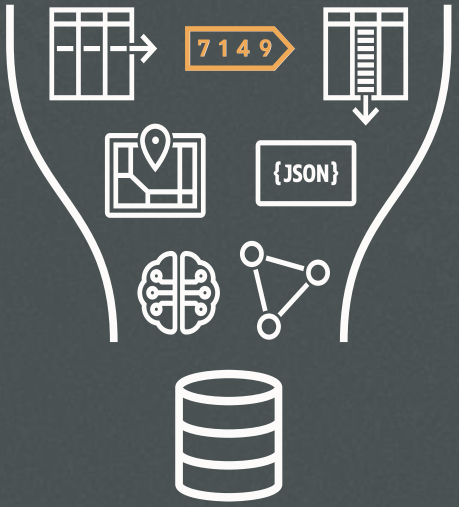
Safe harbor statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, timing, and pricing of any features or functionality described for Oracle's products may change and remains at the sole discretion of Oracle Corporation.

ORACLE

Enabling Generative AI with Oracle AI Vector Search

Better Together: Business Data and Business Vectors

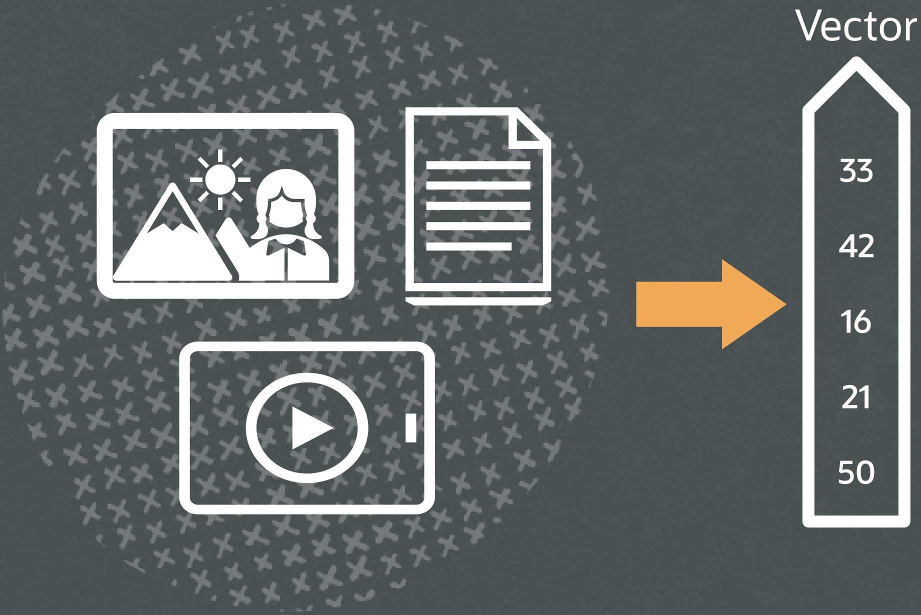


Converged Database

The best solution is to add vector search to your business database

- There is no need to move and synchronize data, manage multiple products, etc.

Vectors are used in AI to capture the semantics of data: Images, documents, videos, or even structured data

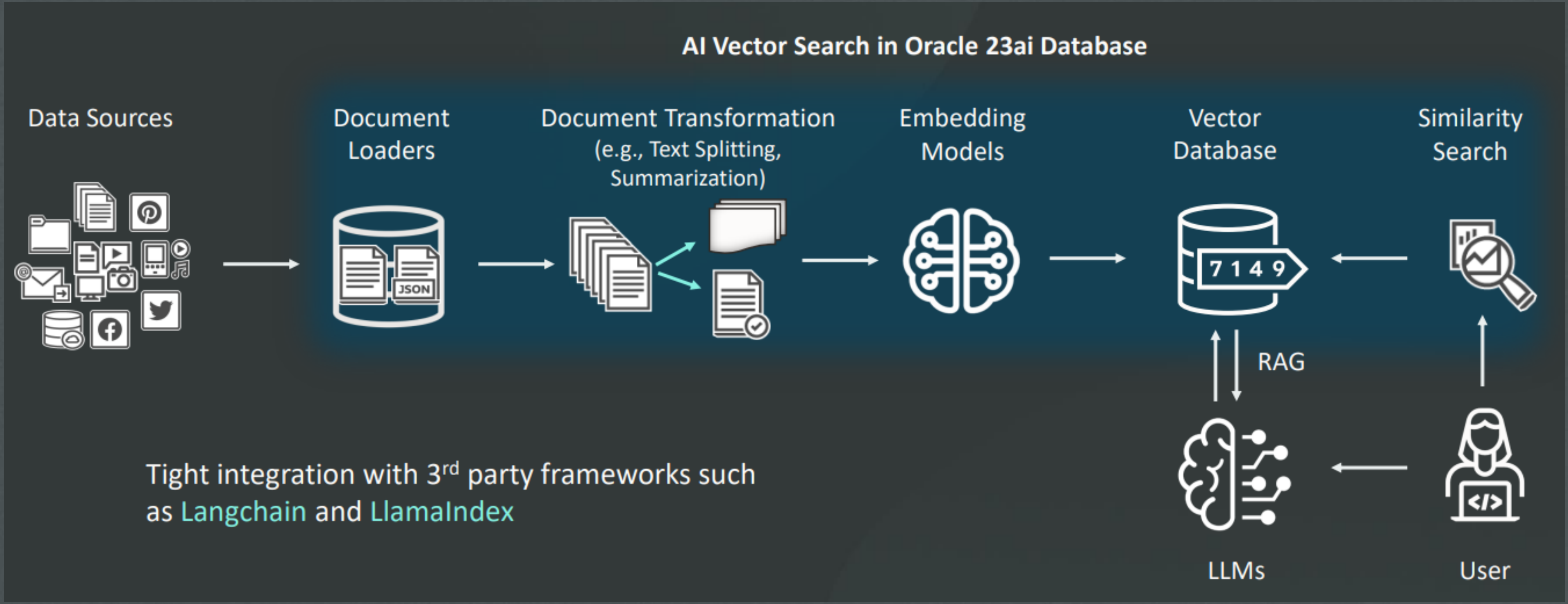


A vector is a sequence of numbers, called dimensions, used to capture the important “features” of the data

Produced by AI Deep Learning Models

Represent the semantic content of data, not the actual words in a document or pixels in an image

AI Vector Search powers Gen AI pipelines



AI Vector Search in Oracle 23ai Database

Data Sources

Document Loaders

Document Transformation (e.g., Text Splitting, Summarization)

Embedding Models

Vector Database

Similarity Search

RAG

LLMs

User

Tight integration with 3rd party frameworks such as Langchain and LlamaIndex

What are Vectors?

