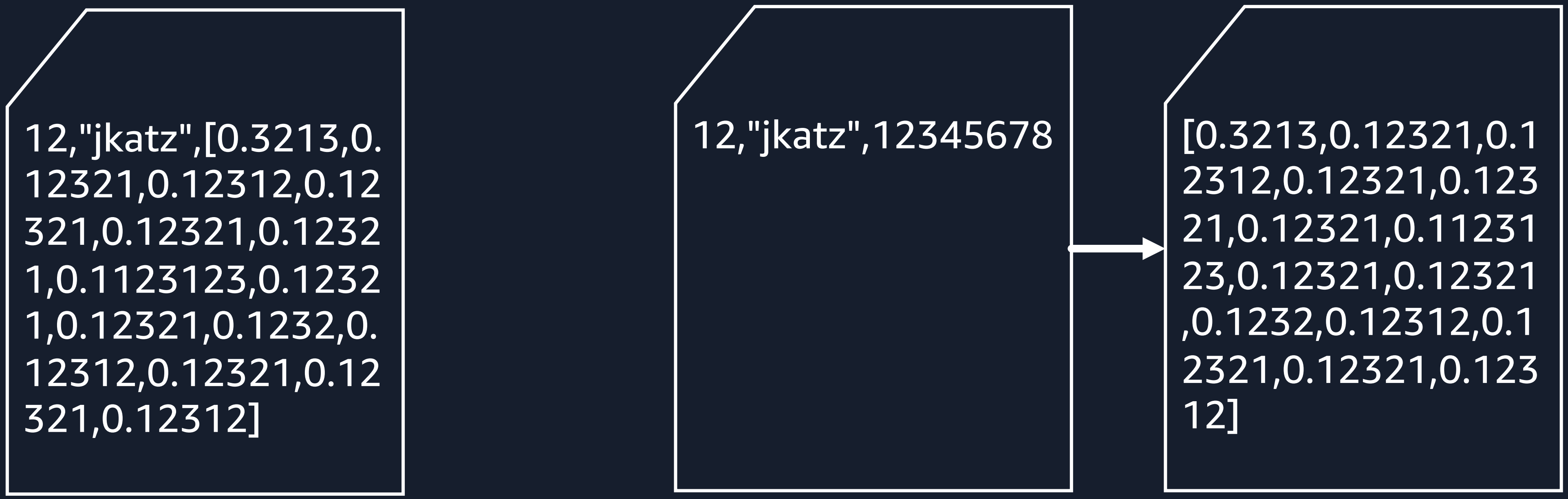
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

🐘PostgreSQL Infrastructure: 🍞 TOAST

- TOAST (<u>T</u>he <u>O</u>versized-<u>A</u>ttribute <u>S</u>torage <u>T</u>echnique) is a mechanism for storing data larger than 8KB
  - By default, PostgreSQL "TOASTs" values over 2KB (510d 4-byte float)
- Storage types:
  - `PLAIN`: Data stored inline with table
  - `EXTENDED`: Data stored/compressed in TOAST table when threshold exceeded
    - pgvector default before 0.6.0
  - <u>`EXTERNAL`</u>: Data stored in TOAST table when threshold exceeded
    - pgvector default 0.6.0+
  - `MAIN`: Data stored compressed inline with table

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Visualizing TOAST for pgvector



PLAIN

EXTENDED / EXTERNAL

aws © 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

⇆ Tradeoffs: Impact of TOAST on vector data

- Traditionally, TOAST data is not on the "hot path"
  - Impacts query plan and maintenance operations

- Compression is ineffective

- **Unable to use for index pages**

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Space utilization on a page

Dimensions	Vectors / Page	Wasted Space (B)
128	15	308
256	7	916
384	5	428
512	3	1,988
768	2	2,000
1,024	1	4,060
1,536	1	2,012
2,000	1	156

```
PAGE_SIZE - PAGE_HEADER - (VECTORS * 4) - VECTORS * (4 * DIMS + 8)
```

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

What do we need to define?

1. Data type
2. Distance functions and operators
3. Indexing strategy

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Distance functions / operators

Euclidean / L2	<->
Cosine	<=>
Inner Product	<#>
Manhattan / Taxicab / L1	<+>
Hamming	<~>
Jaccard	<%>

```
CREATE FUNCTION
  l2_distance(vector, vector)
RETURNS float8
  AS 'MODULE_PATHNAME'
  LANGUAGE C
  IMMUTABLE STRICT
  PARALLEL SAFE;
```

```
CREATE FUNCTION
  cosine_distance(vector,
vector) RETURNS float8
  AS 'MODULE_PATHNAME'
  LANGUAGE C
  IMMUTABLE STRICT
  PARALLEL SAFE;
```

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

PostgreSQL Infrastructure: Function definitions

```
FUNCTION_PREFIX PG_FUNCTION_INFO_V1(l2_distance);

Datum

l2_distance(PG_FUNCTION_ARGS)

{

	Vector	*a = PG_GETARG_VECTOR_P(0);

	Vector	*b = PG_GETARG_VECTOR_P(1);


	CheckDims(a, b);


	PG_RETURN_FLOAT8(sqrt((double)

		VectorL2SquaredDistance(a->dim, a->x, b->x)));

}
```

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

🚀 Shortcut: SIMD using compiler autovectorization

```
VectorCosineSimilarity(int dim, float *ax, float *bx)
{
        /* ... */
        /* Auto-vectorized */
        for (int i = 0; i < dim; i++)
        {
                similarity += ax[i] * bx[i];
                norma += ax[i] * ax[i];
                normb += bx[i] * bx[i];
        }
        /* Use sqrt(a * b) over sqrt(a) * sqrt(b) */
        return (double) similarity / sqrt((double) norma * (double) normb);
}
```

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

🚀 Shortcut: CPU dispatching (AVX-512)

```
TARGET_AVX512_POPCOUNT static uint64
BitHammingDistanceAvx512Popcount(uint32 bytes, unsigned char *ax, unsigned char *bx, uint64 distance)
{
        __m512i         dist = _mm512_setzero_si512();
        for (; bytes >= sizeof(__m512i); bytes -= sizeof(__m512i))
        {
                __m512i         axs = _mm512_loadu_si512((const __m512i *) ax);
                __m512i         bxs = _mm512_loadu_si512((const __m512i *) bx);
                dist = _mm512_add_epi64(dist, _mm512_popcnt_epi64(_mm512_xor_si512(axs, bxs)));
                ax += sizeof(__m512i);
                bx += sizeof(__m512i);
        }

        distance += _mm512_reduce_add_epi64(dist);
        return BitHammingDistanceDefault(bytes, ax, bx, distance);
```

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

What do we need to define?

1. Data type
2. Distance functions and operators
3. Indexing strategy

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

PostgreSQL index interfaces

- GiST (Generalized Search Tree)
  - Supports K-NN queries
- SP-GiST (Space-partitioned Generalized Search Tree)
  - Supports K-NN queries
- GIN (Generalized Inverted Index)
- BRIN (Block Range Index)
- B-tree
- Hash

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Example: Interfacing with GiST

- `consistent`
- `union`
- `penalty`
- `picksplit`
- `same`
- `compress`
- `decompress`
- `distance`
- `fetch`

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

⇆ Index access methods ("custom indexes")

- Let you define indexes that don't fit existing interfaces
  - Properties
  - Methods
- "More work"
  - Responsible for vacuum, WAL, locking, planning, et al.
  - (More) responsible for impact due to upstream changes

aws

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Key index access method properties for pgvector

- `amcanorder => false`
- `amcanorderbyop => true`
- `amcanbuildparallel => true`

Reference: https://www.postgresql.org/docs/current/index-api.html

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Key index access method functions for pgvector

- `ambuild`
- `aminsert`
- `ambulkdelete`
- `amcostestimate`
- `ambeginscan`
- `amrescan`
- `amgettuple`

Reference: https://www.postgresql.org/docs/current/index-functions.html

aws
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Key index access method functions for pgvector

- `ambuild`
- `aminsert`

Reference: https://www.postgresql.org/docs/current/index-functions.html

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Key index access method functions for pgvector

- `ambulkdelete`

Reference: https://www.postgresql.org/docs/current/index-functions.html

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Key index access method functions for pgvector

- `amcostestimate`

Reference: https://www.postgresql.org/docs/current/index-functions.html

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Key index access method functions for pgvector

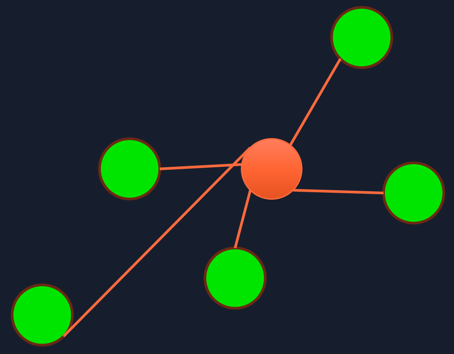
- `ambeginscan`
- `amrescan`
- `amgettuple`

Reference: https://www.postgresql.org/docs/current/index-functions.html

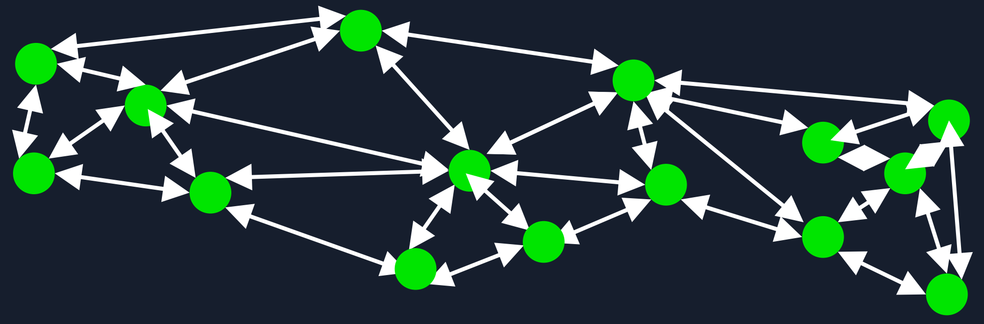
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

pgvector index methods: IVFFlat and HNSW

- IVFFlat
  - K-means based
  - Organize vectors into lists
  - Requires prepopulated data
  - Insert time bounded by # lists



- HNSW
  - Graph based
  - Organize vectors into "neighborhoods"
  - Iterative insertions
  - Insertion time increases as data in graph increases



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

🚀 Shortcut: Store normalized vectors in index

L2 normalization = **v** / || **v** ||

0.0234
0.093
-0.9123
0.1055

Valid?

✅ Same dimensions?
✅ Magnitude > 0?

Normalized?

🛠️ If not, normalize

0.0253
0.1007
-0.9880
0.1142

aws
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

🚀 Shortcut: skip operations with normalization

```
VectorCosineSimilarity(int dim, float *ax, float *bx)
{
        /* ... */
        /* Auto-vectorized */
        for (int i = 0; i < dim; i++)
        {
                similarity += ax[i] * bx[i];
                norma += ax[i] * ax[i];
                normb += bx[i] * bx[i];
        }
        /* Use sqrt(a * b) over sqrt(a) * sqrt(b) */
        return (double) similarity / sqrt((double) norma * (double) normb);
}
```

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

🚀 Shortcut: skip operations with normalization

```
VECTOR_TARGET_CLONES static float
VectorInnerProduct(int dim, float *ax, float *bx)
{
        float           distance = 0.0;

        /* Auto-vectorized */
        for (int i = 0; i < dim; i++)
                distance += ax[i] * bx[i];

        return distance;
}
```

aws
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Shortcut: skip operations with normalization

```
FUNCTION_PREFIX PG_FUNCTION_INFO_V1(vector_negative_inner_product);
Datum
vector_negative_inner_product(PG_FUNCTION_ARGS)
{
        Vector    *a = PG_GETARG_VECTOR_P(0);
        Vector    *b = PG_GETARG_VECTOR_P(1);

        CheckDims(a, b);

        PG_RETURN_FLOAT8((double) -VectorInnerProduct(a->dim, a->x, b->x));
}
```

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

What do we need to define?

1. Data type
2. Distance functions and operators
3. Indexing strategy
    1. IVFFlat
    2. HNSW

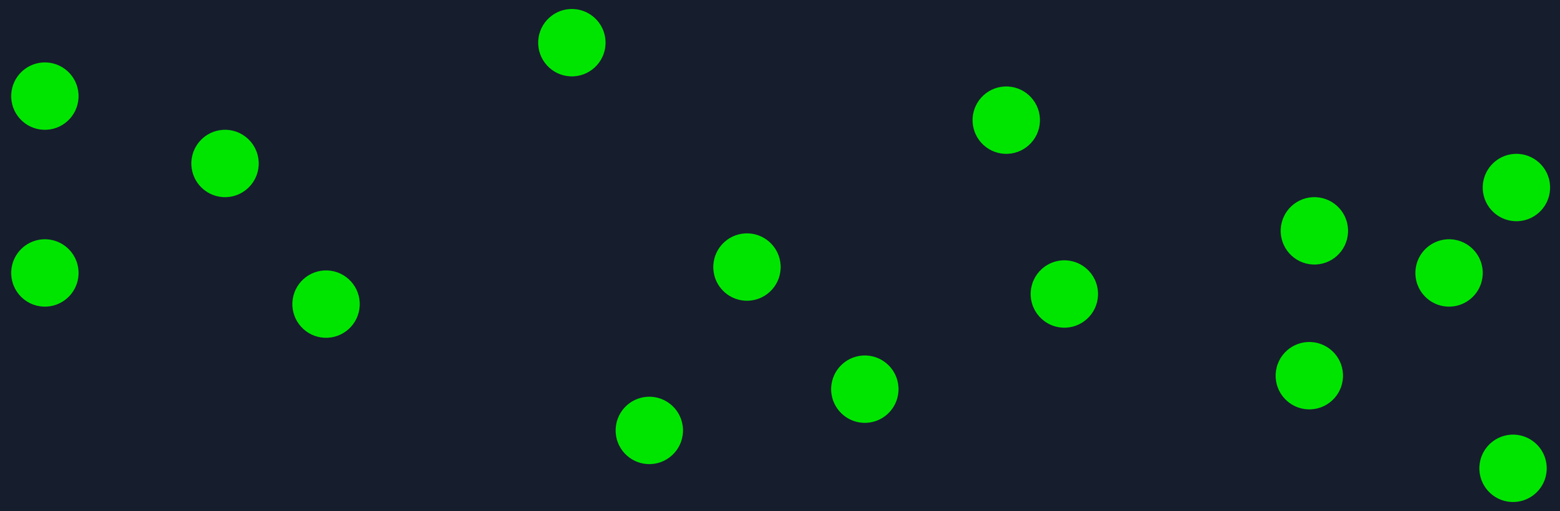
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Building an IVFFlat index

