ORACLE

AI Vector Search - RAG Application using PL/SQL in Oracle Database 23ai

Course ID: CSD020240733

Glenn Wong
Principle Solution Engineer
Oracle Hong Kong Limited

10th Jan 2025

Databases have traditionally performed value-based searches

Find products by name and price

There is a growing volume of unstructured business data that must be searched by semantics or meaning

Find products that match a photo or description

Example

Why the Traditional Query might not be good enough? – Oracle Text

Your Database Table Row only contains the following

ID	animal_name
1	Golden Retriever
2	Poodle
3	Wolf
4	Cat
5	Lion
6	Tiger

1) Simple Text Search

```
SELECT id, animal_name, SCORE(1) as relevance
FROM animals
WHERE CONTAINS(animal_name, 'cat', 1) > 0
ORDER BY SCORE(1) DESC;
```

2) Fuzzy Search

```
SELECT id, animal_name
FROM animals
WHERE CONTAINS(animal_name, 'FUZZY(dog)', 1) > 0;
```

3) Variation Search

```
SELECT id, animal_name
FROM animals
WHERE CONTAINS(animal_name, '$dog', 1) > 0;
```

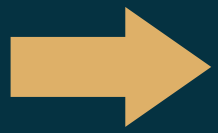
AI Vector Search is a breakthrough capability for searching data by semantics

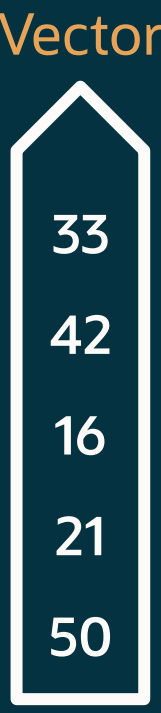
Enhances business applications by allowing them to search a combination of structured and unstructured data

AI Vector Search is also a critical component of the Gen-AI Ecosystem

What are AI Vectors?