Vectors are a fundamental data structure in AI applications

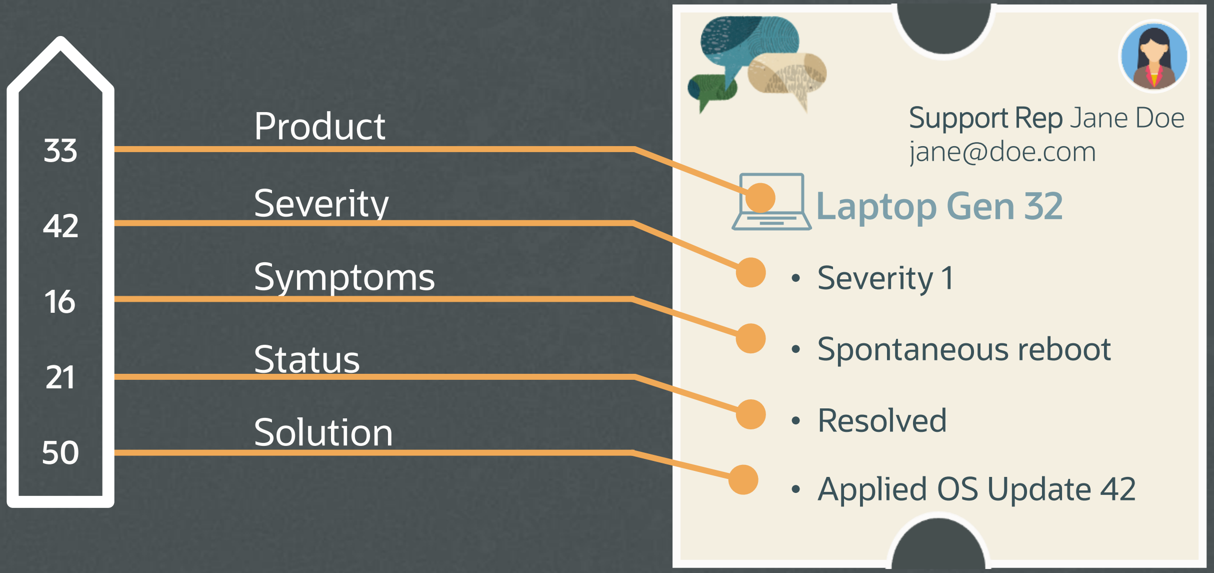
50 21 16 42 33

Example: The Vector for a Support Incident could be …

Vector

Features

Support Incident



33

42

16

21

50

Product

Severity

Symptoms

Status

Solution

Support Rep Jane Doe
jane@doe.com

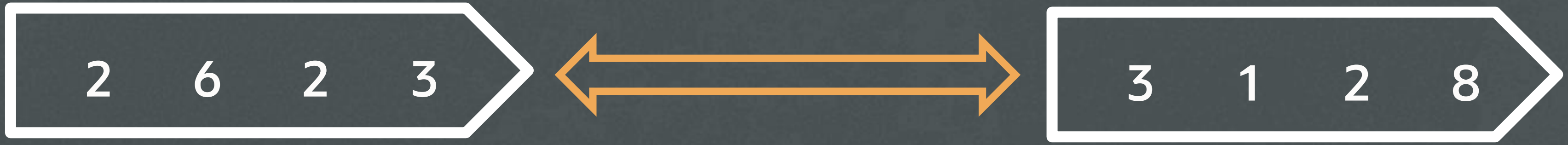
Laptop Gen 32

- Severity 1
- Spontaneous reboot
- Resolved
- Applied OS Update 42

Each dimension (number), represents a different feature of the support incident

Note: Features determined by actual AI models are much more complex

The main operation on vectors is the Mathematical Distance between them



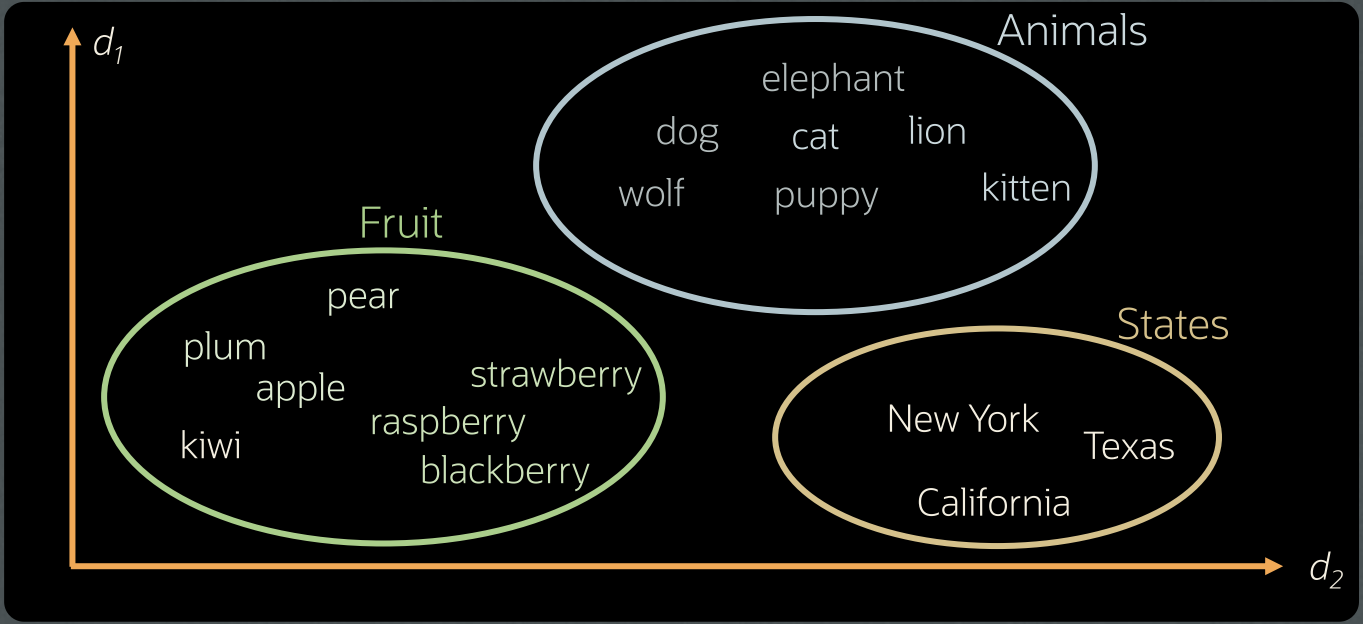
2 6 2 3

3 1 2 8

Distance (Euclidean Squared)
$= ((3-2)^2+(1-6)^2+(2-2)^2+(8-3)^2)$

There are many mathematical distance formulas

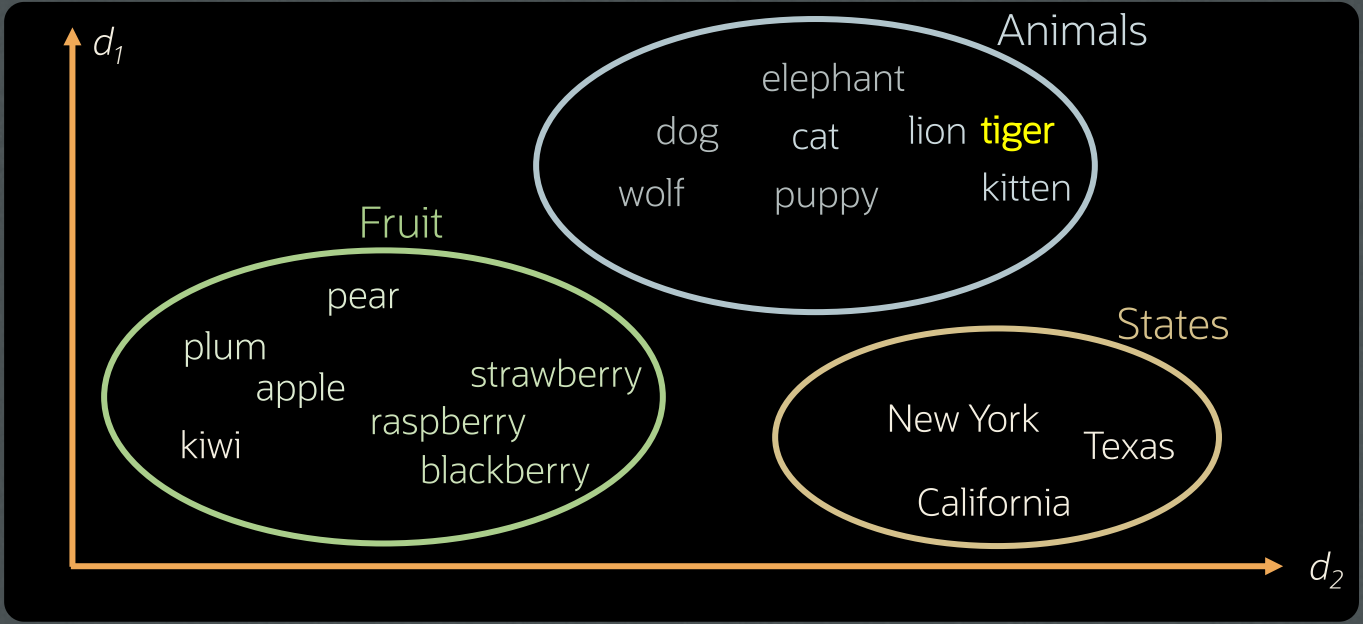
Similarity Property: The more similar two entities, the smaller the distance between their vectors



Documents and images also work the same way

Document vectors that represent similar content are closer than those representing dissimilar content

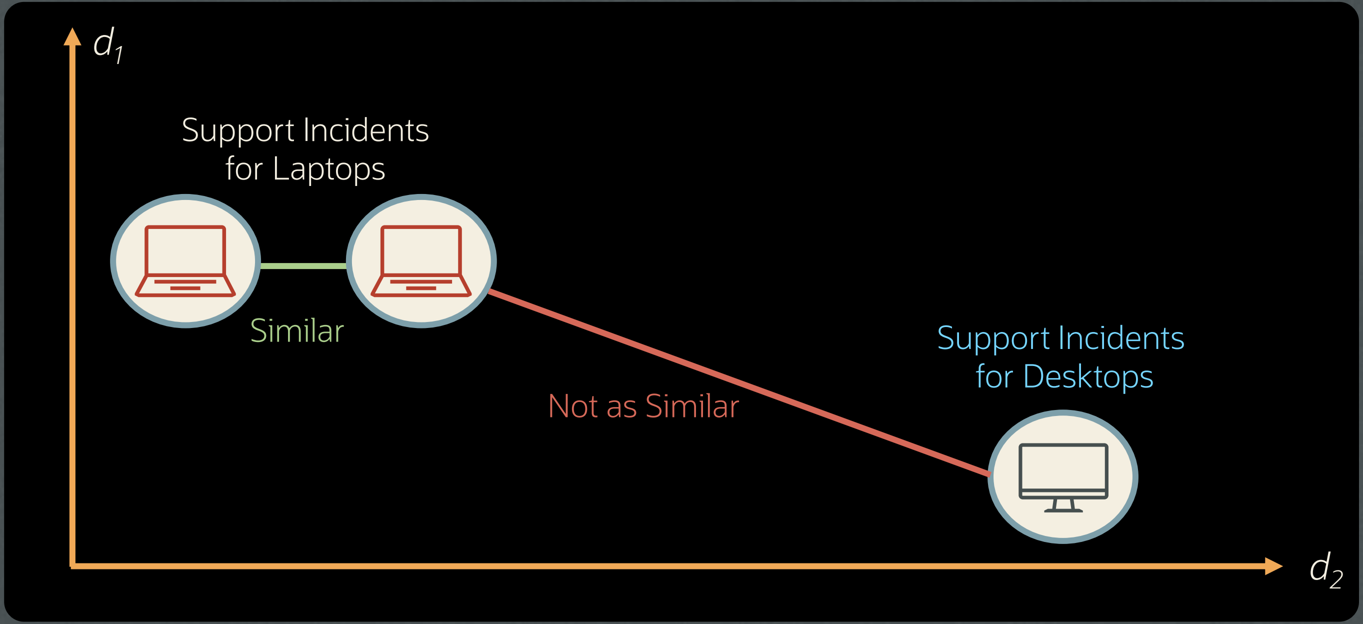
Similarity Property: The more similar two entities, the smaller the distance between their vectors



Documents and images also work the same way

Document vectors that represent similar content are closer than those representing dissimilar content

Support Incidents that are more similar also produce vectors that are closer together



The Similarity Property enables AI Vector Search

Search by Semantic Similarity rather than by Values

- Encode stored data as vectors using your chosen embedding model
- Encode search data as a vector using the same model
- Find the K nearest stored vectors by distance
- Return the data corresponding to the vectors

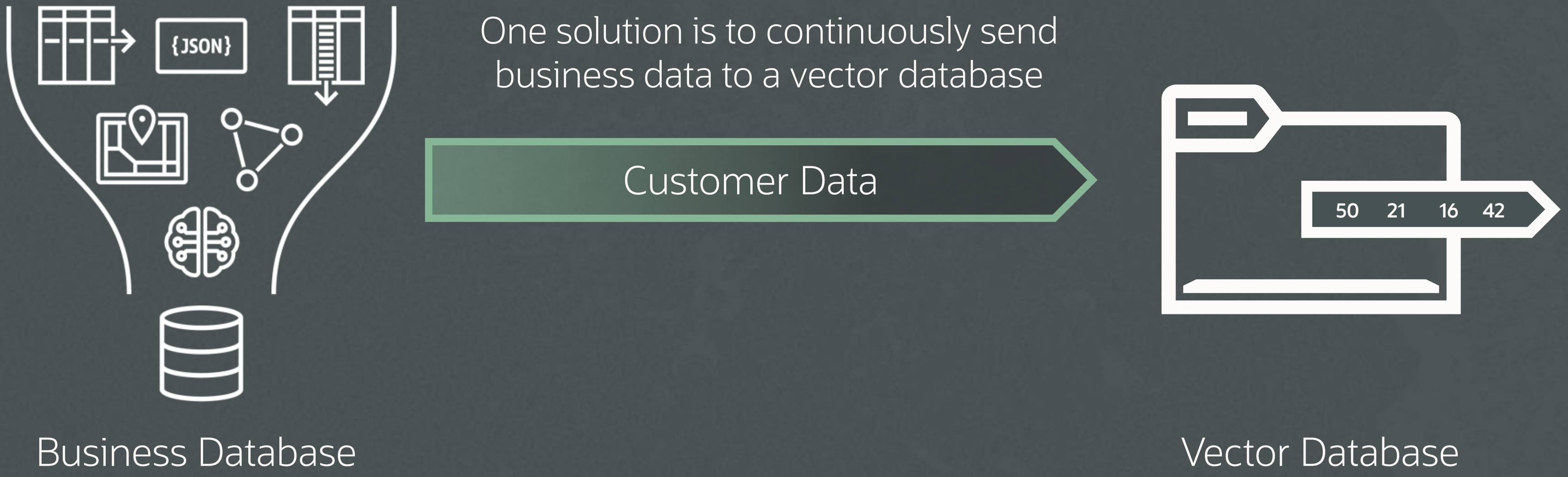
Why is Oracle adding AI Vector Search?

Aren't there dozens of Vector Databases out there already?

Goal: Extend business applications with semantic search for powerful new use cases

Searches on a combination of business data and semantic data requires both types of data to be queried together

One solution is to continuously send business data to a vector database

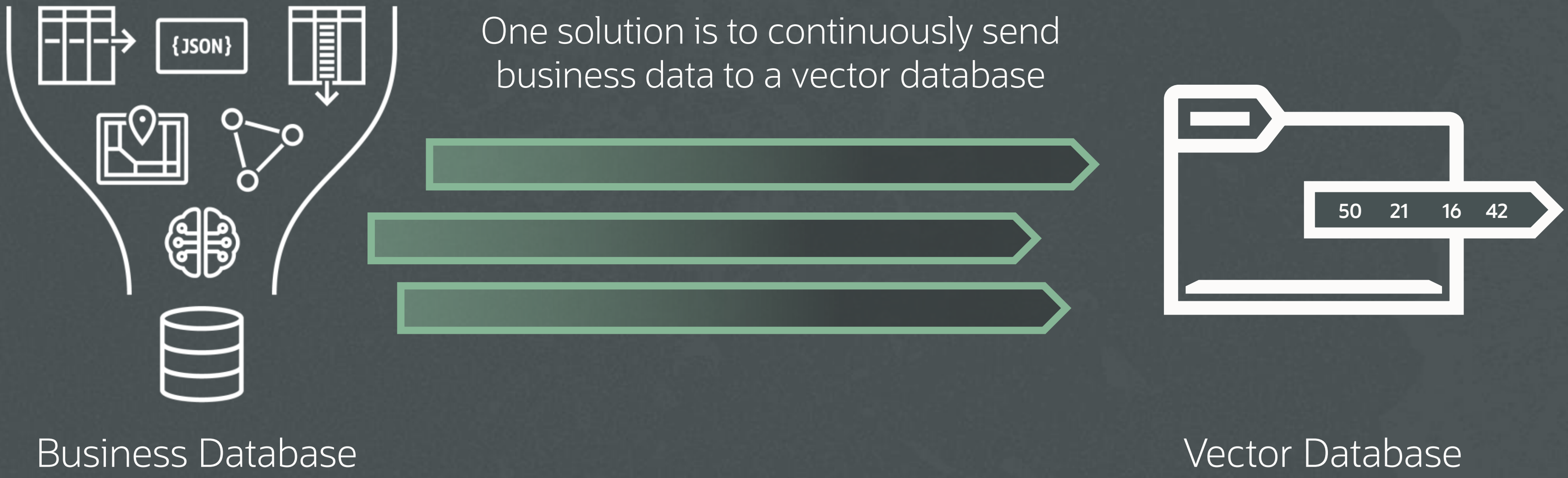


Customer Data

Business Database

Vector Database

Searches on a combination of business data and semantic data requires both types of data to be queried together