1. Sample the overall vectors in the table (`MAX(50*lists), 10,000)`)
   - Uses BlockSampler (ANALYZE method)
2. Calculate K-means (`ivfflat.lists`)
3. Assign vectors to lists in memory
4. Sort vectors in lists
5. Save index to storage

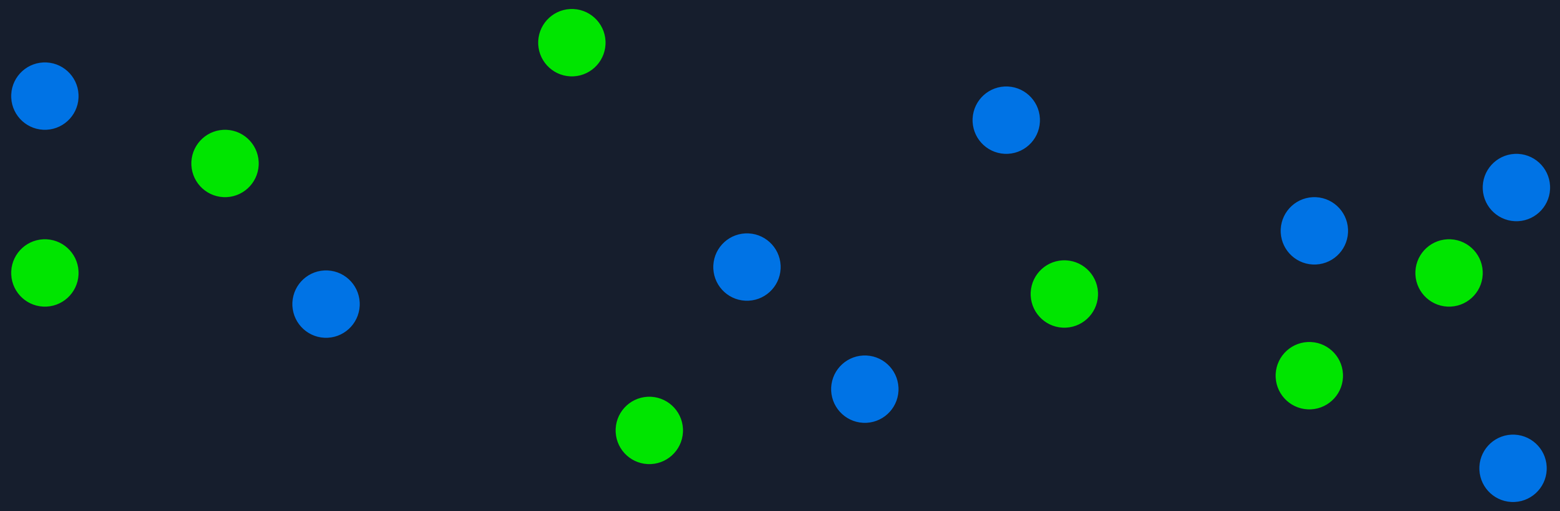
aws
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

IVFFlat: sampling



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

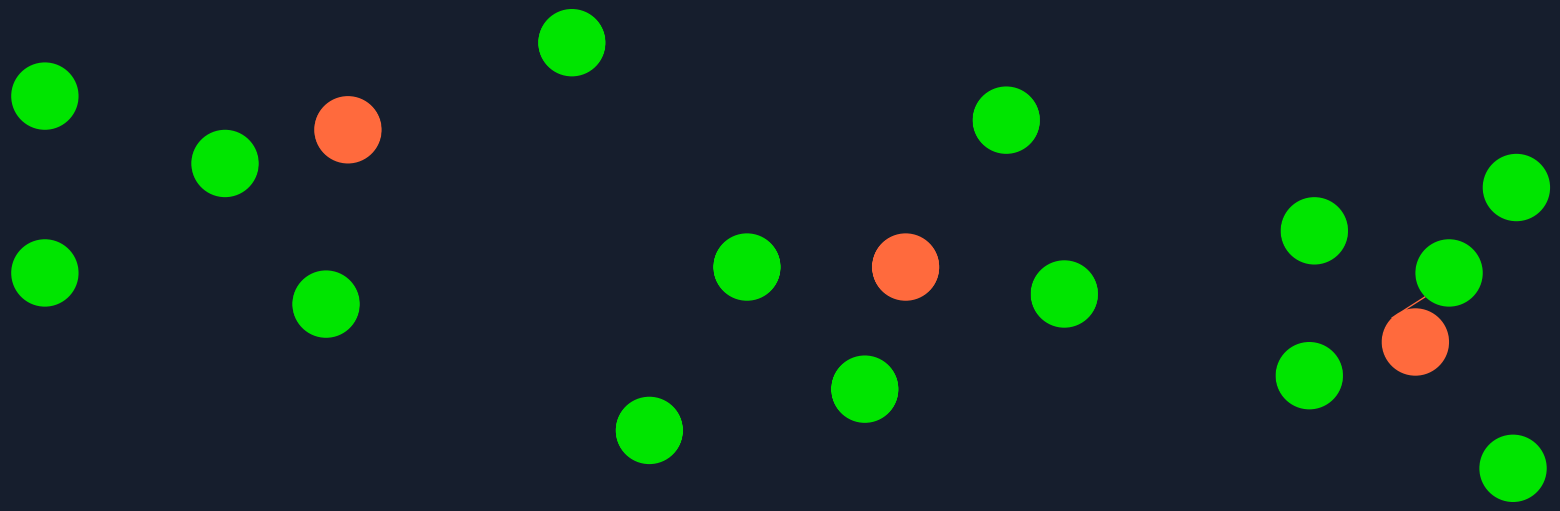
IVFFlat: sampling



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

IVFFlat: K-means

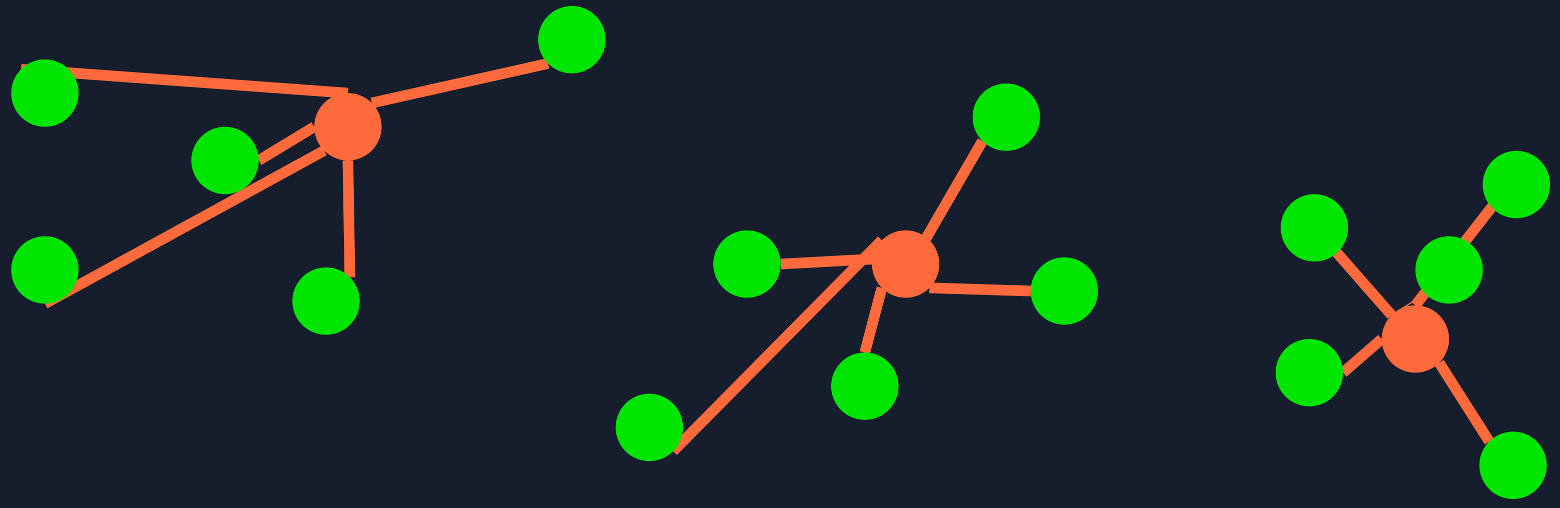
pgvector Elkan's K-means algorithms



`ivfflat.lists = 3`

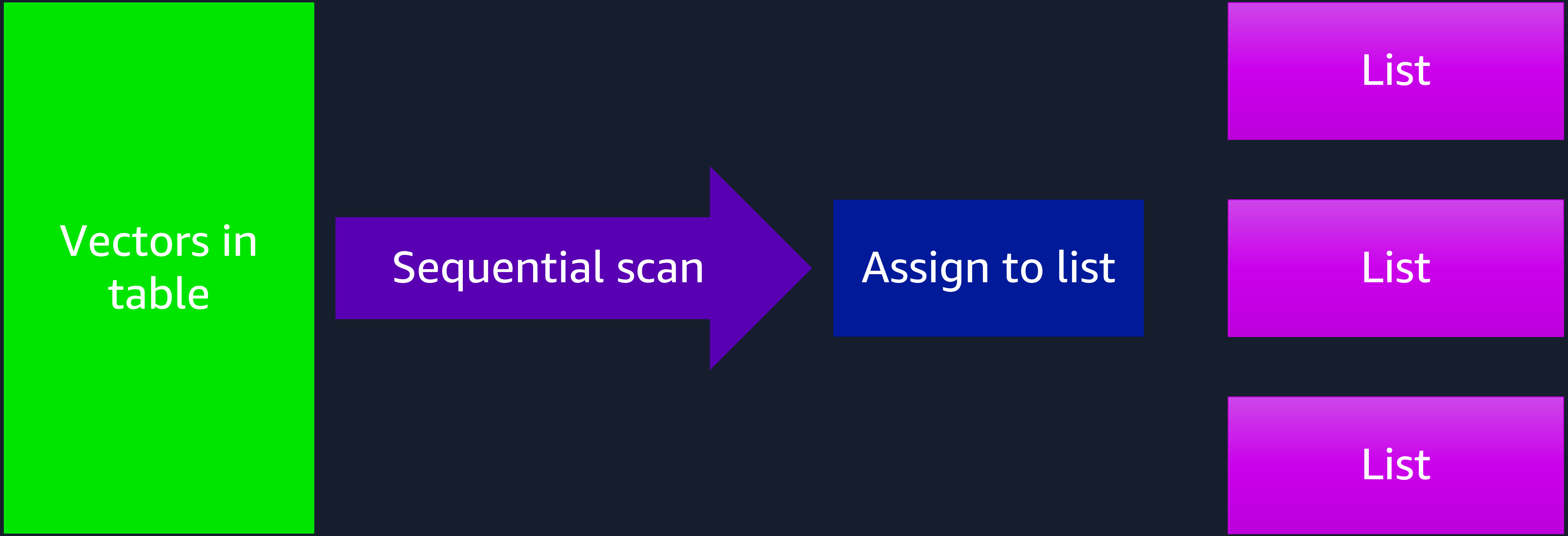
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

IVFFlat: list assignment



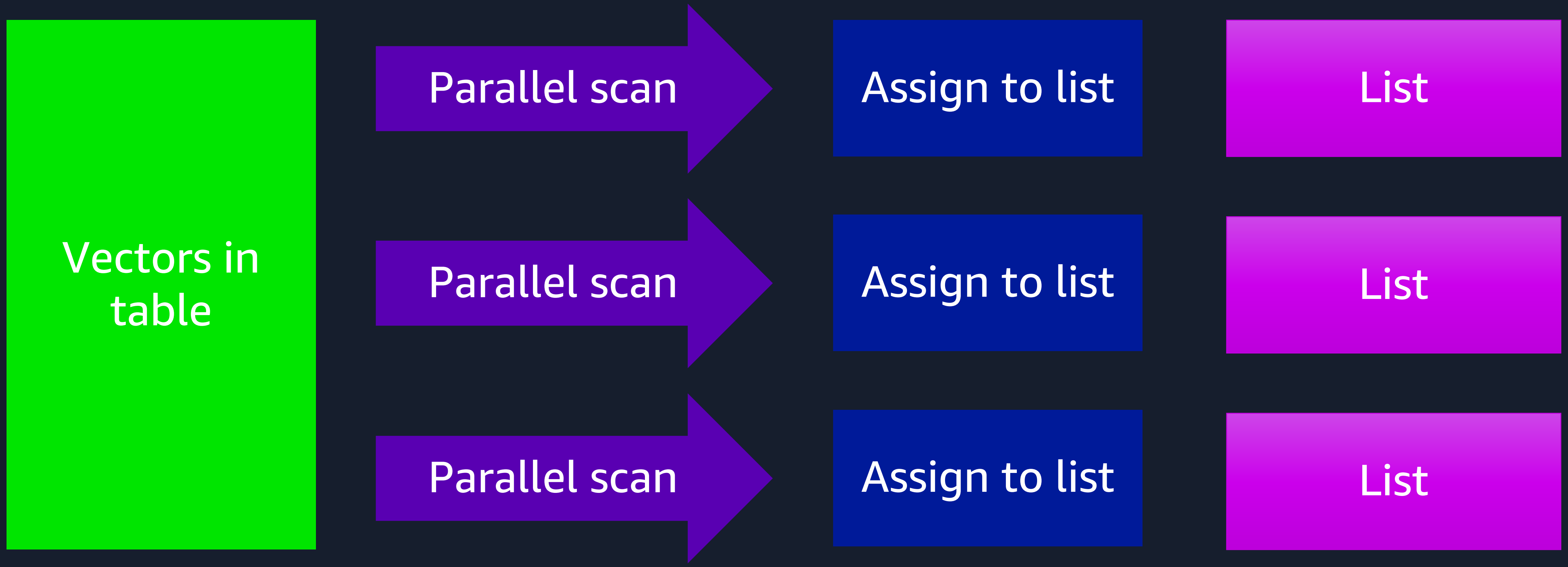
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Parallelism and list assignment