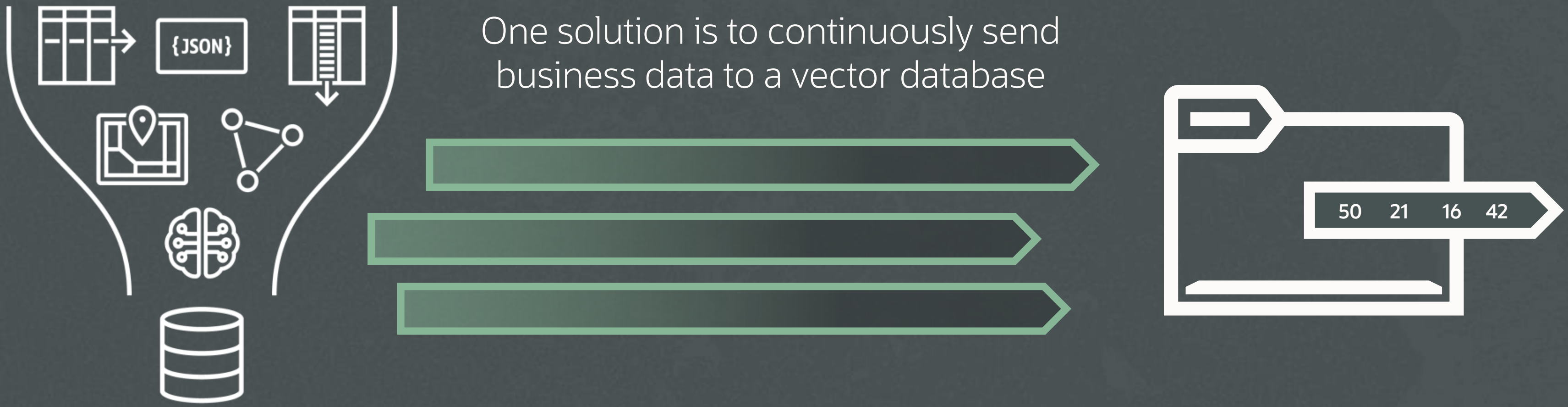
One solution is to continuously send business data to a vector database

Business Database

Vector Database

You need to send a lot of data since you can't predict the question that might be asked

Searches on a combination of business data and semantic data requires both types of data to be queried together



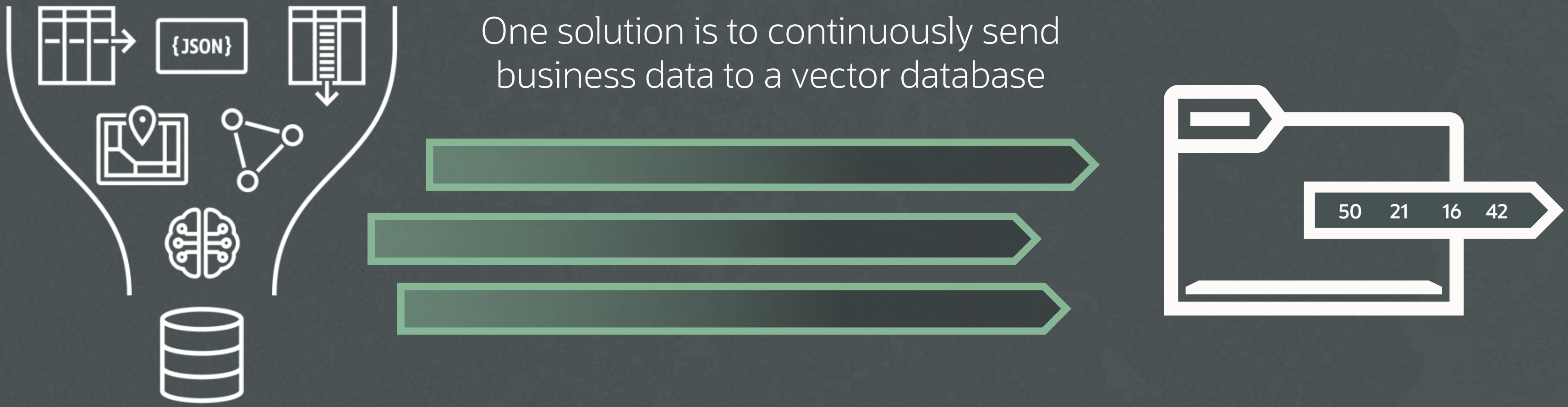
One solution is to continuously send business data to a vector database

Business Database

Vector Database

Causes data staleness, adds complexity, compromises security

Searches on a combination of business data and semantic data requires both types of data to be queried together



One solution is to continuously send business data to a vector database

Business Database

Vector Database

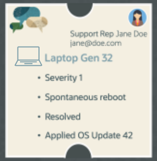
Enterprise DBs typically have an order of magnitude more sophisticated query capabilities, fault-tolerance, security, etc., than Vector DBs

Oracle AI Vector Search enables searches on business data to be combined with semantic searches

Enables combining AI vector search with search on business data about Customers and Products

Support Incident Search Example

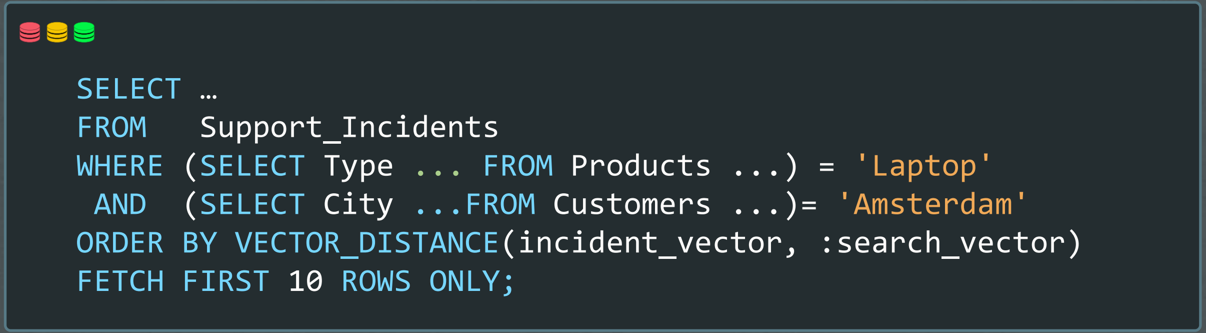
Find the top 10 matching incidents for a laptop reported by customers in Amsterdam



Support Rep Jane Doe
jane@doe.com

Laptop Gen 32

- Severity 1
- Spontaneous reboot
- Resolved
- Applied OS Update 42



```
SELECT …
FROM   Support_Incidents
WHERE (SELECT Type ... FROM Products ...) = 'Laptop'
 AND  (SELECT City ...FROM Customers ...)= 'Amsterdam'
ORDER BY VECTOR_DISTANCE(incident_vector, :search_vector)
FETCH FIRST 10 ROWS ONLY;
```

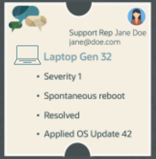
Enables combining AI vector search with search on business data about Customers and Products

Combines customer and product data, and AI search in a few lines of SQL!

A single integrated solution, all data fully consistent

Any developer can learn to use it in 5-minutes

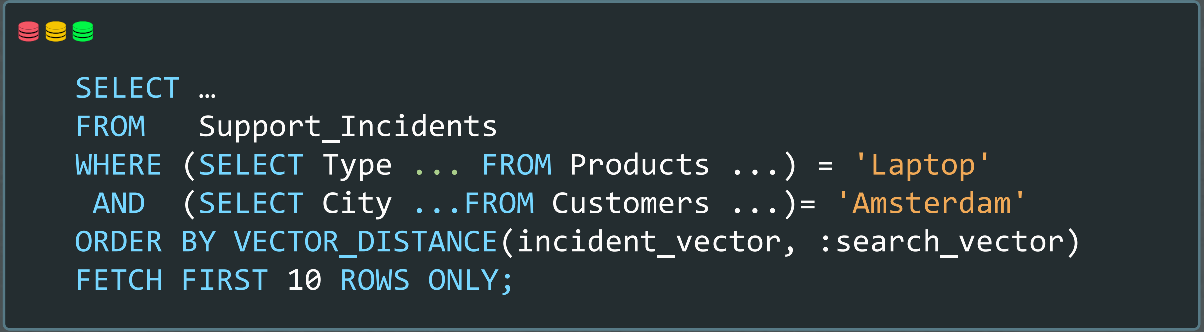
Find the top 10 matching incidents for a laptop reported by customers in Amsterdam



Support Rep Jane Doe
jane@doe.com

Laptop Gen 32

- Severity 1
- Spontaneous reboot
- Resolved
- Applied OS Update 42



```sql
SELECT …
FROM   Support_Incidents
WHERE (SELECT Type ... FROM Products ...) = 'Laptop'
 AND  (SELECT City ...FROM Customers ...)= 'Amsterdam'
ORDER BY VECTOR_DISTANCE(incident_vector, :search_vector)
FETCH FIRST 10 ROWS ONLY;
```

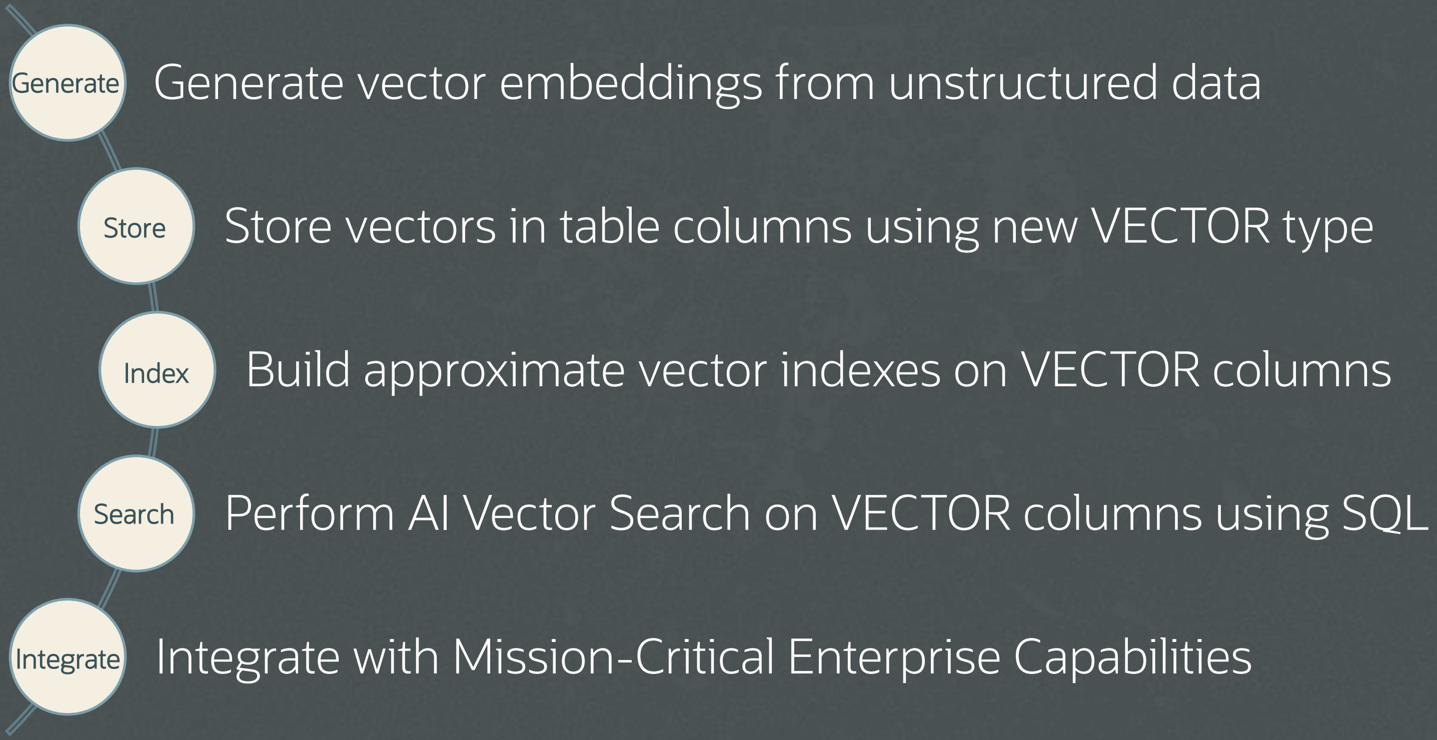
What do Enterprise Customers want from an Enterprise-Grade AI Vector Search solution?

Key Requirements for Enterprise AI Vector Search

- Completeness of solution
- Simplicity of solution
- Sophisticated query support
- Massive scale
- Enterprise-Grade Fault Tolerance

Industrial strength

Oracle AI Vector Search | Summary



Generate — Generate vector embeddings from unstructured data

Store — Store vectors in table columns using new VECTOR type

Index — Build approximate vector indexes on VECTOR columns

Search — Perform AI Vector Search on VECTOR columns using SQL

Integrate — Integrate with Mission-Critical Enterprise Capabilities

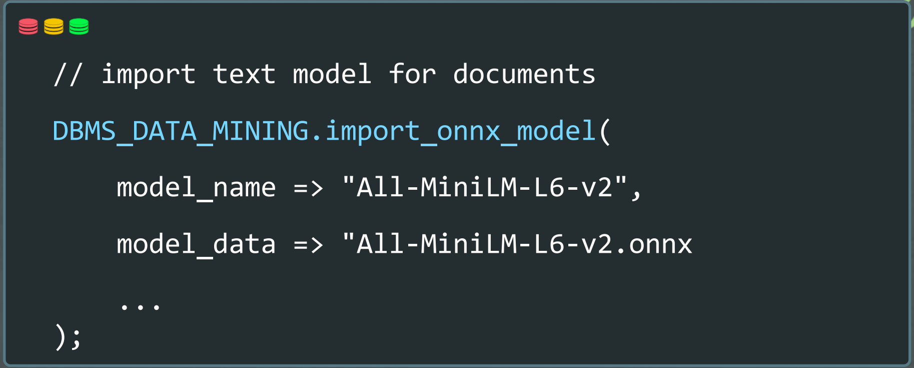
Generate Vectors

New VECTOR_EMBEDDING() function to generate vectors

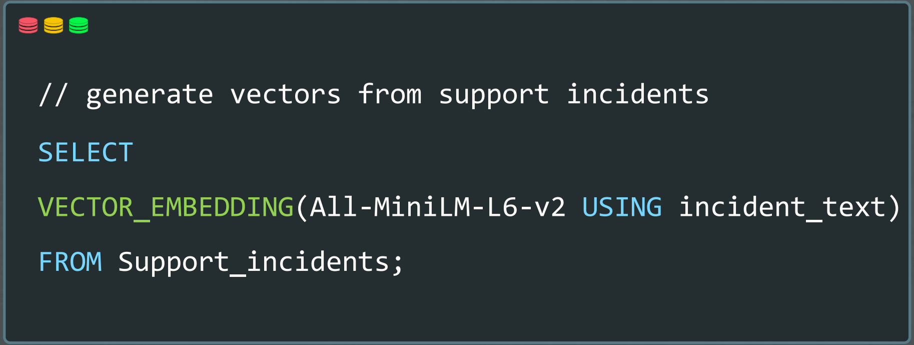
Completeness: Many customers want to be able to generate vectors within the database

Oracle Database supports the Open Neural Net Exchange (ONNX) framework to import models

The VECTOR_EMBEDDING() function can then generate vectors for unstructured data using the imported model



```
// import text model for documents
DBMS_DATA_MINING.import_onnx_model(
    model_name => "All-MiniLM-L6-v2",
    model_data => "All-MiniLM-L6-v2.onnx
    ...
);
```

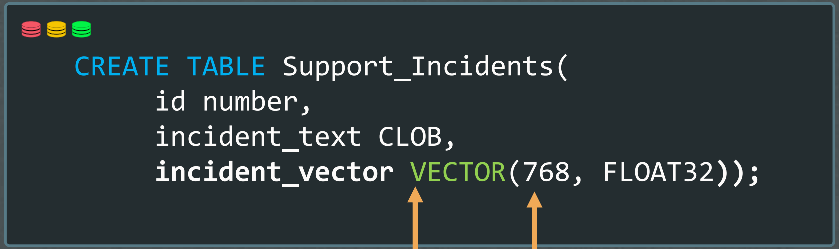


```
// generate vectors from support incidents
SELECT
VECTOR_EMBEDDING(All-MiniLM-L6-v2 USING incident_text)
FROM Support_incidents;
```

Store (& Process) Vectors

VECTOR Datatype to store and process vectors

New VECTOR datatype



```
CREATE TABLE Support_Incidents(
     id number,
     incident_text CLOB,
     incident_vector VECTOR(768, FLOAT32));
```

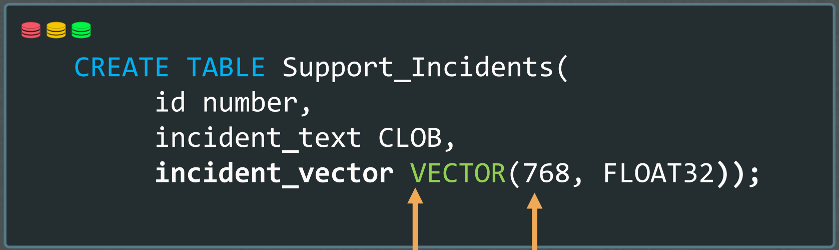
Optional
# of dimensions

Optional
format

Format for dimension values can be `FLOAT32`, `FLOAT64`, and `INT8`

VECTOR Datatype to store and process vectors

New VECTOR datatype



```
   CREATE TABLE Support_Incidents(
        id number,
        incident_text CLOB,
        incident_vector VECTOR(768, FLOAT32));
```

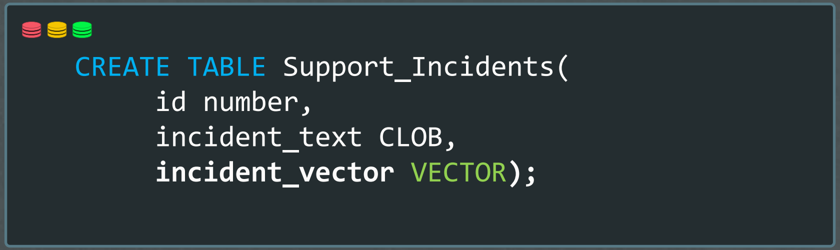
Optional
# of dimensions

Optional
format

Format for dimension values can be `FLOAT32`, `FLOAT64`, and `INT8`

Additional formats coming …

Alternatively, you can simply specify the column as a VECTOR



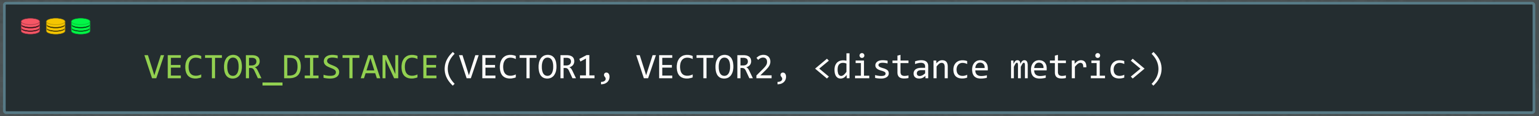
```
   CREATE TABLE Support_Incidents(
        id number,
        incident_text CLOB,
        incident_vector VECTOR);
```

Why is this needed? Flexibility:

- Embedding models are changing constantly but the schema can stay the same
- Support vectors from multiple embedding models in the same column

Vector Processing Operators

The main operation on vectors is to find how similar they are



```
VECTOR_DISTANCE(VECTOR1, VECTOR2, <distance metric>)
```

Different embedding models can use different distance metrics like Euclidean, cosine similarity, dot product, etc

All embedding models must obey the same similarity property

E.g. `VECTOR_DISTANCE(<Tiger Vec>, <Lion Vec>) < VECTOR_DISTANCE(<Tiger Vec>, <Apple Vec>)`

Vectors Indexes

Approximate Vector Indexes

Exact search for top-K matches will be 100% accurate but very slow

New vector indexes trade-off some search accuracy for 100x speed

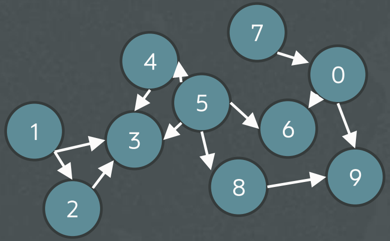
Neighbor Graph Vector Index – Graph-based index where vertices represent vectors and edges between vertices represent *similarity*

- In-Memory only index - highly efficient for both accuracy and speed

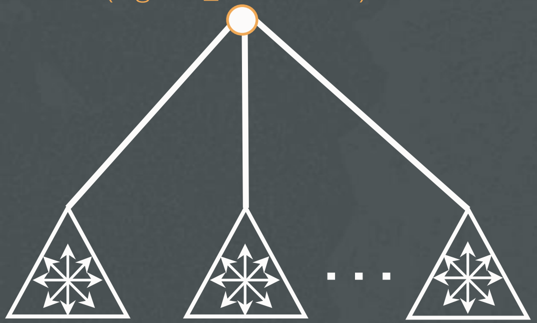
Neighbor Partition Vector Index – Partition-based index with vectors clustered into table partitions based on *similarity*

- Efficient scale-out index for unlimited data size

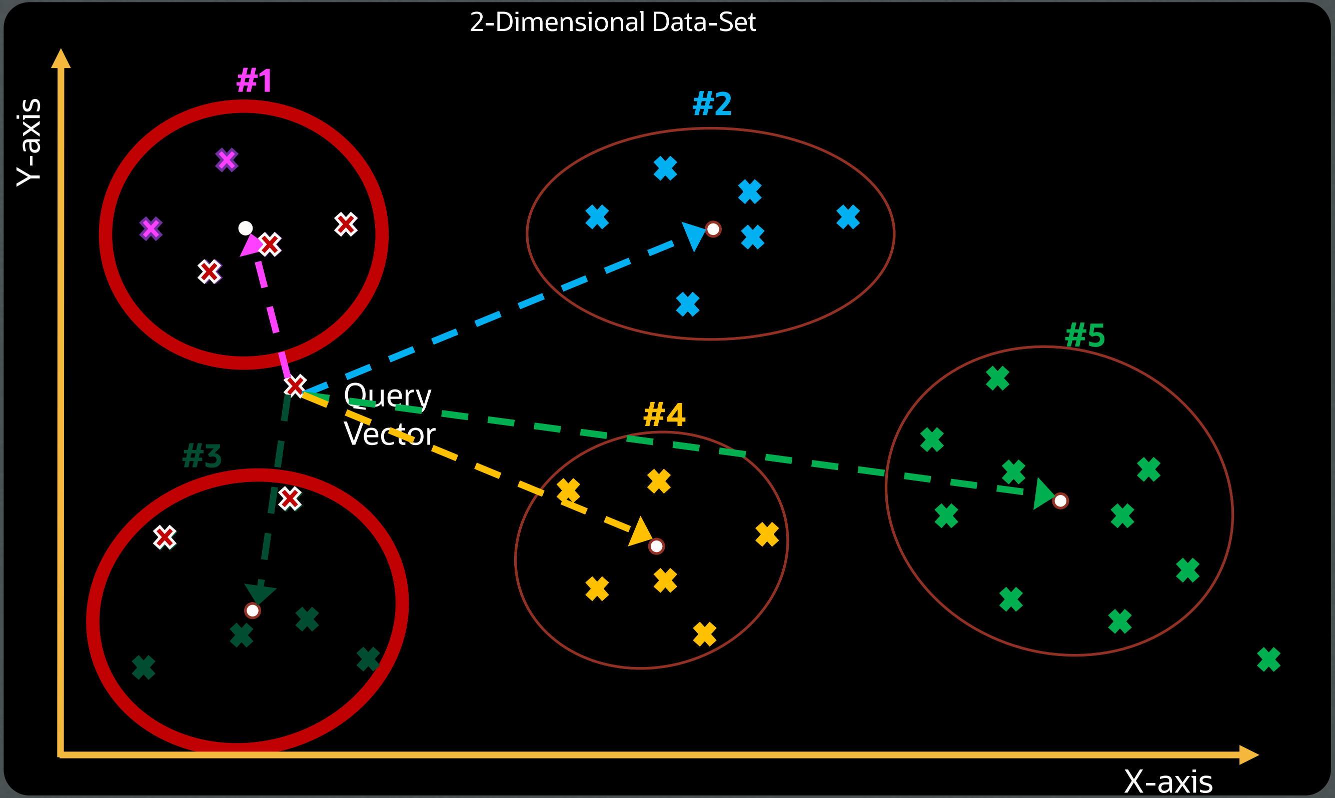
Graph Vector Index
(e.g. HNSW Index)