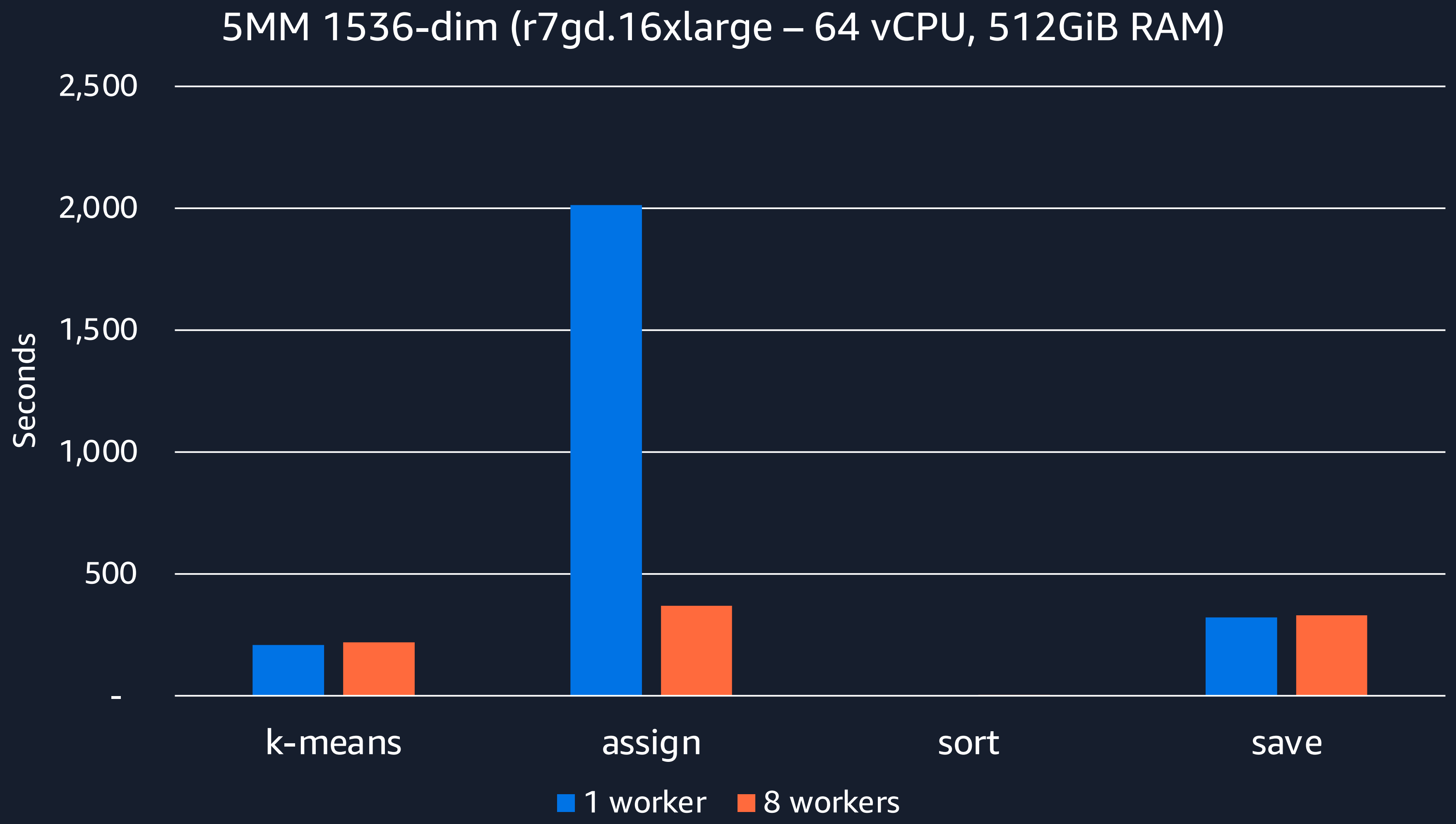
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Parallelism and list assignment



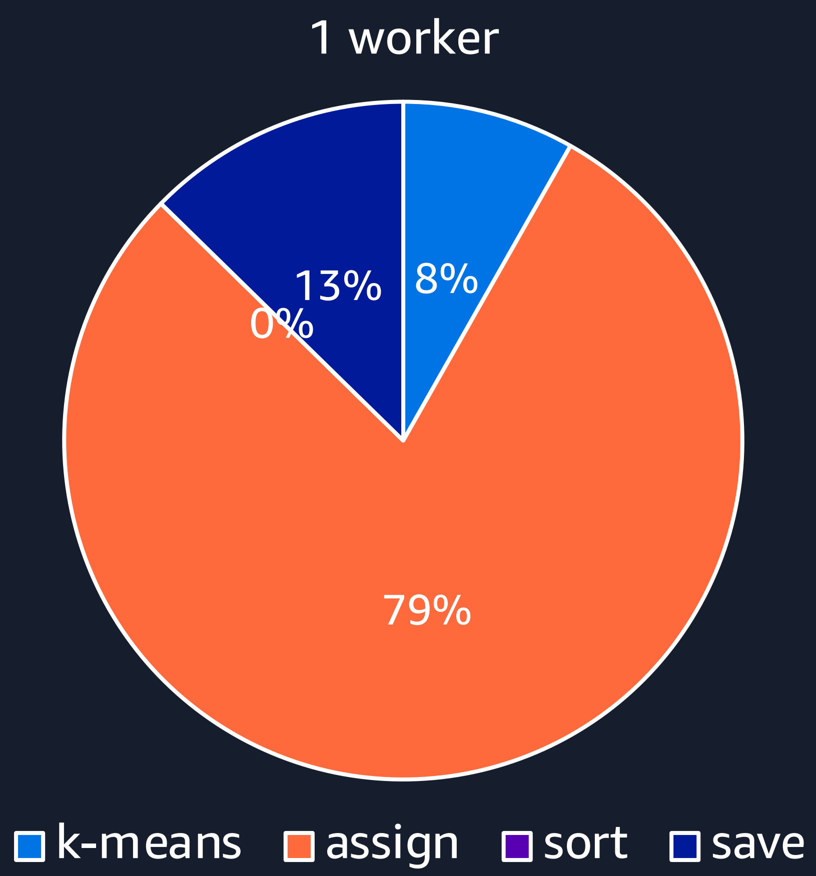
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

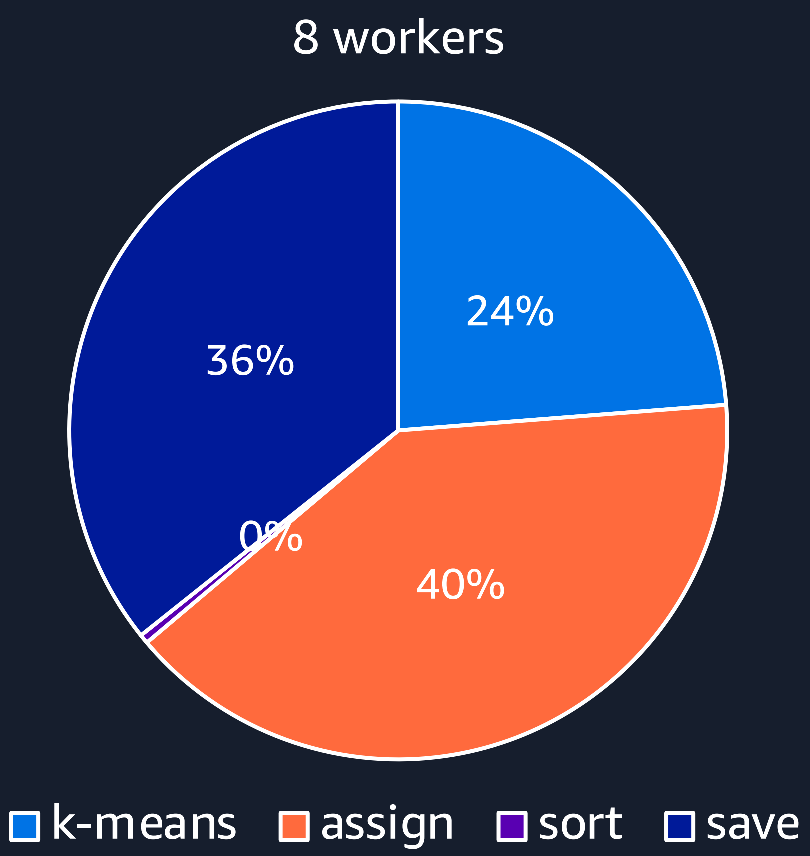
Parallelism and list assignment



aws
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

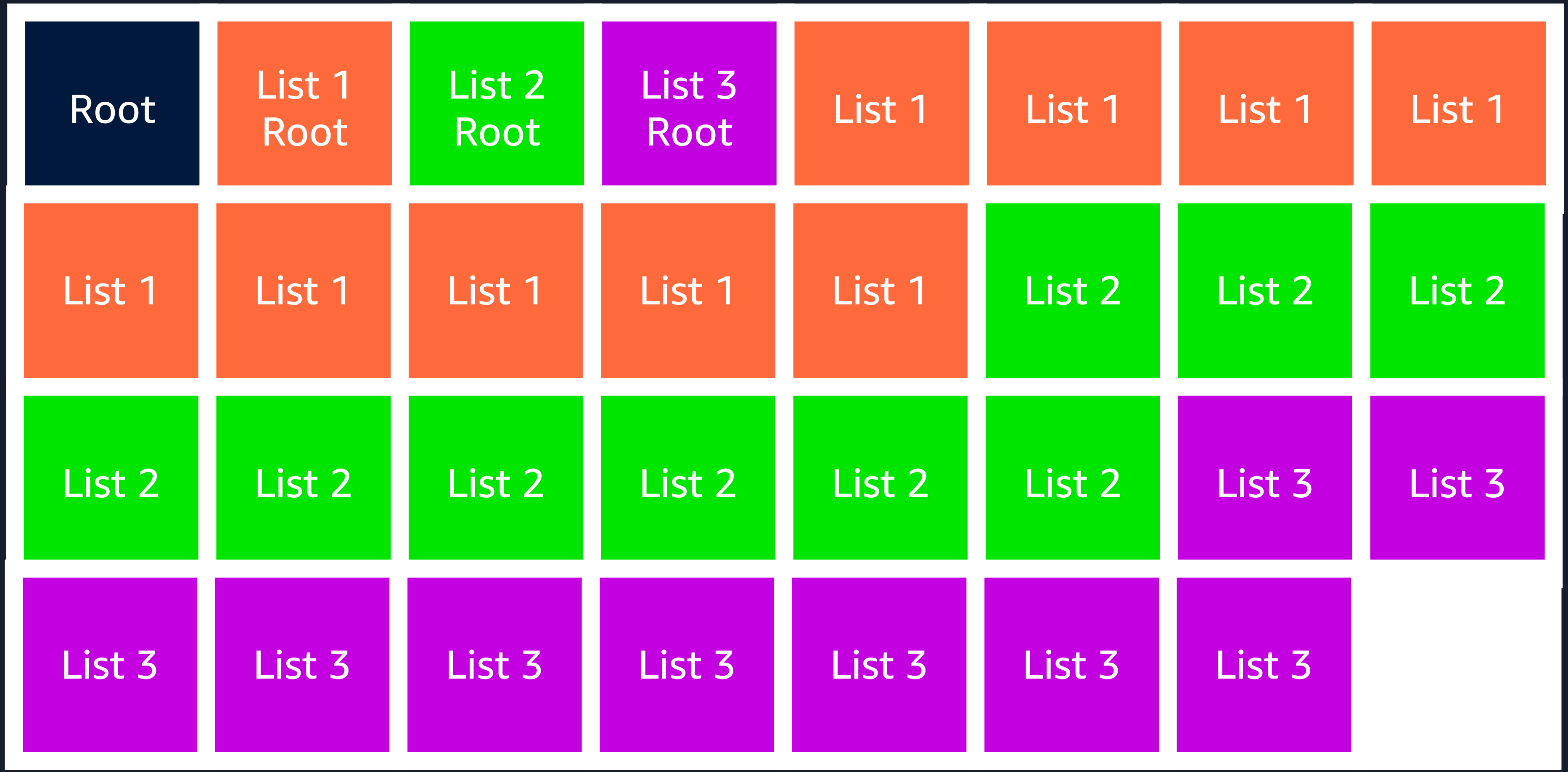
Parallelism and list assignment





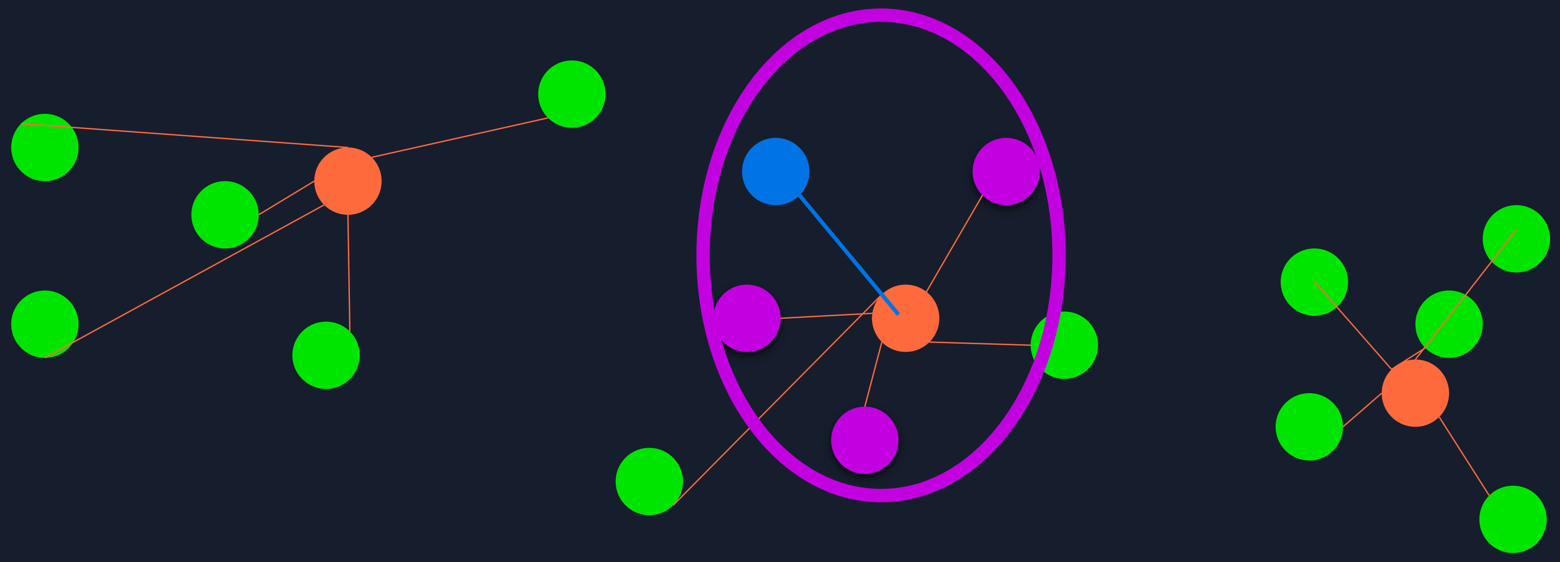
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