Partition Vector Index
(e.g. IVF_FLAT index)



Neighbor Partition Vector Index – Search



1. Group vectors into partitions using OML's K-means clustering algo (K = 5)

2. Compute distance from query vector to each partition's centroids

3. Identify the 2 nearest partitions

4. Compute distance from query vector to all points in Cluster #1 and #3 to find Top 5 closest matches (shown in red)

Graph Vector Index

Multi-layer in-memory graph index

In-memory index designed for speed and accuracy

Considered the “B+ tree index for Vectors”

Construction

The lowest layer of the graph has all the vectors

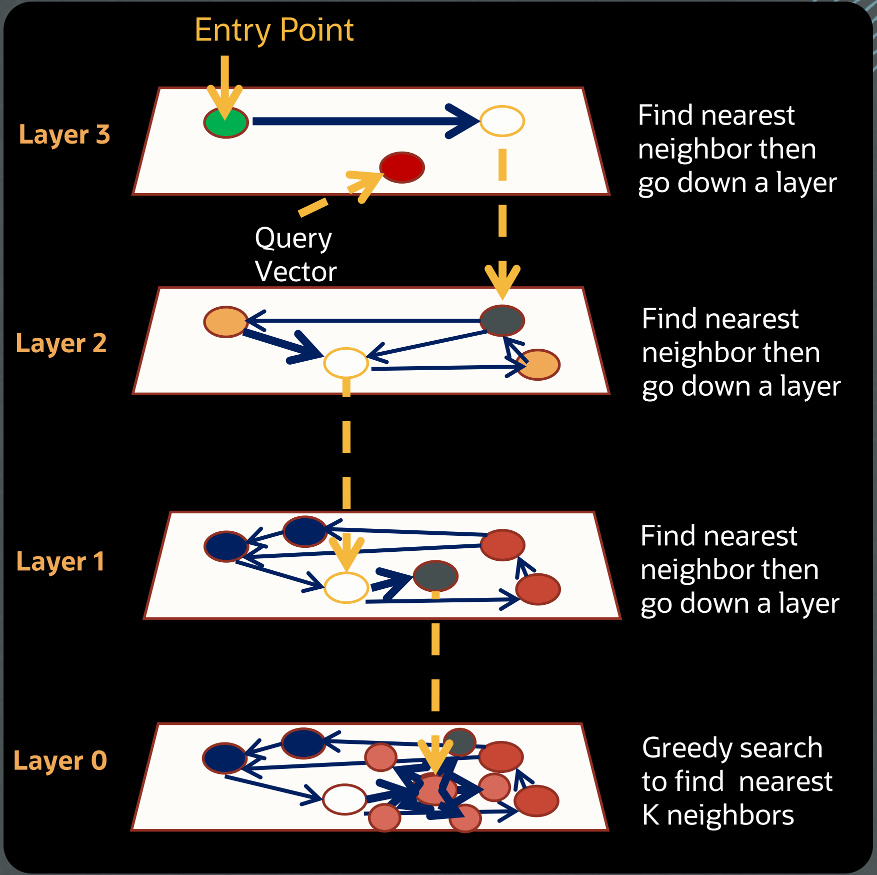
Higher layers have a decaying fraction of vectors

Vectors are connected based on similarity

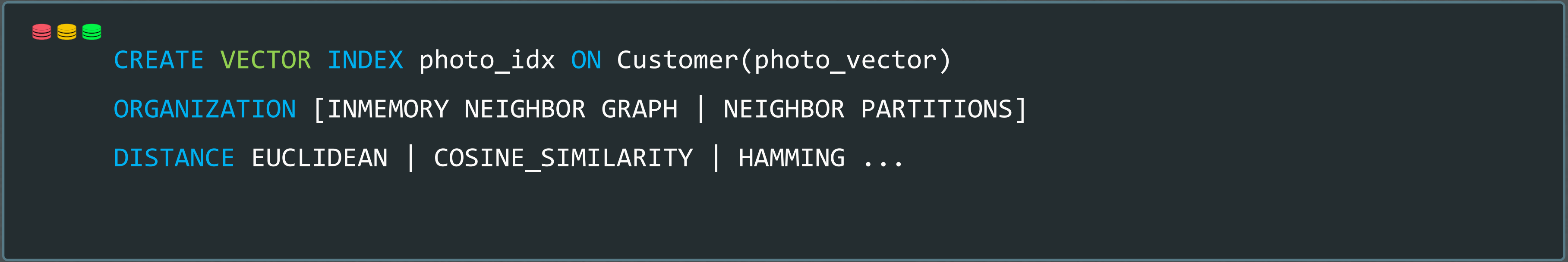
Search

Search begins from the top layer
When the nearest vector is found, the search continues in the layer below

The search completes in the lowest layer when the Top K nearest vectors to the query vector are found


Entry Point
Layer 3
Find nearest neighbor then go down a layer
Query Vector
Layer 2
Find nearest neighbor then go down a layer
Layer 1
Find nearest neighbor then go down a layer
Layer 0
Greedy search to find nearest K neighbors

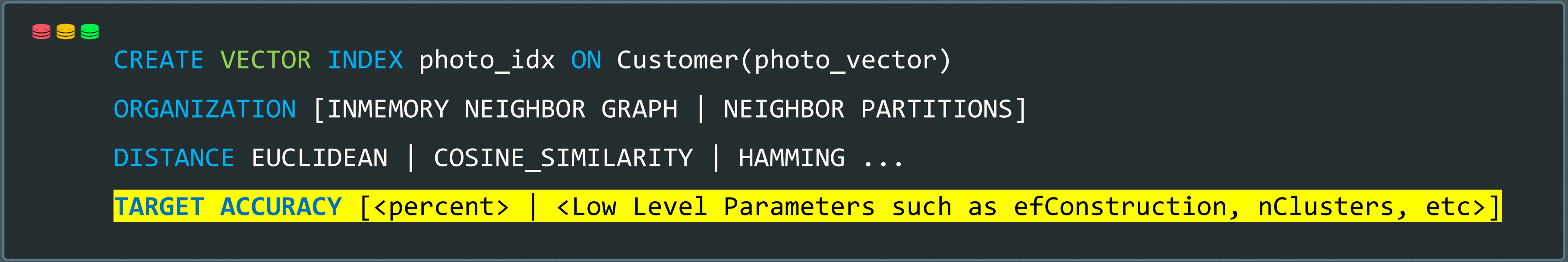
Vector Index Creation



```
CREATE VECTOR INDEX photo_idx ON Customer(photo_vector)
ORGANIZATION [INMEMORY NEIGHBOR GRAPH | NEIGHBOR PARTITIONS]
DISTANCE EUCLIDEAN | COSINE_SIMILARITY | HAMMING ...
```

`ORGANIZATION`: If data fits in-memory, use `INMEMORY NEIGHBOR GRAPH` else use `NEIGHBOR PARTITIONS`

Vector Index Creation – TARGET ACCURACY



```
CREATE VECTOR INDEX photo_idx ON Customer(photo_vector)
ORGANIZATION [INMEMORY NEIGHBOR GRAPH | NEIGHBOR PARTITIONS]
DISTANCE EUCLIDEAN | COSINE_SIMILARITY | HAMMING ...
TARGET ACCURACY [<percent> | <Low Level Parameters such as efConstruction, nClusters, etc>]
```

ORGANIZATION: If data fits in-memory, use INMEMORY NEIGHBOR GRAPH else use NEIGHBOR PARTITIONS

TARGET ACCURACY: Specify the default accuracy (recall) when the index is used

- Simplicity: Easiest for users to specify accuracy as a percent instead of index algorithm parameters
- Continuous calibration used to map target accuracy to low level parameter values
- Specialists can still specify low-level parameters if they want (Some customers want this

Querying Vectors

Vector Query

A new APPROXIMATE keyword in the Row Limiting (FETCH) clause indicates similarity search:

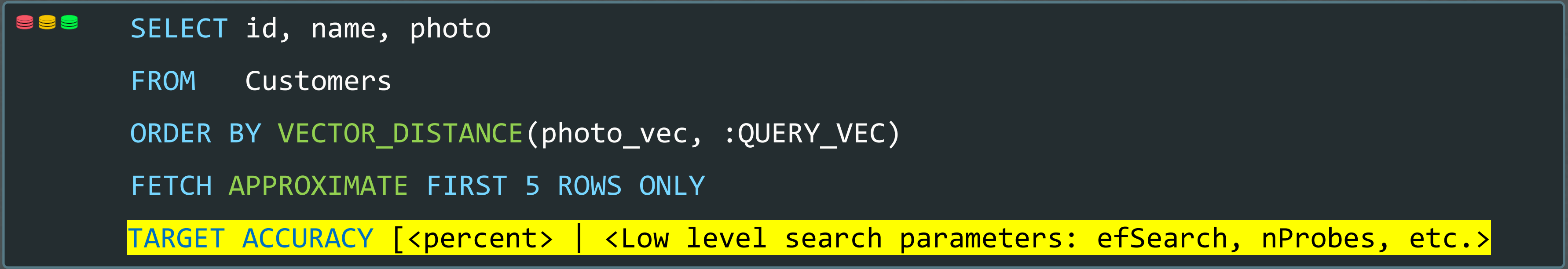
Find the top 5 Customers by similarity with a search photo vector:

```
SELECT id, name, photo
FROM   Customers
ORDER BY VECTOR_DISTANCE(photo_vec, :QUERY_VEC)
FETCH APPROXIMATE FIRST 5 ROWS ONLY
```

Vector Query – TARGET ACCURACY

A new APPROXIMATE keyword in the Row Limiting (FETCH) clause indicates similarity search:

Find the top 5 Customers by similarity with a search photo vector:



```
SELECT id, name, photo
FROM   Customers
ORDER BY VECTOR_DISTANCE(photo_vec, :QUERY_VEC)
FETCH APPROXIMATE FIRST 5 ROWS ONLY
TARGET ACCURACY [<percent> | <Low level search parameters: efSearch, nProbes, etc.>
```

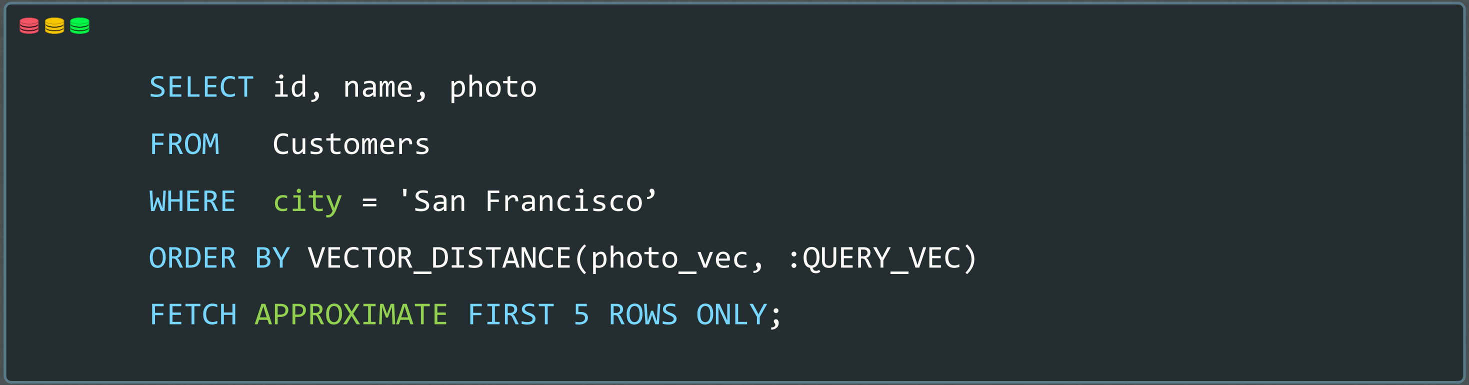
TARGET ACCURACY: Specify the desired accuracy (recall), if different from index accuracy

- Simplicity: Easiest for users to specify accuracy as a percent instead of index search parameters
- Continuous calibration used to map target accuracy to low-level search parameter values
- Specialists can still specify low-level parameters if they want (Some customers want this)

Vector Query – With Attribute Filters

Vector similarity search queries can easily be combined with relational filters, joins, e.g.

Find the top 5 Customers by similarity with a search photo vector who live in San Francisco:

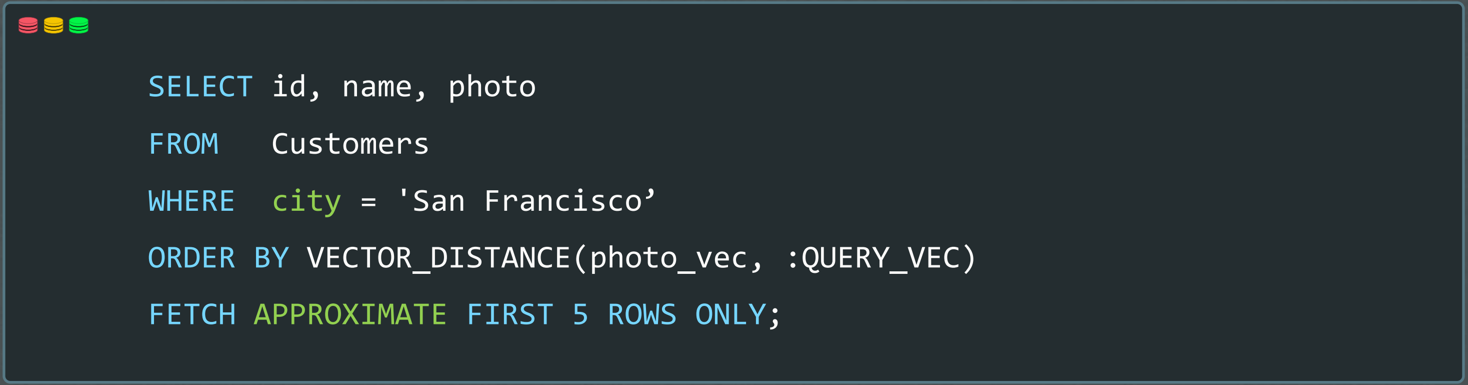


```
SELECT id, name, photo
FROM   Customers
WHERE  city = 'San Francisco’
ORDER BY VECTOR_DISTANCE(photo_vec, :QUERY_VEC)
FETCH APPROXIMATE FIRST 5 ROWS ONLY;
```

Vector Query – With Attribute Filters

Vector similarity search queries can easily be combined with relational filters, joins, e.g.

Find the top 5 Customers by similarity with a search photo vector who live in San Francisco:



```
SELECT id, name, photo
FROM   Customers
WHERE  city = 'San Francisco’
ORDER BY VECTOR_DISTANCE(photo_vec, :QUERY_VEC)
FETCH APPROXIMATE FIRST 5 ROWS ONLY;
```

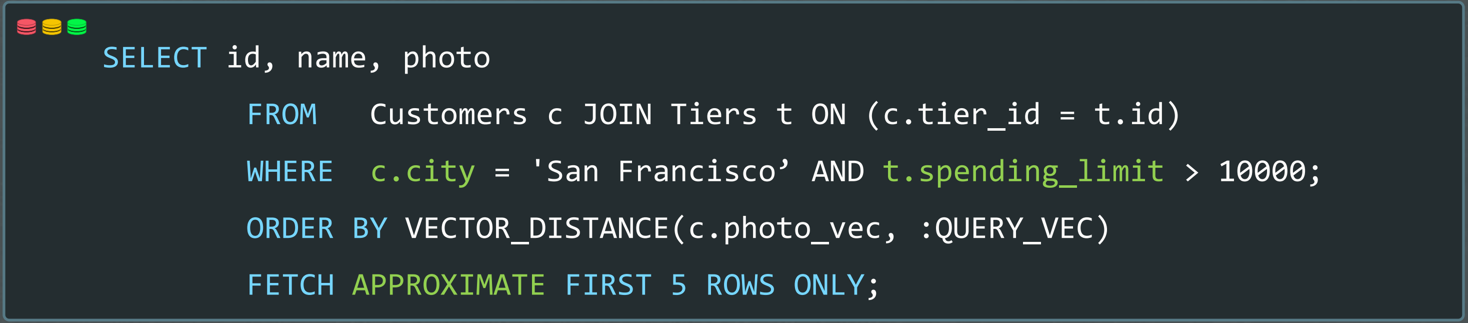
Optimizer chooses the best strategy based on filter selectivity:

- PREFILTER: For high selectivity, apply the filter first, construct a bloom filter, and apply it during the index search
- INFILTER: For medium selectivity, apply the filter as index is being searched
- POST FILTER: For low selectivity, first determine top n*K matches, then apply filter and return top K

Vector Query – With Attribute Filters and Joins

Vector similarity search queries can easily be combined with relational filters, joins, e.g.

Find the top 5 Customers by similarity with a search photo vector who live in San Francisco and who have credit limits greater than $10k based on their tier status:



```
    SELECT id, name, photo
            FROM   Customers c JOIN Tiers t ON (c.tier_id = t.id)
            WHERE  c.city = 'San Francisco’ AND t.spending_limit > 10000;
            ORDER BY VECTOR_DISTANCE(c.photo_vec, :QUERY_VEC)
            FETCH APPROXIMATE FIRST 5 ROWS ONLY;
```

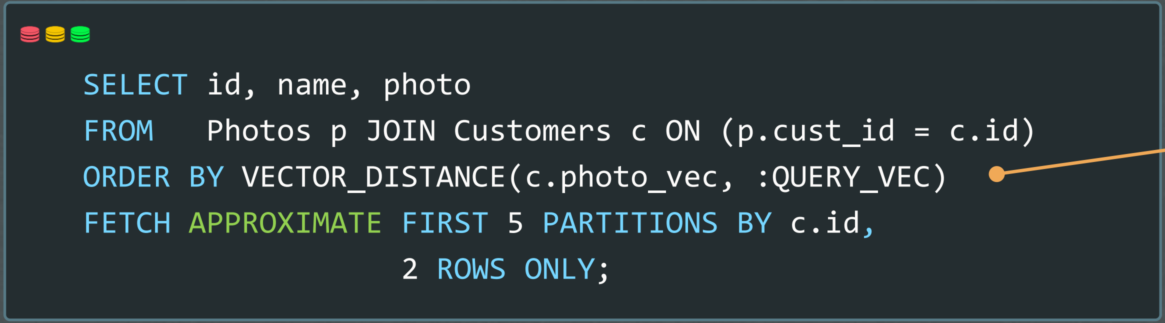
Most enterprise data is normalized, so this is an essential capability

Vector Query – Joins with Multi-Vector Grouping

Multi-Vector group by is a scenario in which one entity has multiple vectors, for example:

- A document may be divided into different chunks, each with a different vector
- A person may have multiple photos, each with a different vector

E.g. Find the top 5 Customers and their top 2 matching photos by similarity with a search photo



```
SELECT id, name, photo
FROM   Photos p JOIN Customers c ON (p.cust_id = c.id)
ORDER BY VECTOR_DISTANCE(c.photo_vec, :QUERY_VEC)
FETCH APPROXIMATE FIRST 5 PARTITIONS BY c.id,
                  2 ROWS ONLY;
```

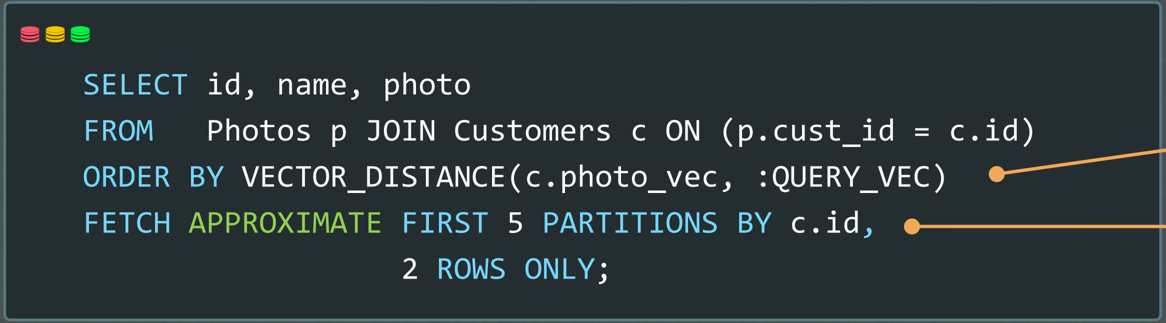
Order join by distance
to search photo

Vector Query – Joins with Multi-Vector Grouping

Multi-Vector group by is a scenario in which one entity has multiple vectors, for example:

- A document may be divided into different chunks, each with a different vector
- A person may have multiple photos, each with a different vector

E.g. Find the top 5 Customers and their top 2 matching photos by similarity with a search photo



```
SELECT id, name, photo
FROM   Photos p JOIN Customers c ON (p.cust_id = c.id)
ORDER BY VECTOR_DISTANCE(c.photo_vec, :QUERY_VEC)
FETCH APPROXIMATE FIRST 5 PARTITIONS BY c.id,
                  2 ROWS ONLY;
```

Order join by distance to search photo

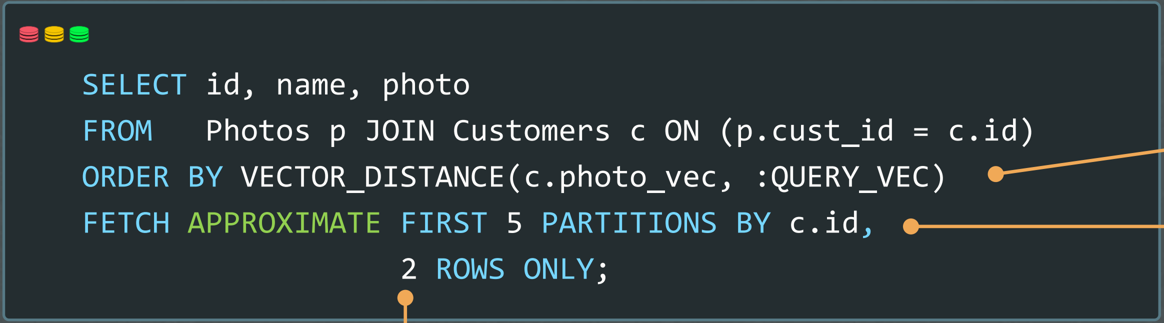
Partition join by Customer ID and find the 5 partitions with closest matching photos

Vector Query – Joins with Multi-Vector Grouping

Multi-Vector group by is a scenario in which one entity has multiple vectors, for example:

- A document may be divided into different chunks, each with a different vector
- A person may have multiple photos, each with a different vector

E.g. Find the top 5 Customers and their top 2 matching photos by similarity with a search photo



```
SELECT id, name, photo
FROM   Photos p JOIN Customers c ON (p.cust_id = c.id)
ORDER BY VECTOR_DISTANCE(c.photo_vec, :QUERY_VEC)
FETCH APPROXIMATE FIRST 5 PARTITIONS BY c.id,
                  2 ROWS ONLY;
```

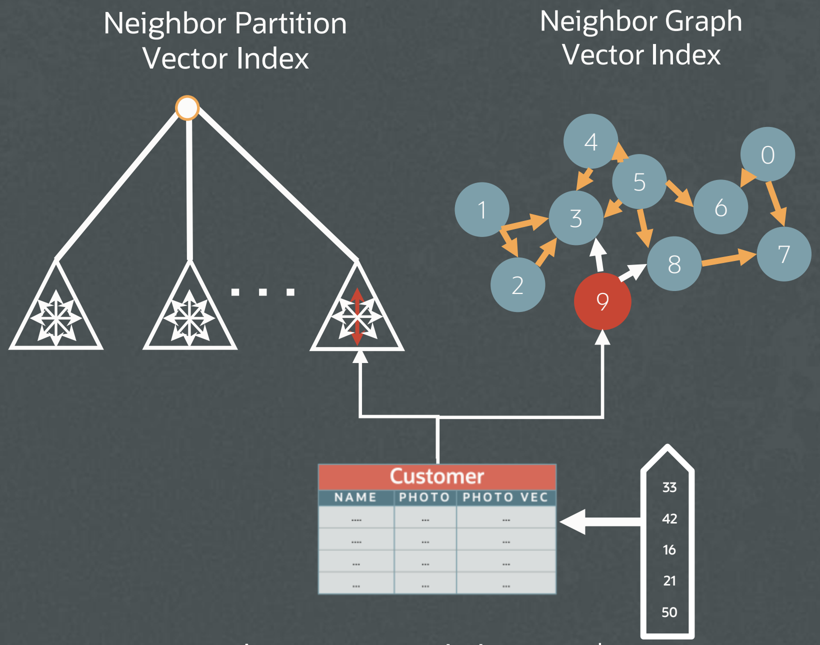
Order join by distance to search photo

Partition join by Customer ID and find the 5 partitions with closest matching photos