IVFFlat: Save index to storage



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Querying an IVFFlat index

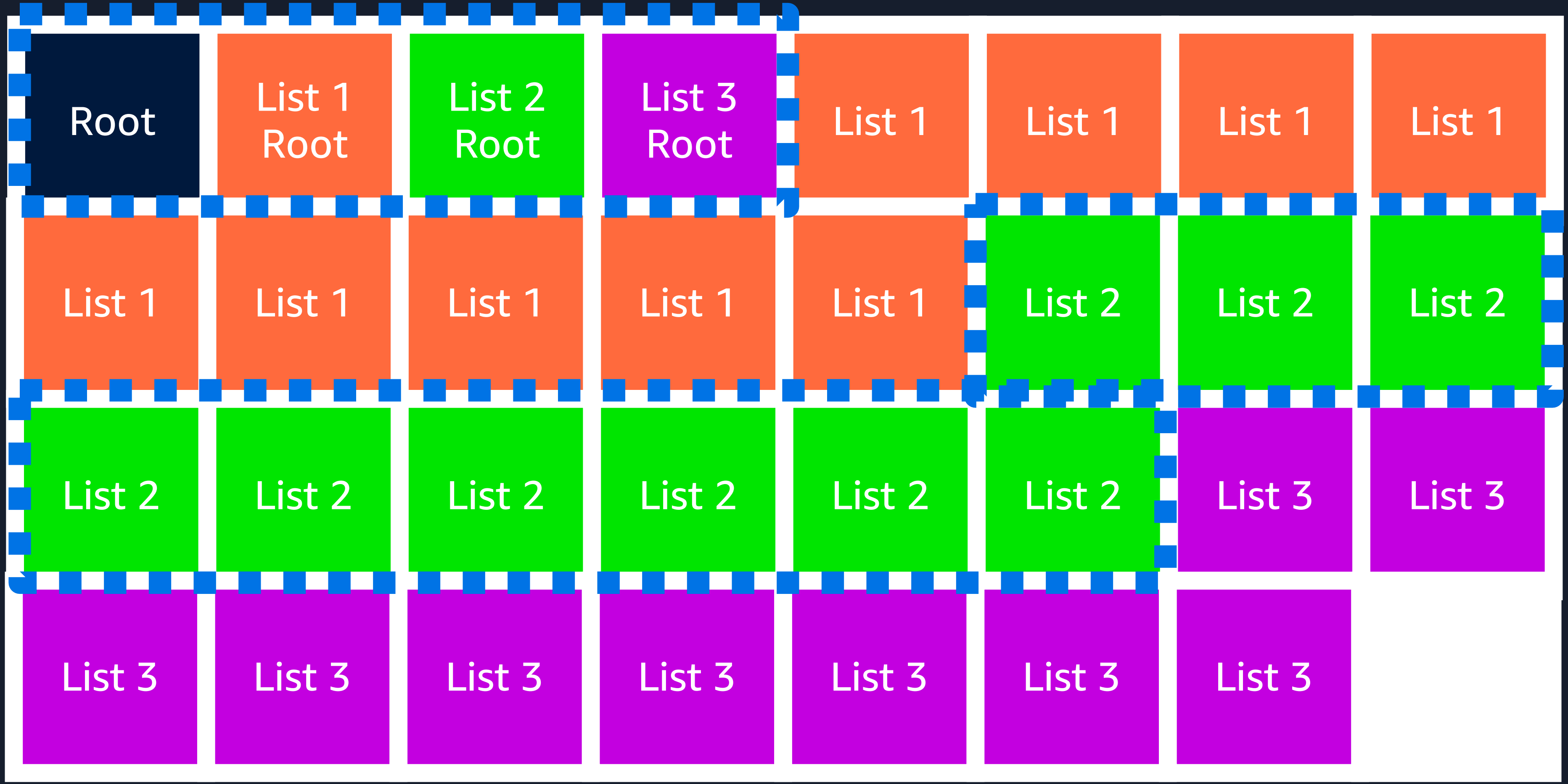


```
SET ivfflat.probes TO 1

SELECT id FROM products ORDER BY $1 <-> embedding LIMIT 3
```

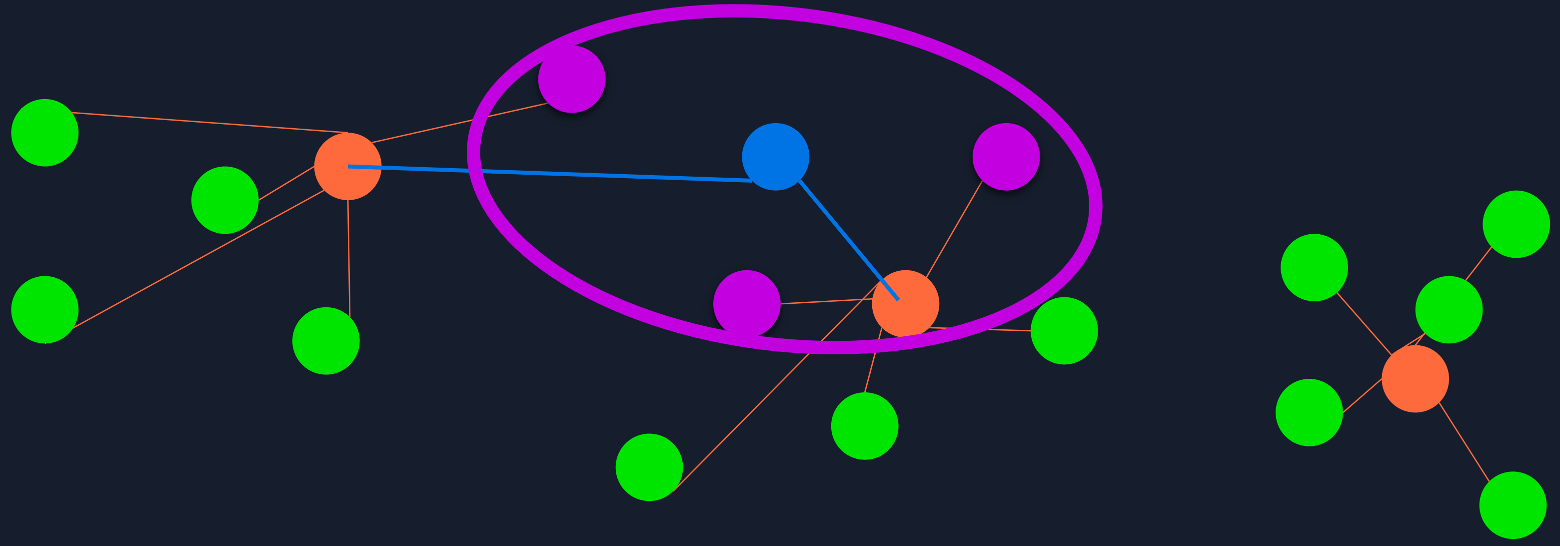
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Querying an index an IVFFlat index (1 probe)



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Querying an IVFFlat index

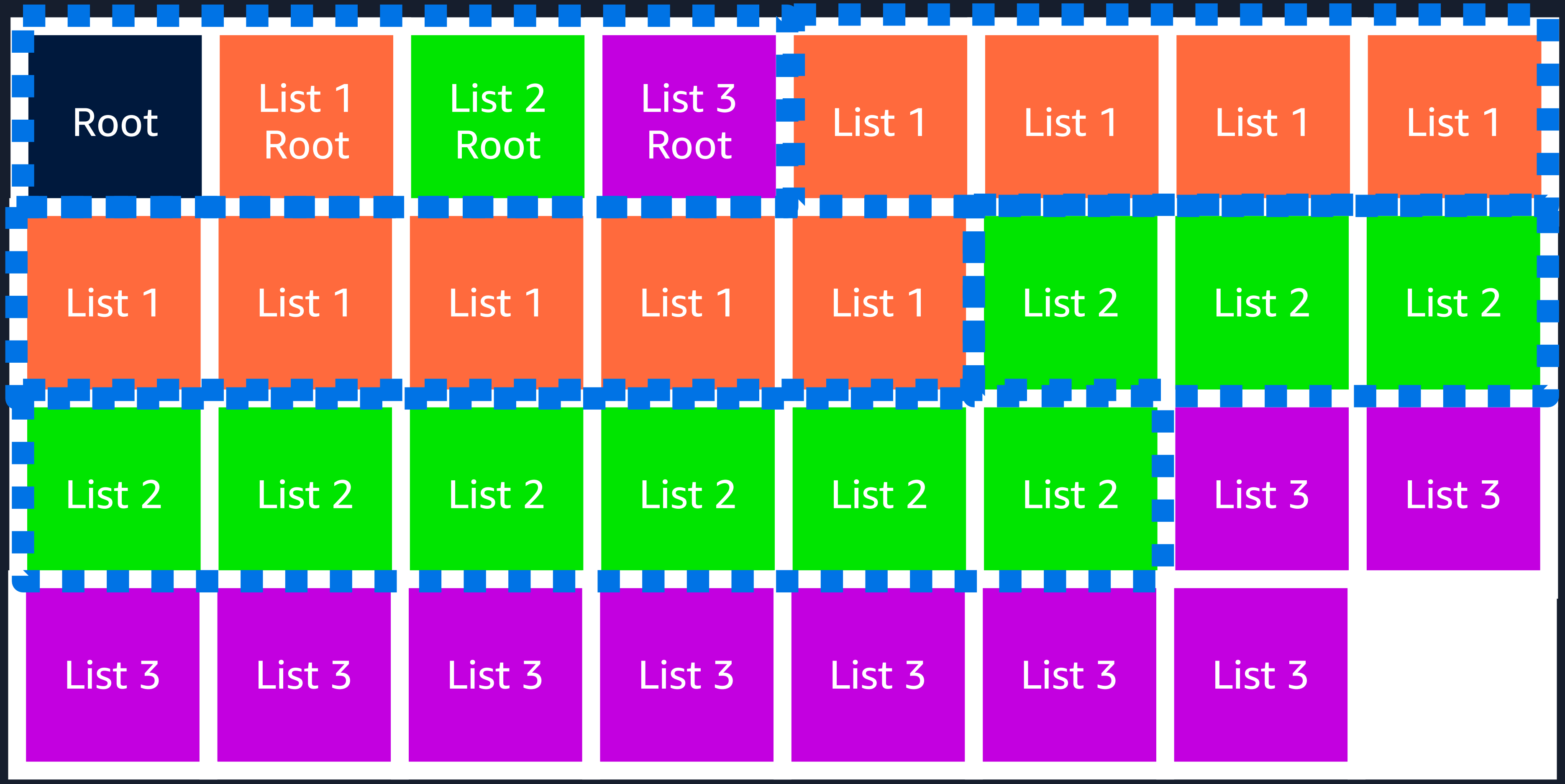


```
SET ivfflat.probes TO 2

SELECT id FROM products ORDER BY $1 <-> embedding LIMIT 3
```

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Querying an IVFFlat index (2 probes)



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

IVFFlat considerations

- Temporal locality is directly impacted by both cluster quality and query patterns
- Latency grows linearly with probes
- Lookups outside of memory can be very expensive
- Insertions / updates can skew lookups and query quality
- Opportunities
  - Streaming I/O
  - Quantization (available, requires more evaluation)
  - Additional algorithmic improvements (e.g. SPANN)

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

What do we need to define?

1. Data type
2. Distance functions and operators
3. Indexing strategy
    1. IVFFlat
    2. HNSW

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Hierarchical navigable small worlds (HNSW)

- Each vector organized into "microclusters" ("neighborhoods")

- Spend minimal time in "upper layers" – most search in bottom layer ("Layer 0")

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

HNSW index building parameters

`m`

Maximum number of bidirectional links between indexed vectors

Default: 16

`ef_construction`

Number of vectors to maintain in “nearest neighbor” list

Default: 64

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

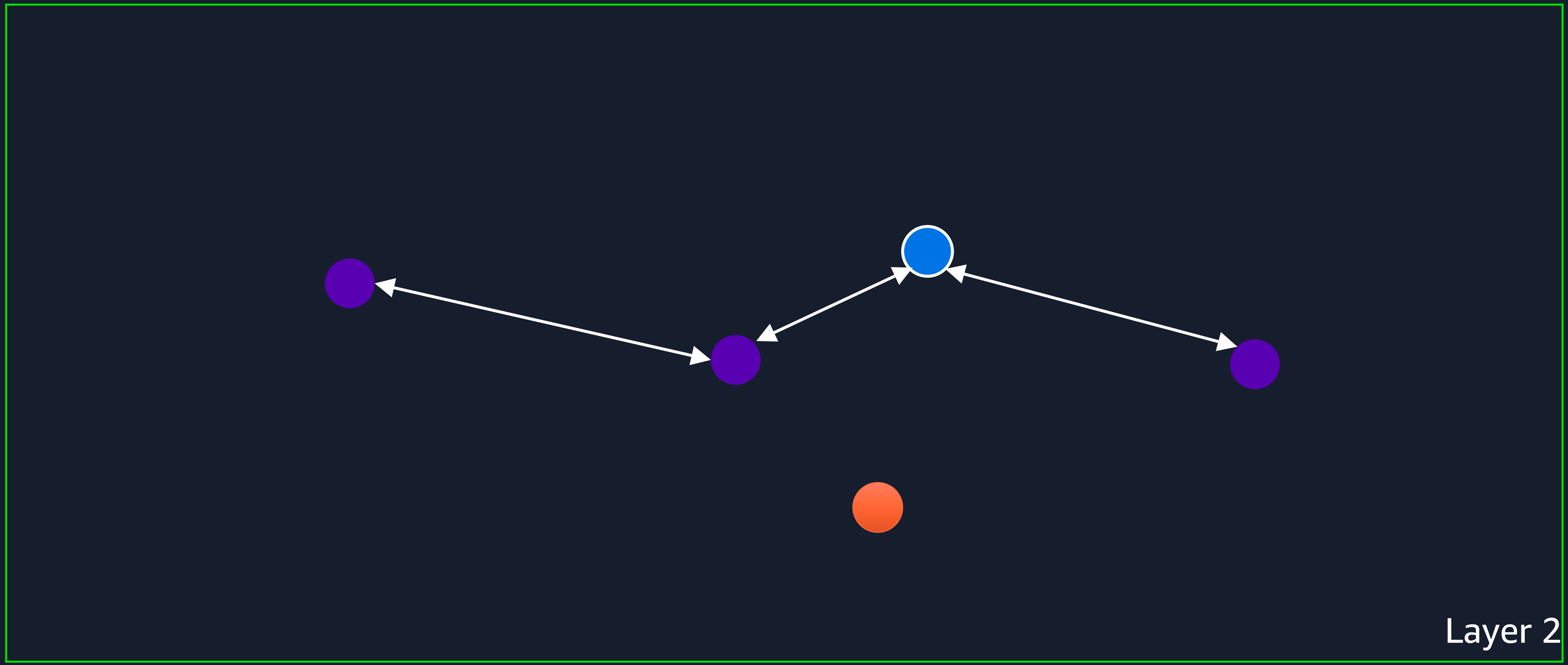
HNSW query parameters

`hnsw.ef_search`

- Number of vectors to maintain in “nearest neighbor” list
- Before v0.8, must be greater than or equal to `LIMIT`
- v0.8+, can use `hnsw.iterative_search` to satisfy unmet `LIMIT`

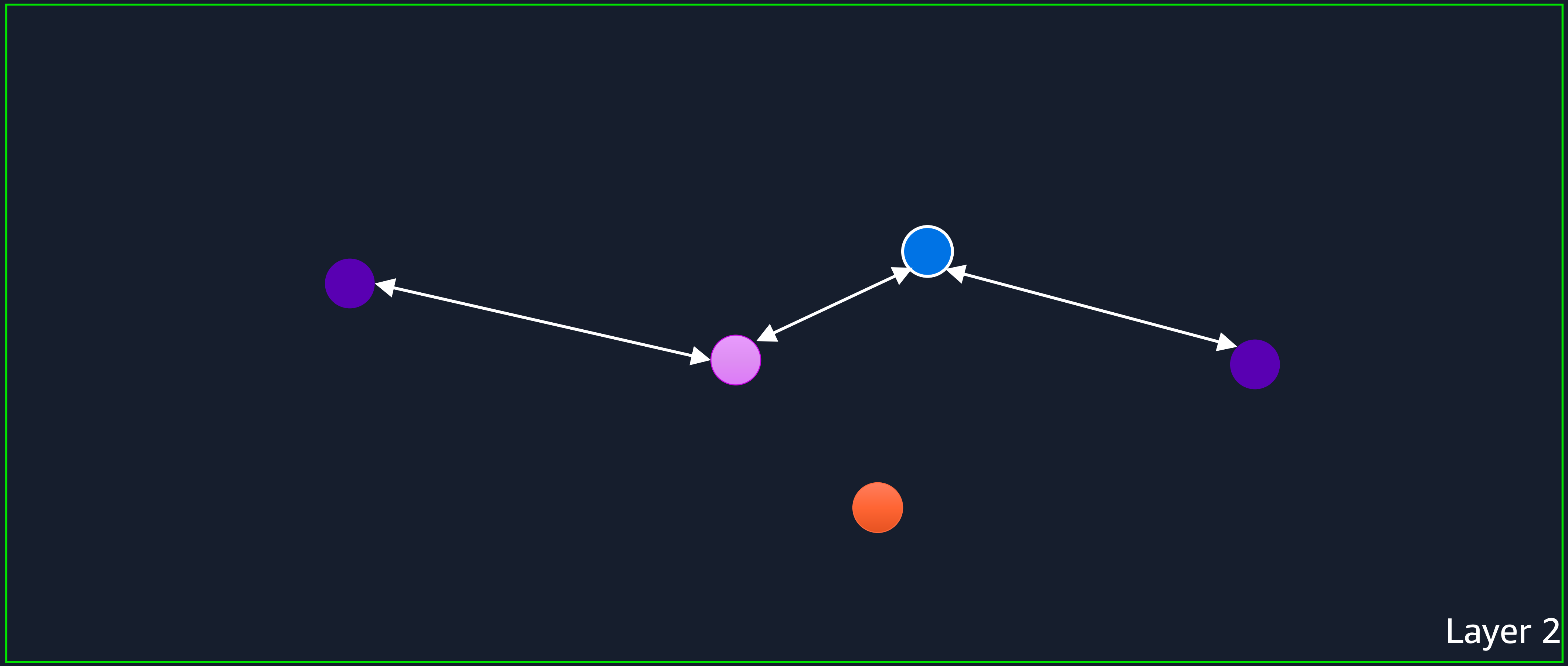
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Querying an HNSW index



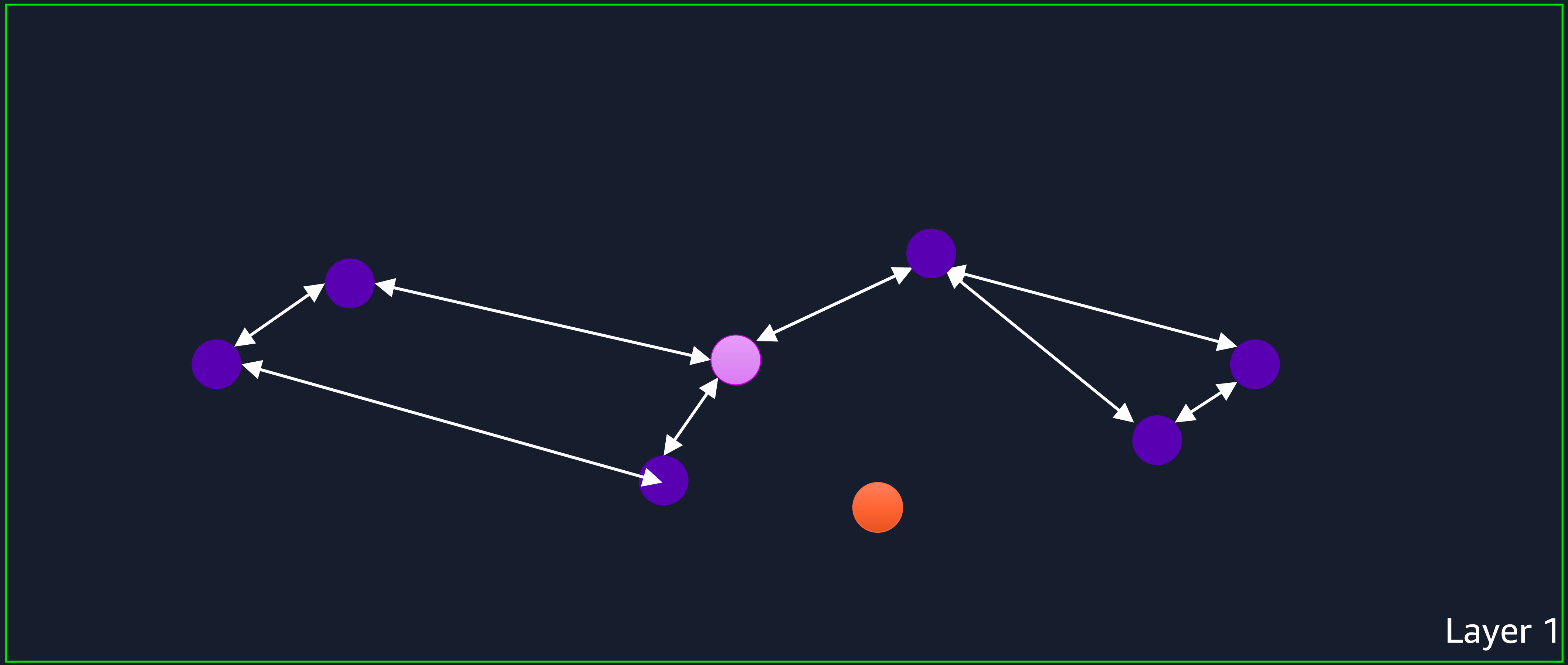
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Querying an HNSW index



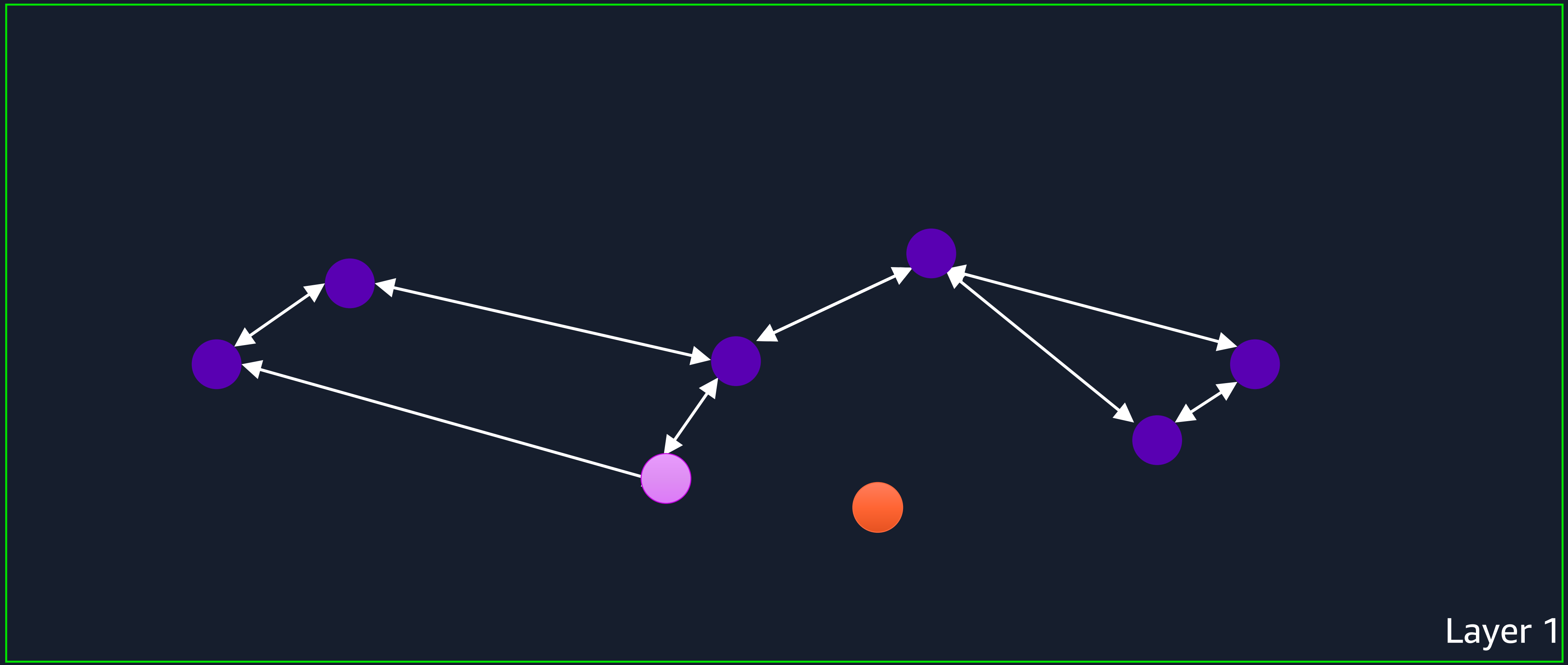
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Querying an HNSW index



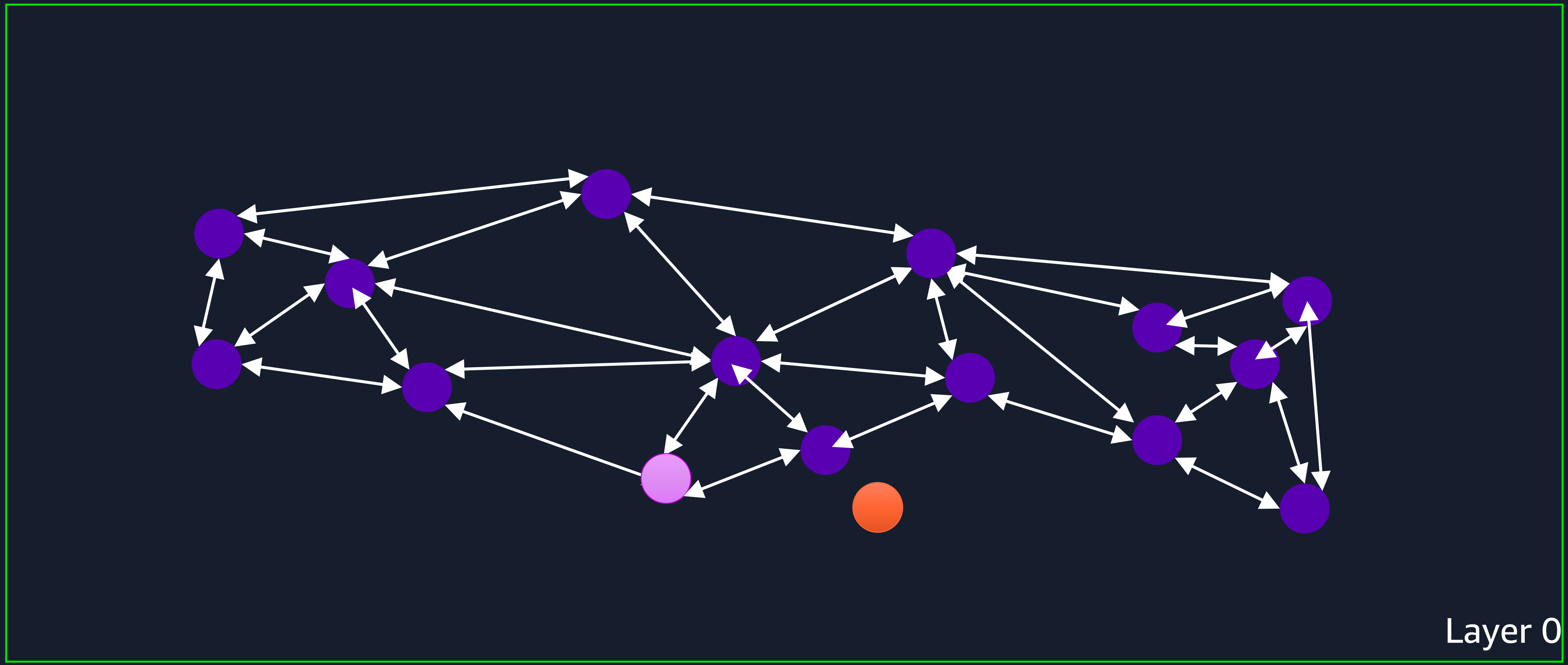
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Querying an HNSW index