Within each partition find the 2 closest photos

Integrate with Mission-Critical Enterprise Capabilities

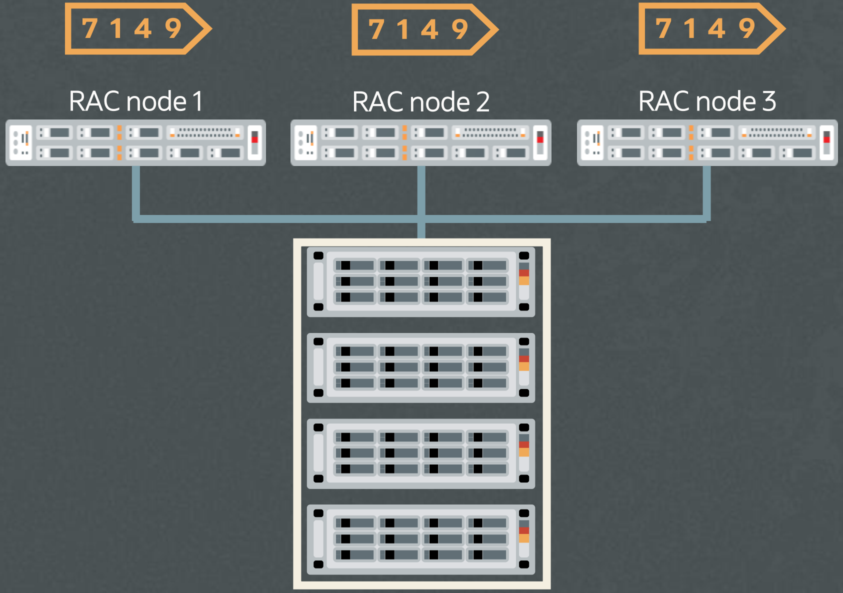
AI Vector Search | Transactions



A new vector is inserted

Oracle's AI Vector Search Indexes maintain transactional consistency with DML activity

AI Vector Search | Scale-Out with Real Application Clusters

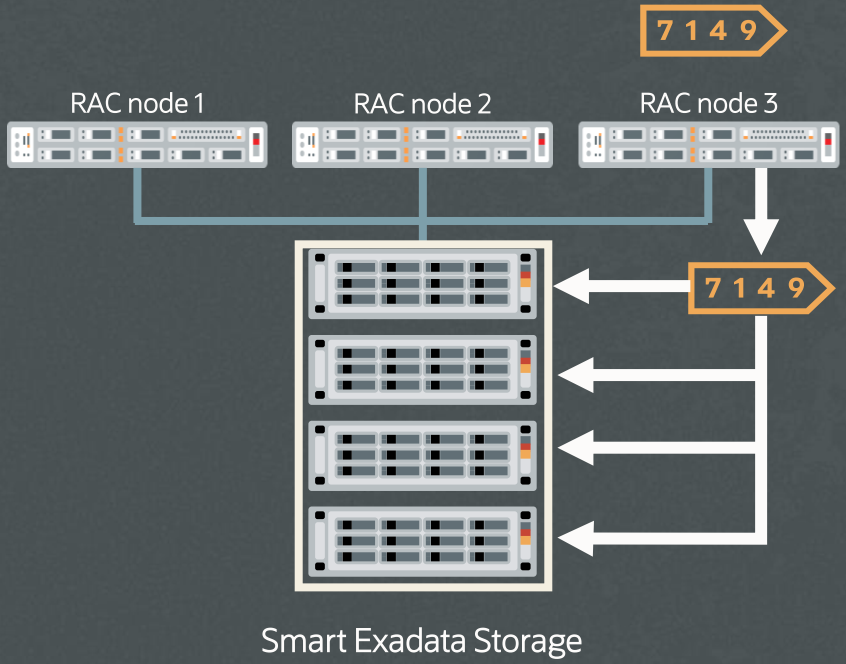


Smart Exadata Storage

AI Vector search transparently scales vector processing across the compute nodes in a RAC cluster

With full data consistency

AI Vector Search | Scale-Out with Exadata Smart Storage



Oracle AI Vector search can be transparently offloaded to smart Exadata storage for faster search

AI Vector Search | Scale-Out with Partitioning

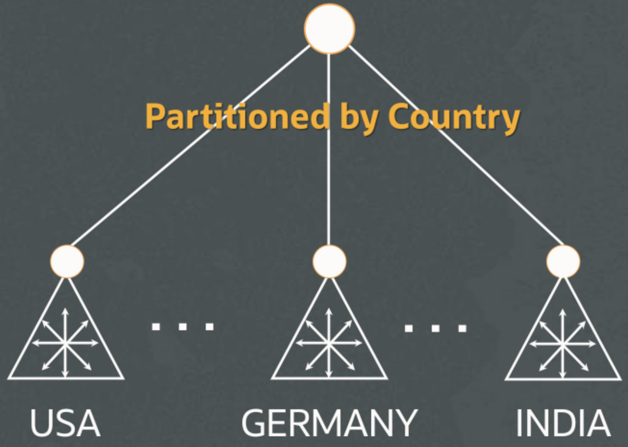
Partition or Sub-Partition by Relational Attributes

Build Vector index on each Partition

Filter by Partition key, perform Vector Search on qualifying partitions only

Can be 1000x faster

Vector index of customer photos



"Find top 10 matching customers in USA"

AI Vector Search | Scale-Out with Sharding

Database native sharding enables planet scale vector search

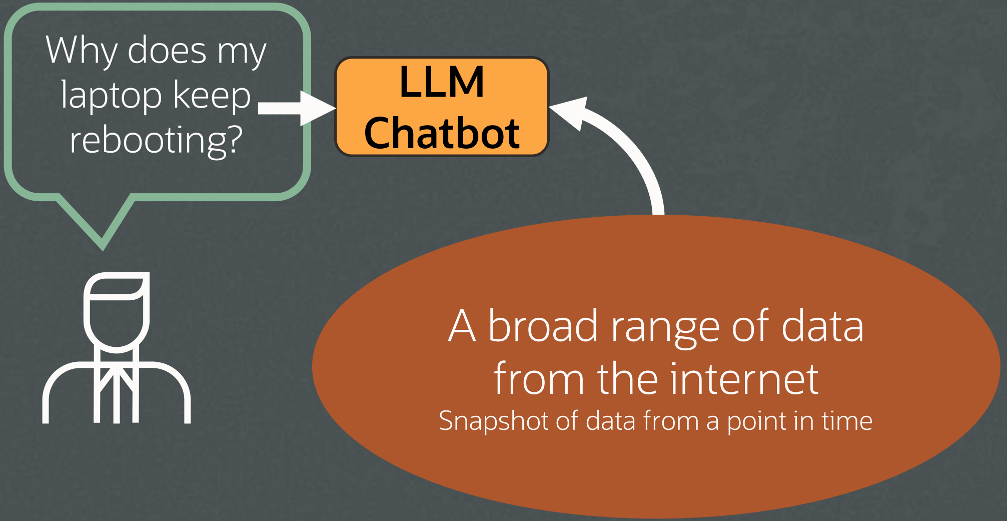
Sharding can be used both for unlimited scale or data sovereignty

Build Vector index on each independent shard database

Oracle AI Vector Search also allows you to interact with business data using Natural Language

Role of Vector Databases in Generative AI

LLMs are frozen on a past snapshot of the internet with no access to private enterprise data



Why does my laptop keep rebooting?

LLM Chatbot

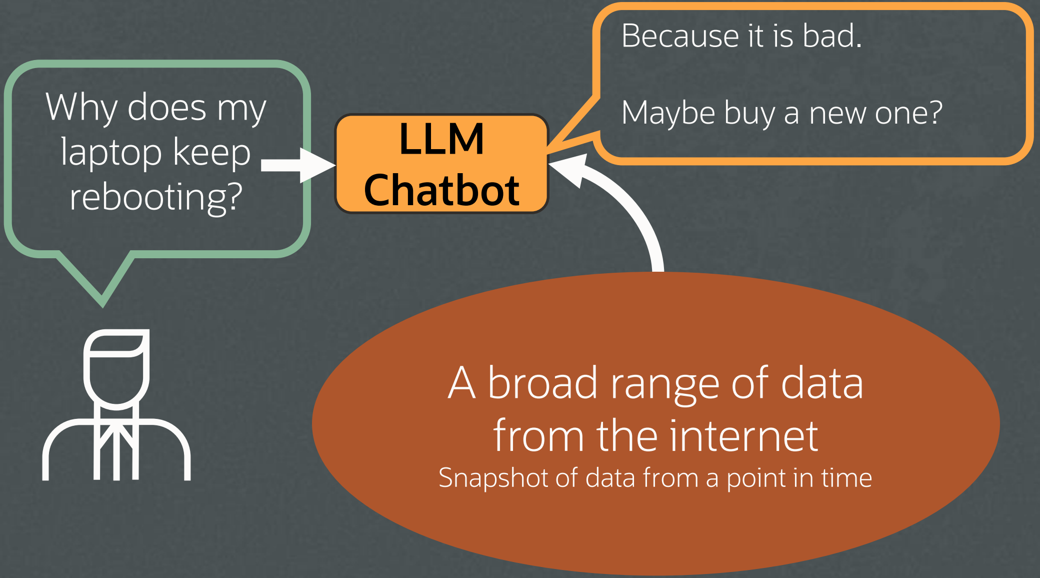
A broad range of data from the internet

Snapshot of data from a point in time

Role of Vector Databases in Generative AI

LLMs are frozen on a past snapshot of the internet with no access to private enterprise data

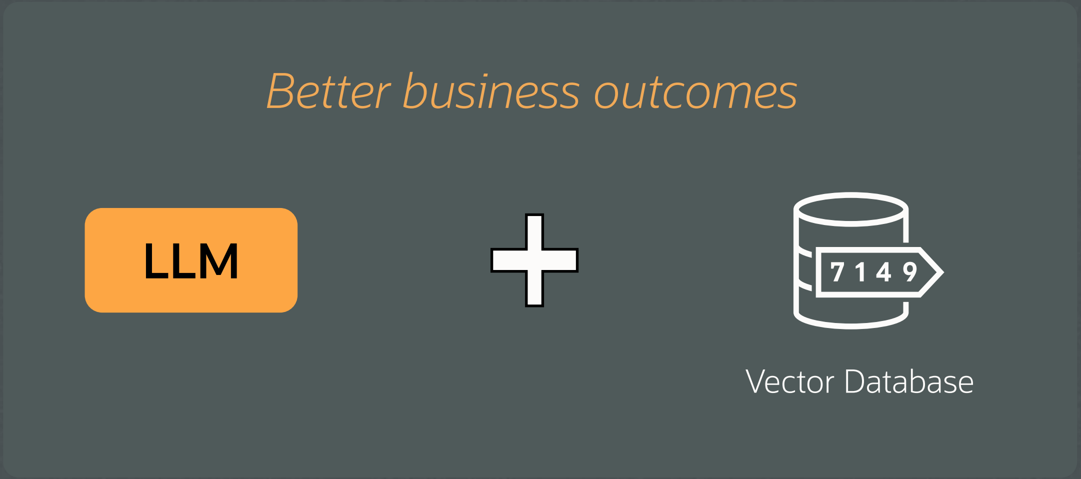
LLMs by themselves therefore often provide poor-quality responses to support questions


Why does my laptop keep rebooting?
LLM Chatbot
Because it is bad.
Maybe buy a new one?
A broad range of data from the internet
Snapshot of data from a point in time

Role of Vector Databases in Generative AI

Provide enterprise content to enhance LLM interactions (retrieval augmentation)

Avoid having to train LLMs on sensitive enterprise data (not secure, expensive)