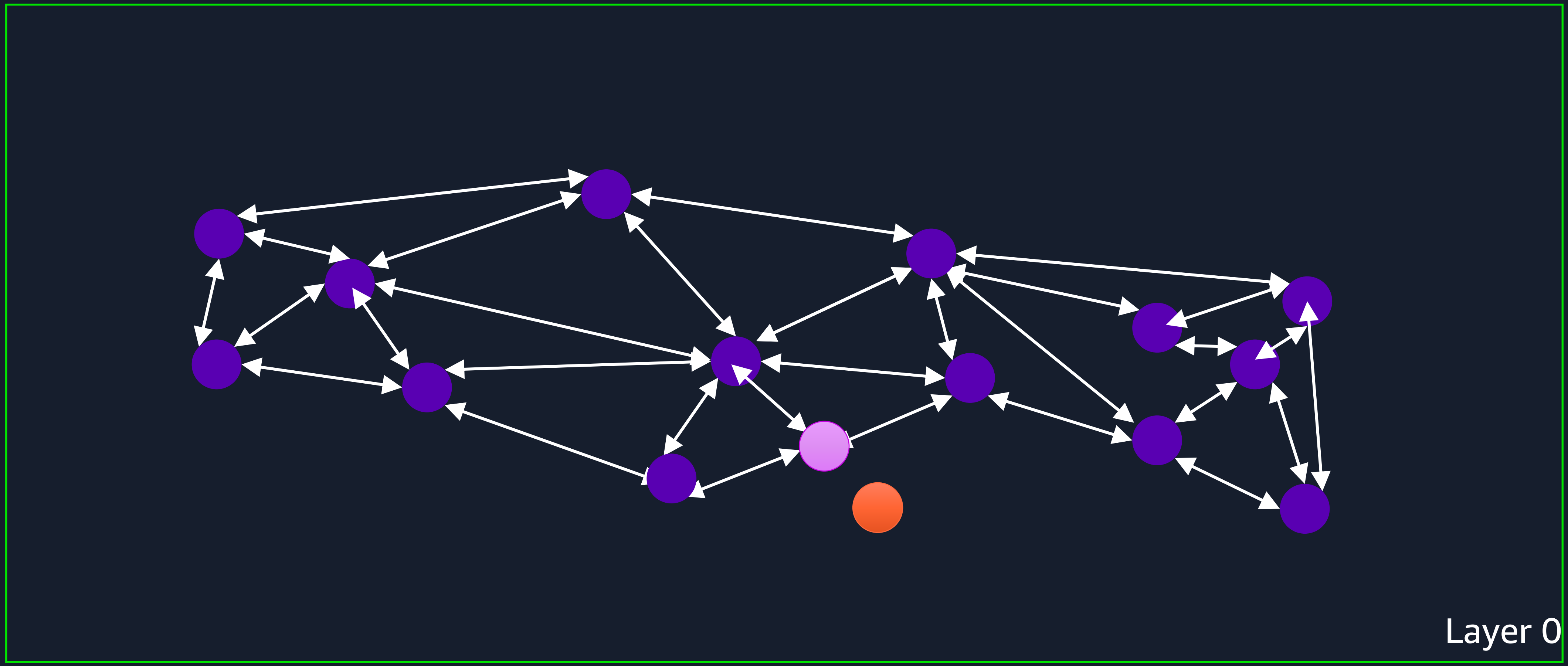
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Querying an HNSW index



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Querying an HNSW index



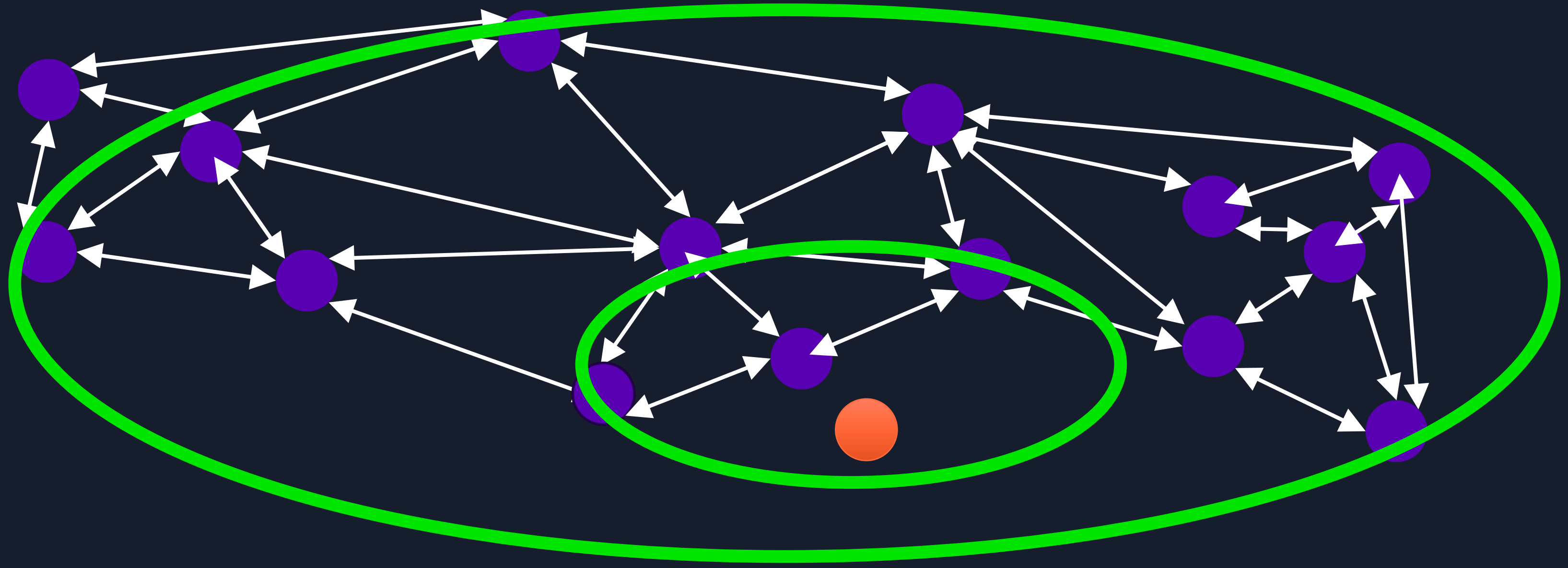
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

🚀 What happens when searching a HNSW index?

- Maintain a list of visited vectors (tuple IDs / TIDs)
- Maintain an ordered list of candidates with distances
- `ef_search` is 1 at Layer 1+
- `ef_search` is `ef_search` (default 40) at Layer 0

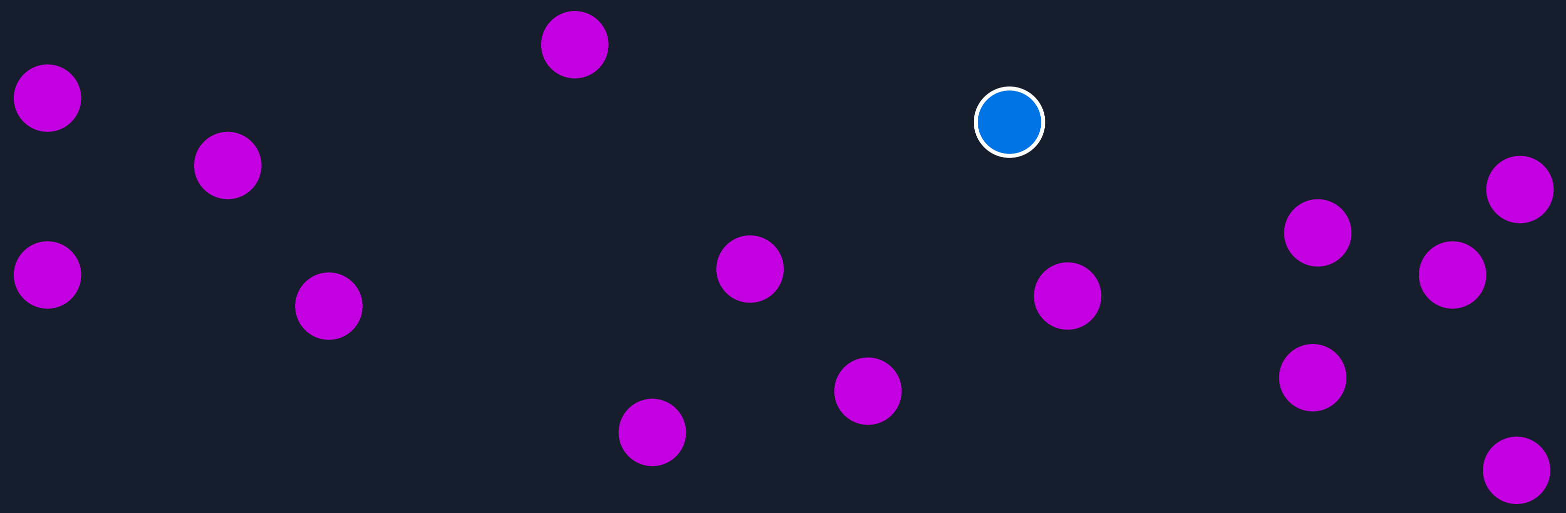
Visited
0x0102030405060708
0x0102030405060709
0x0102030405060710

Candidates	
0x0102030405060708	0.0123
0x0102030405060709	0.0434
0x0102030405060710	0.0845



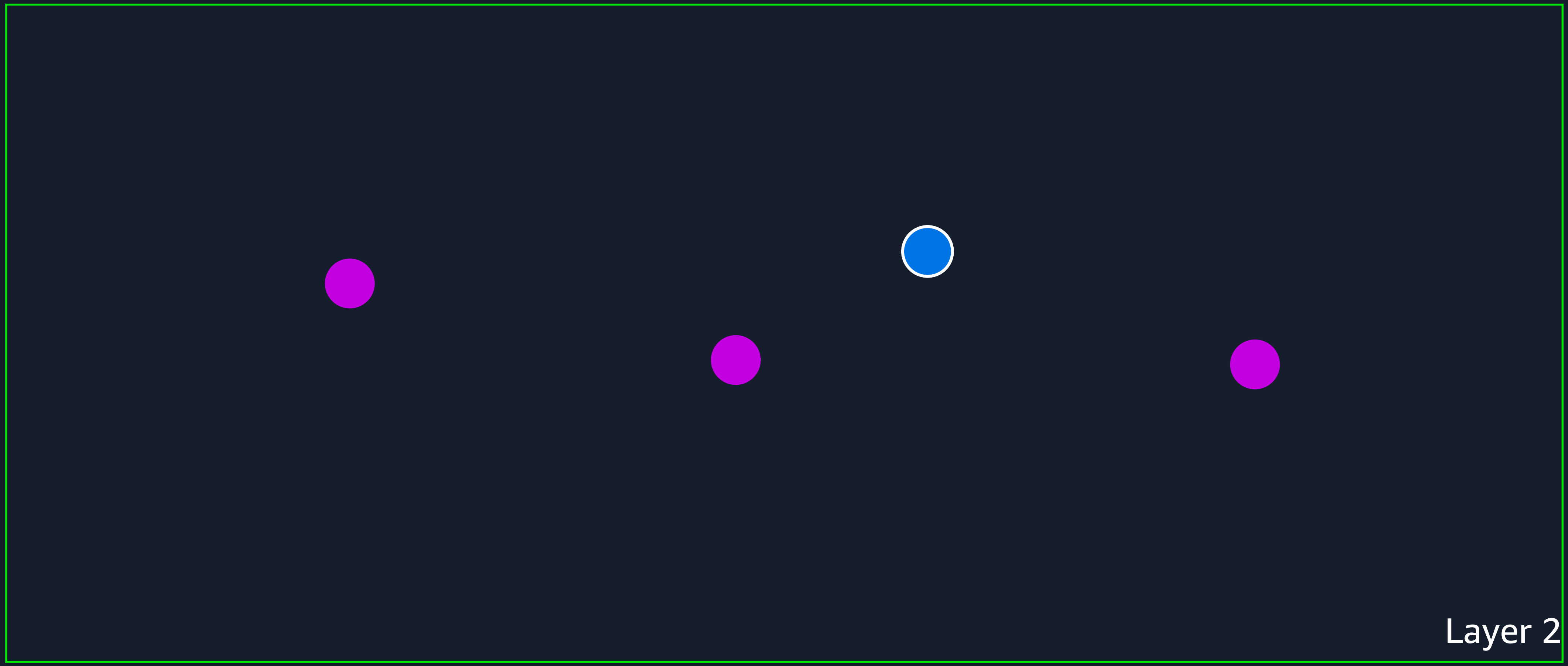
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Building an HNSW index



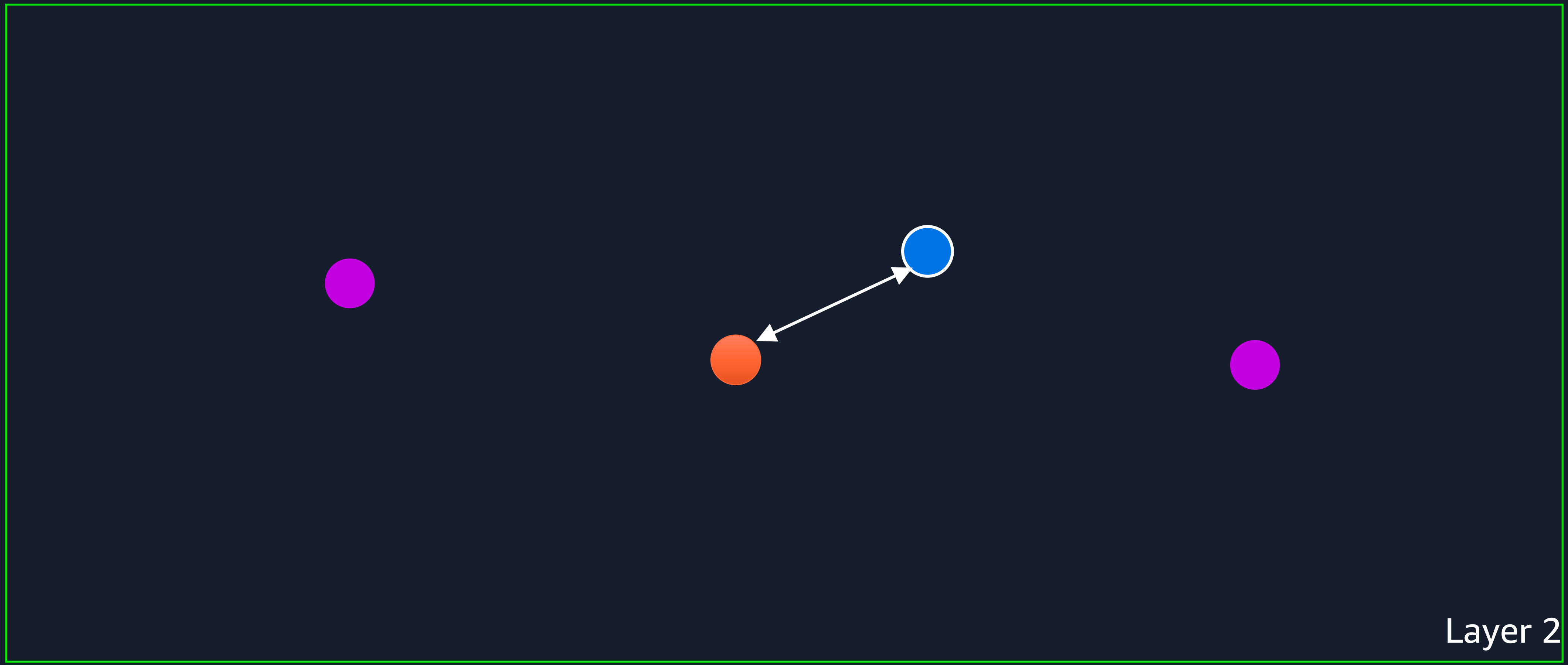
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Building an HNSW index



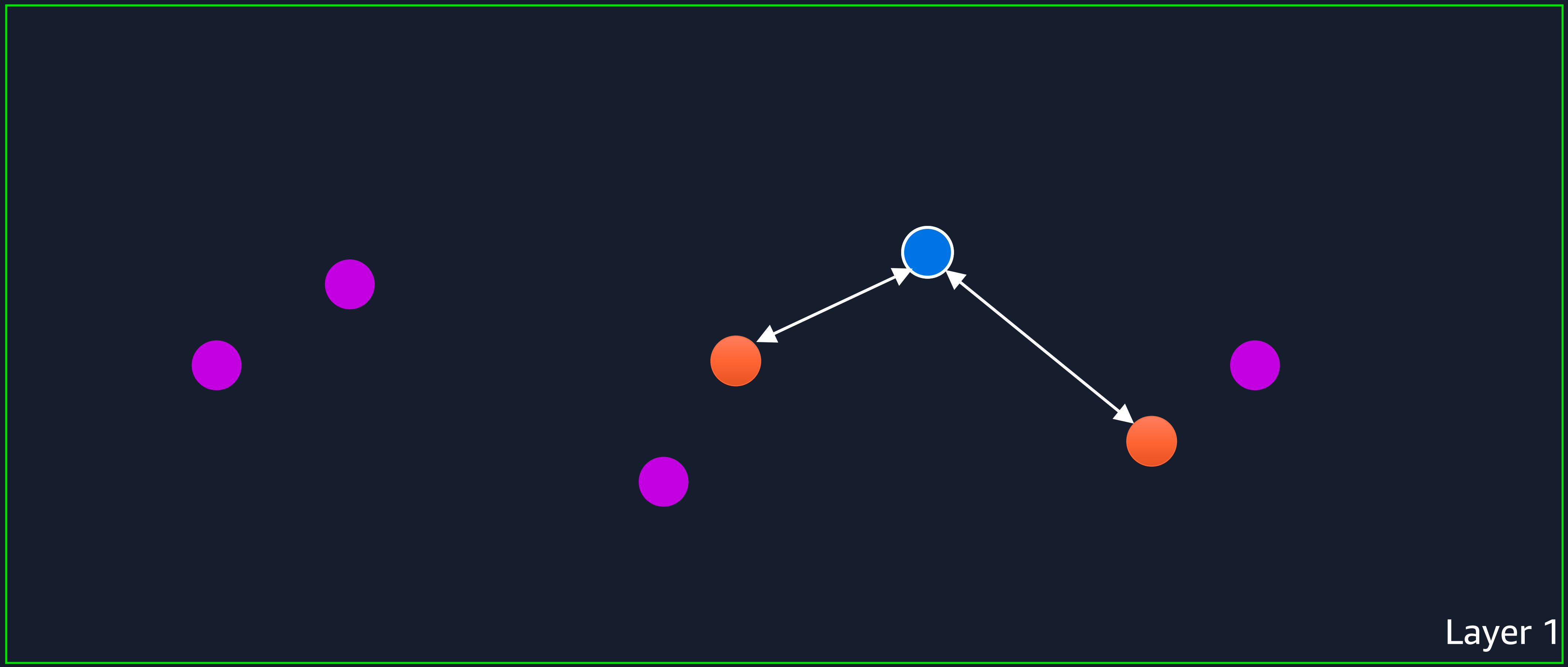
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Building an HNSW index



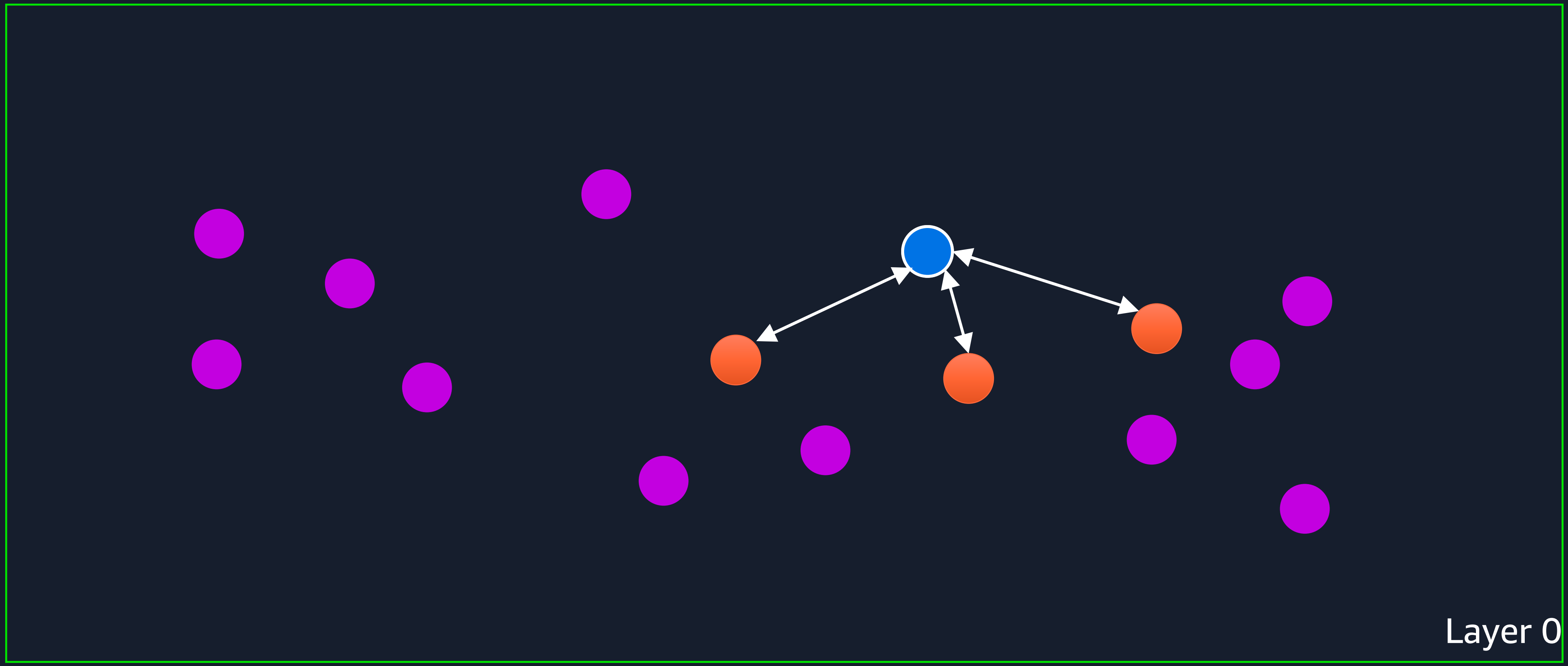
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Building an HNSW index



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Building an HNSW index



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Building an HNSW index

1. Determine entry level
2. Determine insertion method ("in memory" or "on-disk")
3. Find neighbors (similar to querying)
   - Layer 0: m * 2
   - Otherwise: m
4. Add vector to graph
5. Update neighbors' bidirectional links

```
entryLevel = (int) (-log(RandomDouble()) * ml);
```

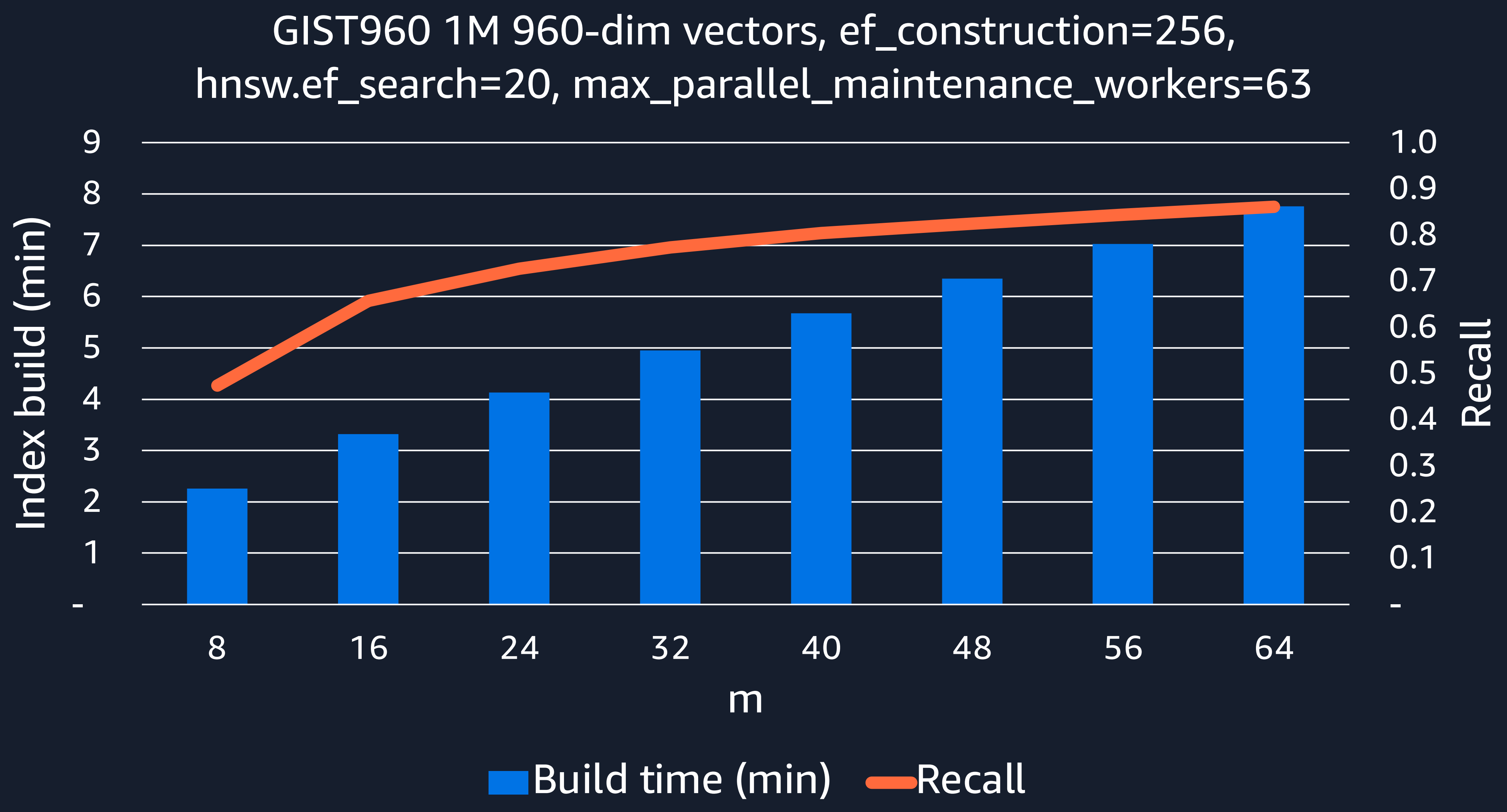
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

HNSW entry level distribution

m	Layer 0 Entry Level	Layer 1 Entry Level
2	50%	25%
4	75%	19%
8	87%	11%
12	92%	8%
16	94%	6%
20	95%	5%
24	96%	4%
32	97%	3%
36	97%	3%
48	98%	2%
64	98%	2%

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

How “m” impacts index build time & search quality



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

How "m" impacts query time via tuples scanned

m=16, ef_construction=64 **# tuples scanned**					
ef	SIFT (N=1M)	GIST (N=1M)	GLoVE25 (N=1.1M)	1536d (N=5M)	768d (N=10M)
10	427	512	438	456	498
20	643	779	652	650	695
40	1044	1272	1049	1005	1050
80	1774	2212	1761	1629	1762
120	2438	3099	2420	2214	2449
200	3638	4755	3629	3328	3833
400	6247	8402	6303	5836	7190
800	10619	14706	10938	10563	13258

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

How "m" impacts query time via tuples scanned

ef	1536d (N=5M,m=16)	1536d (N=5M,m=64)	768d (N=10M,m=16)	768d (N=10M,m=64)
10	456	605	498	1425
20	650	1257	695	2038
40	1005	2292	1050	3246
80	1629	4049	1762	5691
120	2214	5728	2449	8046
200	3328	8601	3833	12664
400	5836	15158	7190	23284
800	10563	27249	13258	42200

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

HNSW scan cost estimation

```
/*
 * HNSW cost estimation follows a formula that accounts for the total
 * number of tuples indexed combined with the parameters that most
 * influence the duration of the index scan, namely: m - the number of
 * tuples that are scanned in each step of the HNSW graph traversal
 * ef_search - which influences the total number of steps taken at layer 0
 *
 * The source of the vector data can impact how many steps it takes to
 * converge on the set of vectors to return to the executor. Currently, we
 * use a hardcoded scaling factor (HNSWScanScalingFactor) to help
 * influence that, but this could later become a configurable parameter
 * based on the cost estimations.
 *
 * The tuple estimator formula is below:
 *
 * numIndexTuples = entryLevel * m + layer0TuplesMax * layer0Selectivity
 */
```