Better business outcomes

LLM

+

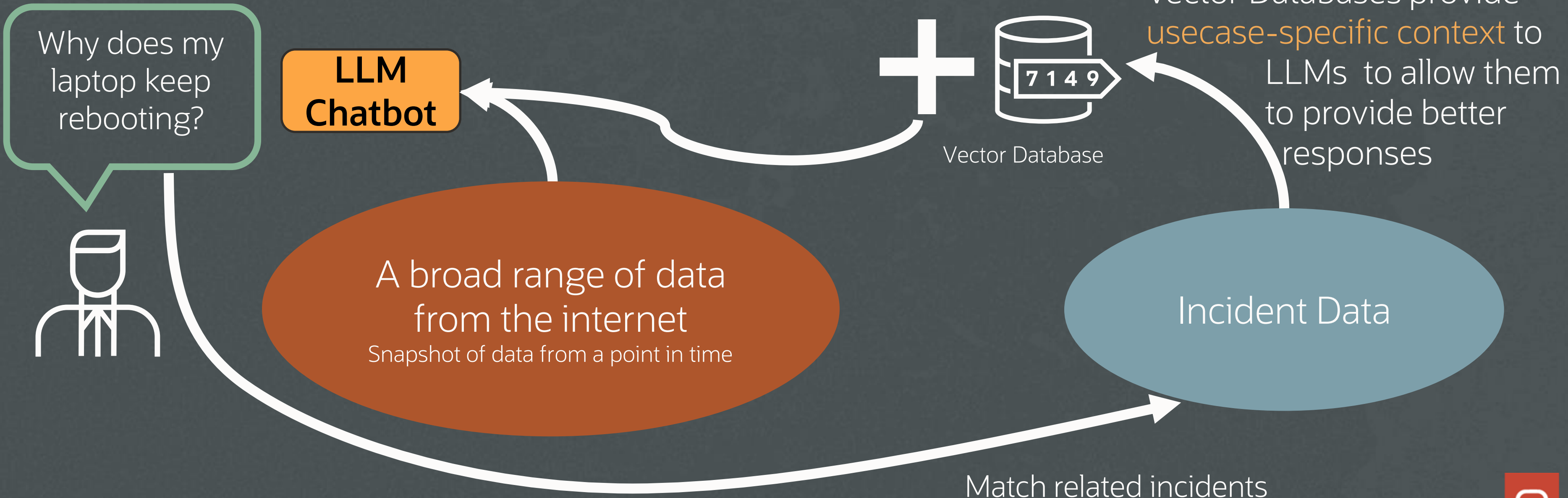
7 1 4 9

Vector Database

Role of Vector Databases in Generative AI

When augmented with enterprise information they provide better answers

Known as Retrieval Augmented Generation (RAG)

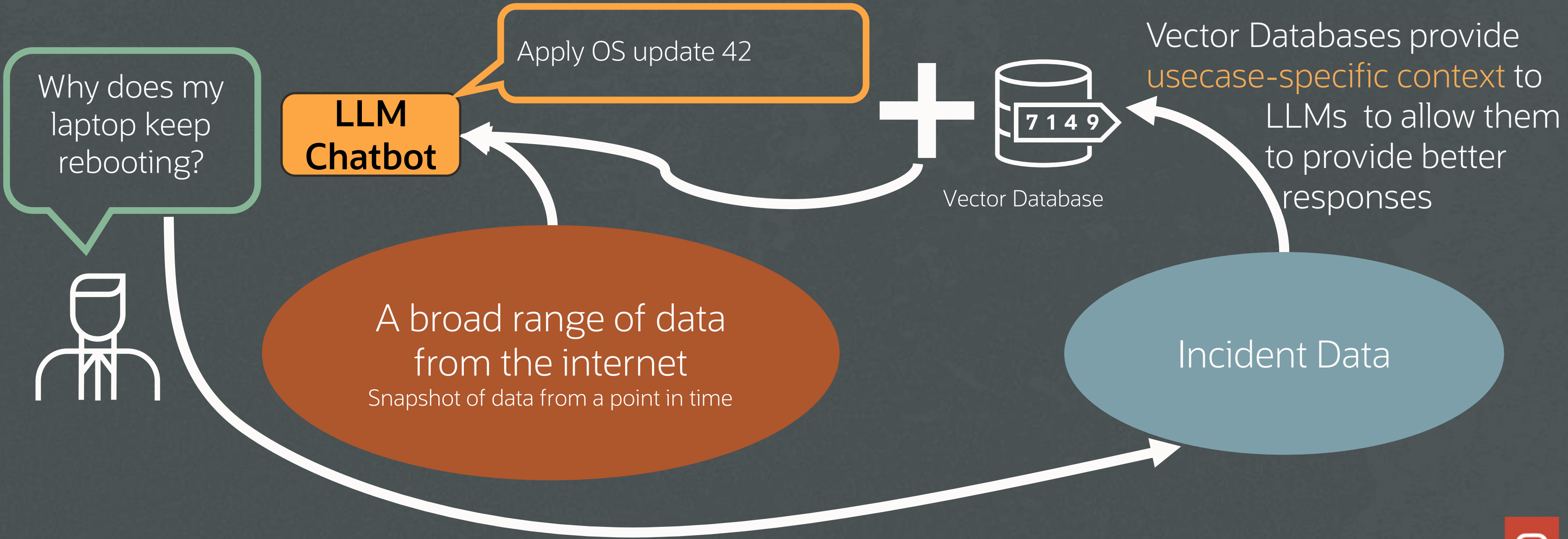

Why does my laptop keep rebooting?
LLM Chatbot
A broad range of data from the internet
Snapshot of data from a point in time
7 1 4 9
Vector Database
Incident Data
Match related incidents

Vector Databases provide usecase-specific context to LLMs to allow them to provide better responses

Role of Vector Databases in Generative AI

When augmented with enterprise information they provide better answers

Known as Retrieval Augmented Generation (RAG)



Why does my laptop keep rebooting?

LLM Chatbot

Apply OS update 42

Vector Database

Vector Databases provide usecase-specific context to LLMs to allow them to provide better responses

A broad range of data from the internet

Snapshot of data from a point in time

Incident Data

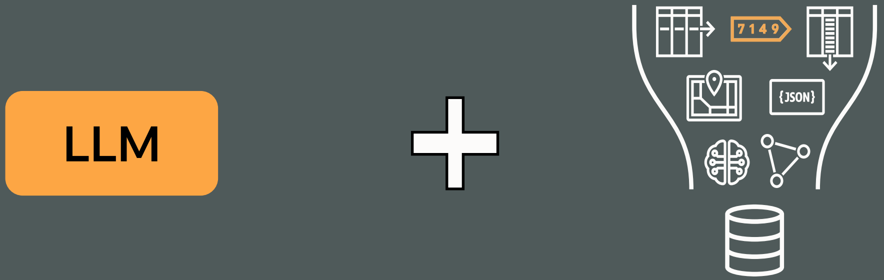
Role of Converged Oracle Database in Generative AI

Oracle is a Converged Database: Support for vectors in addition to Relational, JSON, Text, etc.

No need for data movement, avoids the cost, complexity, and security risk of multiple systems

Easily combine business data and vector data for ultra-sophisticated interactions with LLMs

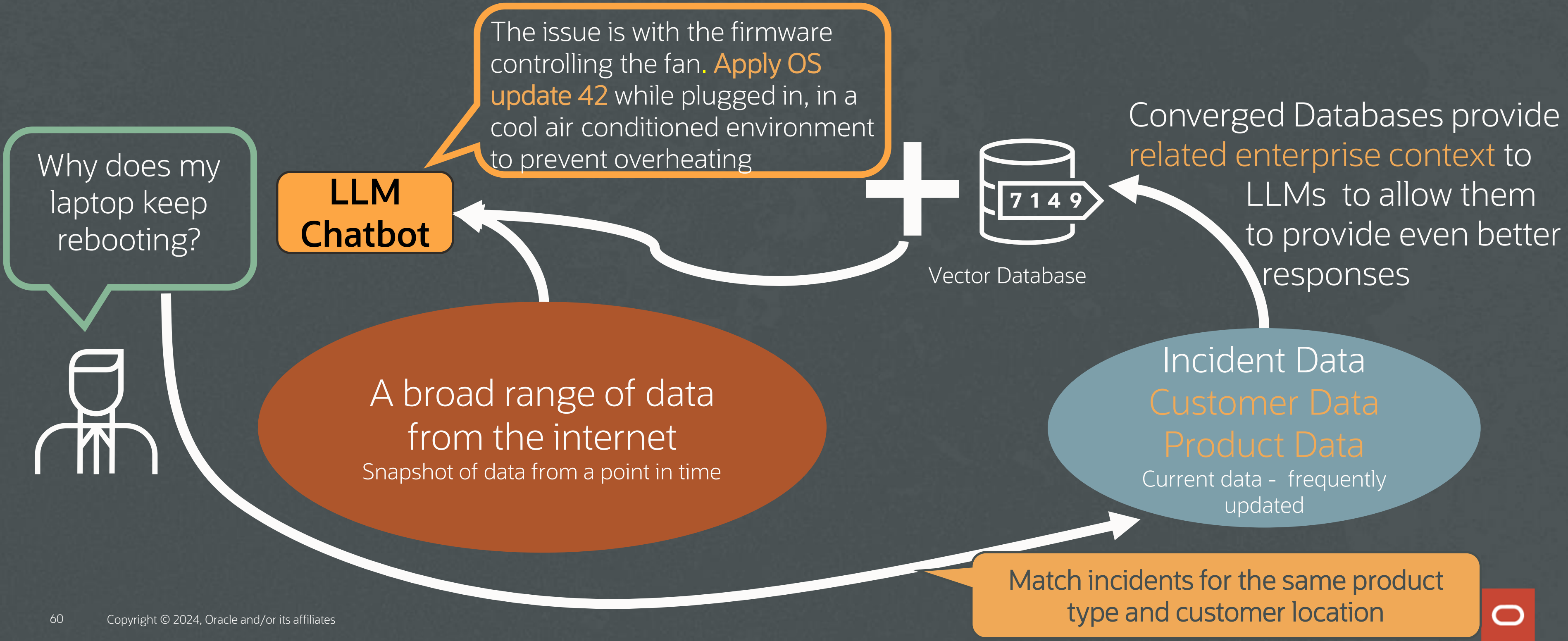
Ideal business outcomes



Converged Database
with Vector Support

Role of Converged Oracle Database in Generative AI

Converged Business Databases allow business rules, filters, security policies to be applied to RAG



Why does my laptop keep rebooting?

LLM Chatbot

The issue is with the firmware controlling the fan. Apply OS update 42 while plugged in, in a cool air conditioned environment to prevent overheating

Vector Database

Converged Databases provide related enterprise context to LLMs to allow them to provide even better responses

A broad range of data from the internet

Snapshot of data from a point in time

Incident Data

Customer Data

Product Data

Current data - frequently updated

Match incidents for the same product type and customer location

Why Oracle is the Ideal AI Vector Platform?

Oracle Powers Mission-Critical Extreme-Scale OLTP for the World

- Unlike classic AI search, AI Vector Search use cases are highly response-time critical
- AI Vector Search therefore is like Real-Time OLTP with many use cases, such as:

  Real-time detection of fraudulent financial transactions or cellular phone call

  Real-time match of an image with person(s) of interest

  Real-time match of a customer profile with prior customers for risk and creditworthiness
- Since mission-critical OLTP is Oracle's key differentiator, AI Vector Search plays directly into Oracle's greatest strength: 24 x 7 x 365, Secure, Petabyte Scale, Millions of Transactions / Sec

Many of Oracle's existing mission-critical OLTP use cases may be augmented with AI Vector Search in the future

Over 45 Years of Development
Powers AI Vector Search in Oracle

AI Vector Search is being added to Oracle Database 23ai without any major re-architecture, due to:

Advanced SQL Functionality: AI Vector Search is a simple extension

Advanced Data Engine: AI Vector Search is a simple extension

Industry-Leading Scalability: AI vector data seamlessly benefits from existing scalability mechanisms

Key Takeaways

Oracle AI Vector Search powers the Modern Enterprise

AI Vector Search is seamlessly integrated with Oracle Database 23ai database features for enterprise-grade performance and reliability

Converged SQL Processing

- Perform AI-vector powered similarity searches directly on your business data
- Simple to combine Relational, Document, Spatial, Graph, ML, and AI Vector Search in a single query – no need for multiple single-purpose databases

Advanced Data Engine

- Benefit from Approximate Indexing, In-Memory, Partitioning, Compression

Industry-Leading Scalability

- Transparent scale-out via RAC and Sharding
- Exadata Offload for efficient scale-out processing and capacity in storage tier

Efficient orchestration of Gen-AI pipelines

- Natively in the database or through integrated 3rd party frameworks

ORACLE

Thank You