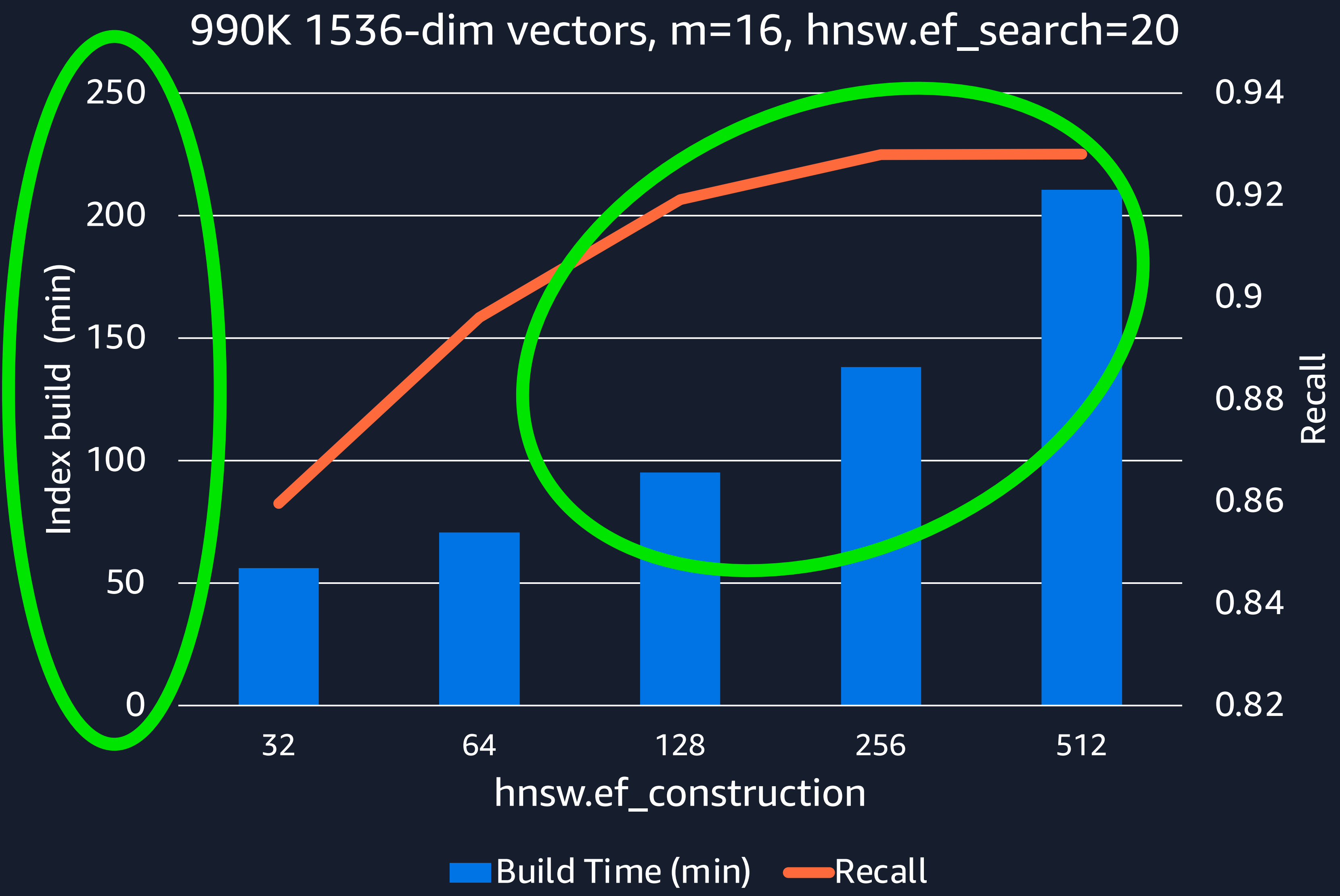
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Why is cost estimation important?

- Guides PostgreSQL query planner to select "best path"
- Filtering (`WHERE` clause)
  - A different index (B-tree) or a sequential scan may be a better choice based on selectivity

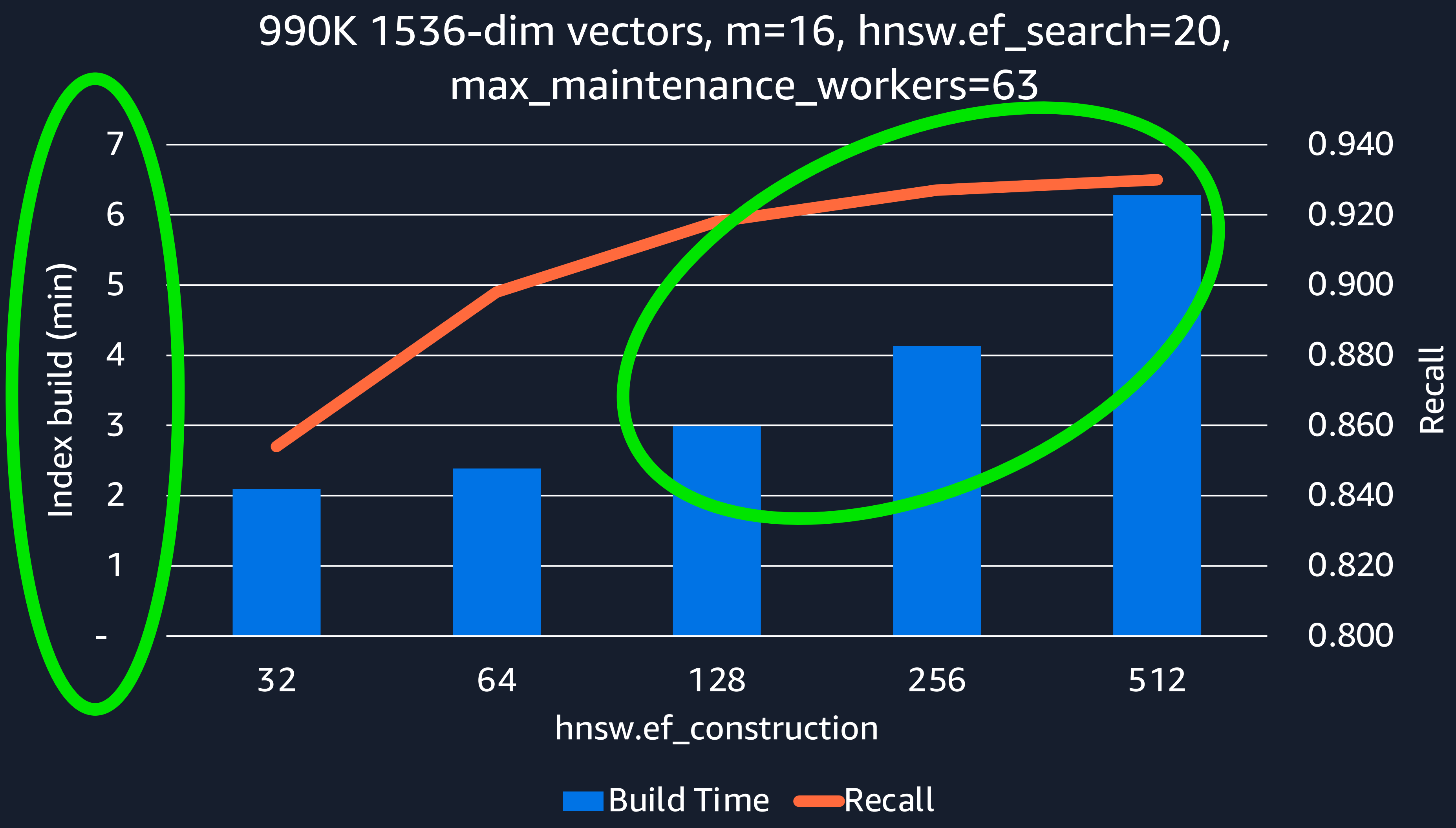
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Why index build speed matters (serial build)



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

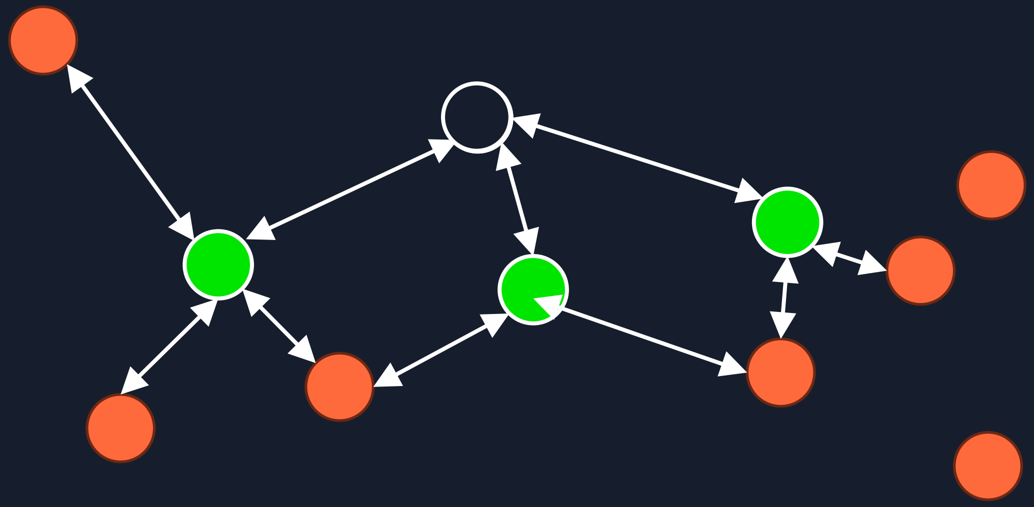
Why index build speed matters (parallel build)



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

pgvector and HNSW index maintenance

- Innovation: pgvector HNSW implementation supports updates and deletes



Phase 1: Hide

Phase 2: Repair

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

HNSW considerations

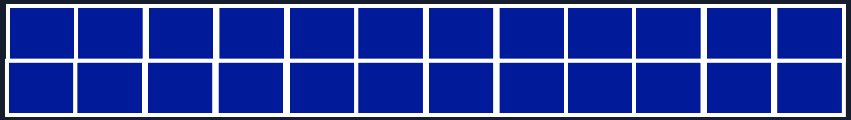
- Embedding model impacts overall query time
- Filtering
  - Iterative scans vs. using other search methods
  - Bitmap scans(?)
- Opportunities to accelerate time spent in Layer 0
- Opportunities
  - Streaming I/O
  - Parallel vacuum
  - "Smart" graph repair to improve clustering

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

What is quantization?

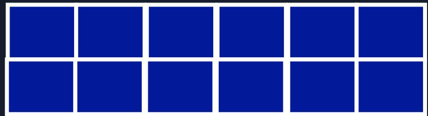
Flat

[0.0435122, -0.2304432, -0.4521324,
0.98652234, -0.1123234, 0.75401234]



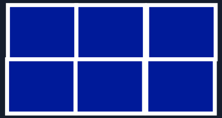
Scalar quantization (2-byte float)

[0.0432, -0.234, -0.452,0.986,
-0.112, 0.751]



Scalar quantization (1-byte uint)

[129, 99, 67, 244, 126, 230]



Binary quantization

[1, 0, 0, 1, 0, 1]

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

pgvector and scalar quantization (2 byte)

```sql
CREATE INDEX ON documents USING
      hnsw((embedding::halfvec(3072)) halfvec_cosine_ops);


SELECT id
FROM documents
ORDER BY embedding::halfvec(3072) <=> $1::halfvec(3072)
LIMIT 10;
```

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

pgvector and binary quantization

```sql
CREATE INDEX ON documents USING
     hnsw ((binary_quantize(embedding)::bit(3072)) bit_hamming_ops);


SELECT i.id FROM (
     SELECT id, embedding <=> $1 AS distance
     FROM items
     ORDER BY
       binary_quantize(embedding)::bit(3072) <~> binary_quantize($1)
     LIMIT 40 -- set to hnsw.ef_search
) i
ORDER BY i.distance
LIMIT 10;
```

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

1536d 5MM (r7i.16xlarge, m=16, ef_construction=256)			
	Flat	2-byte float	Binary (rerank)
Index Size (GB)	38.15	19.07	2.34
Index build time (min)	21	13	4
Recall @ ef_search = 40	0.931	0.929	0.811
QPS @ ef_search = 40	24,216	27,084	33,984
Recall @ ef_search = 80	0.965	0.961	0.900
QPS @ ef_search = 80	11,057	12,759	20,410
Recall @ ef_search = 220	0.989	0.983	0.963
QPS @ ef_search = 220	5,242	5,983	7,856

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

1536d 5MM (r7i.16xlarge, m=16, ef_construction=256)			
	Flat	2-byte float	Binary (rerank)
Index Size (GB)	38.15	19.07	2.34
Index build time (min)	21	13	4
Recall @ ef_search = 40	0.931	0.929	0.811
QPS @ ef_search = 40	24,216	27,084	33,984
Recall @ ef_search = 80	0.965	0.961	0.900
QPS @ ef_search = 80	11,057	12,759	20,410
Recall @ ef_search = 220	0.989	0.983	0.963
QPS @ ef_search = 220	5,242	5,983	7,856

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

1536d 5MM (r7i.16xlarge, m=16, ef_construction=256)			
	Flat	2-byte float	Binary (rerank)
Index Size (GB)	38.15	19.07	2.34
Index build time (min)	21	13	4
Recall @ ef_search = 40	0.931	0.929	0.811
QPS @ ef_search = 40	24,216	27,084	33,984
Recall @ ef_search = 80	0.965	0.961	0.900
QPS @ ef_search = 80	11,057	12,759	20,410
Recall @ ef_search = 220	0.989	0.983	0.963
QPS @ ef_search = 220	5,242	5,983	7,856

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

1536d 5MM (r7i.16xlarge, m=16, ef_construction=256)			
	Flat	2-byte float	Binary (rerank)
Index Size (GB)	38.15	19.07	2.34
Index build time (min)	21	13	4
Recall @ ef_search = 40	0.931	0.929	0.811
QPS @ ef_search = 40	24,216	27,084	33,984
Recall @ ef_search = 80	0.965	0.961	0.900
QPS @ ef_search = 80	11,057	12,759	20,410
Recall @ ef_search = 220	0.989	0.983	0.963
QPS @ ef_search = 220	5,242	5,983	7,856

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Ongoing work

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Areas to further explore

- "Multi-column" vector indexes
- Efficient batch queries
- Recall boosting techniques (statistical binary quantization, hybrid search)
- Demonstrably improved algorithms
- Upstream PostgreSQL changes that help vector search patterns

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Conclusion

- What works in memory may or may not work with storage-based systems

- Extensible framework of PostgreSQL simplifies adding new search systems
  - "You have vector search…and every other PostgreSQL feature"

- Rapidly evolving space, including open areas of research (e.g., filtering)

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Thank you!

Jonathan Katz

jkatz@amazon.com

@jkatz05

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.