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



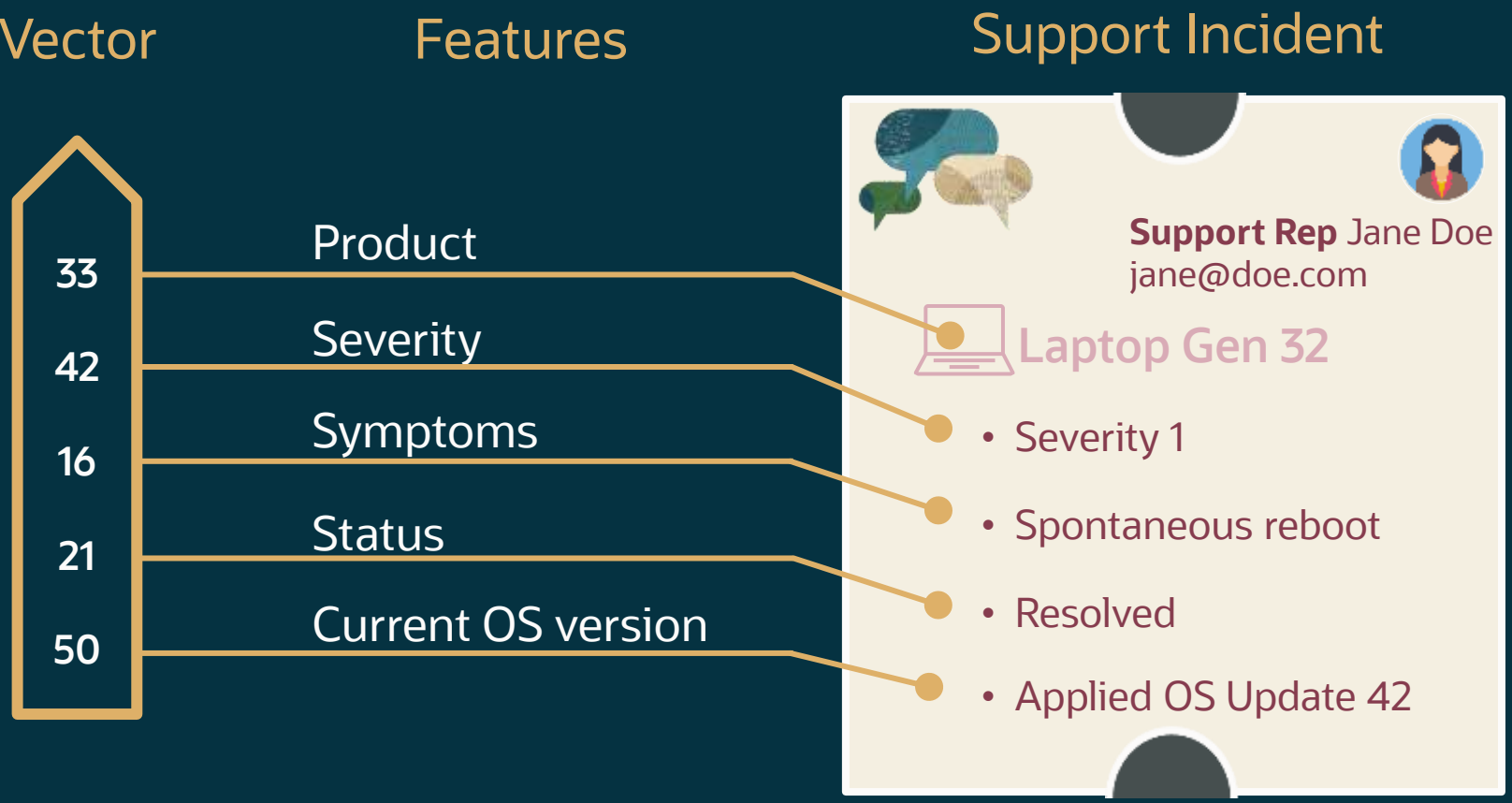


A vector is a sequence of numbers, that encode the important “features” of the data

Produced by AI Deep Learning Models

Represent the semantics of data, not the actual contents

Example: The Vector for a Support Incident could be ...



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

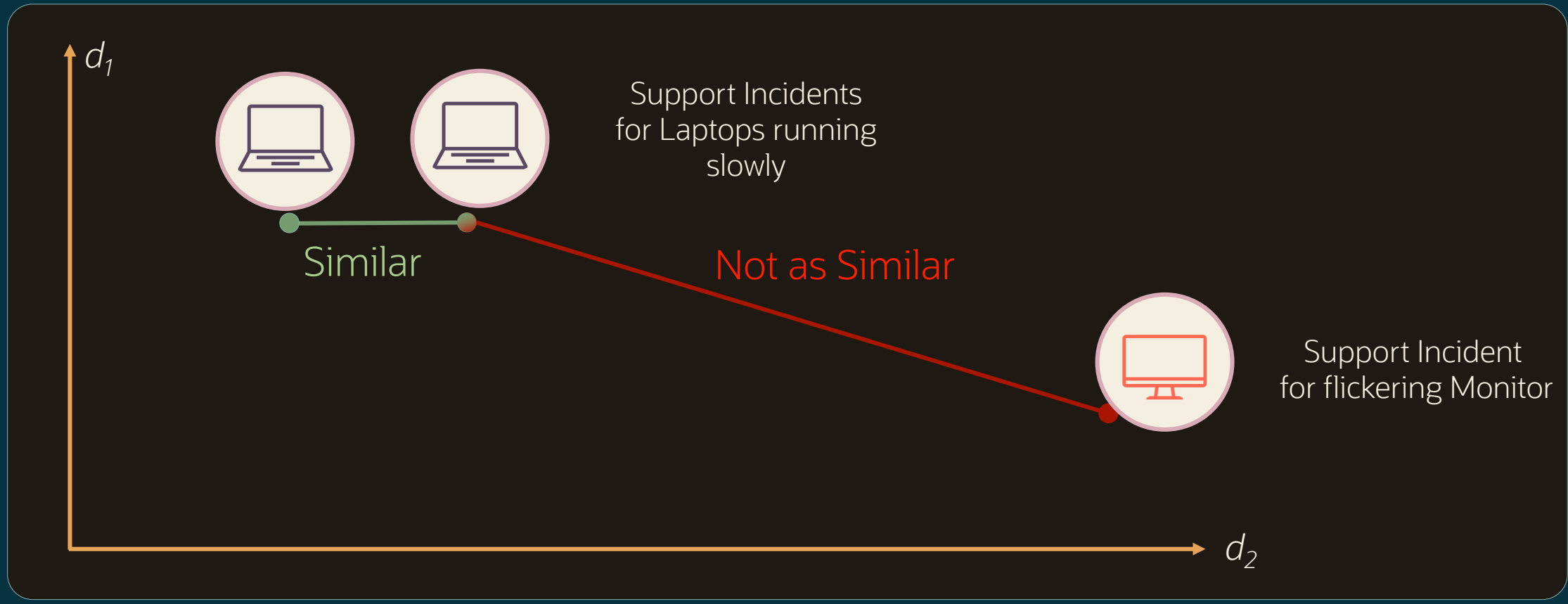
Note: Features determined by actual AI models are much more complex

The main operation on vectors is the Mathematical Distance between them

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

There are many mathematical distance formulas

Similarity Property: Support Incidents that are more similar produce vectors that are closer together



Vector Distance Metrics

Vector Distance Metrics	Real-Life Analogy	When to Use
Euclidean/Euclidean Squared Distance (L2)	Walking distance between two points in a park - the straight-line path you'd take	Best for measuring actual physical distances, like finding the nearest coffee shop to your current location
Cosine Similarity	Finding similar songs based on their mood/style, regardless of how long they are	Perfect for finding similar texts or documents, like finding similar movie reviews or product descriptions
Dot Product	Shopping recommendations based on both preference AND how strongly you feel about those preference	Good for recommendations systems, like “if you bought this, you might also like”
Manhattan Distance	Taxi driver distance in a city grid – can only drive along streets, no diagonal shortcuts	Useful when movement is restricted to fixed paths, like planning delivery routes in a city
Hamming Distance	Comparing DNA sequences to find mutations – counting how many positions are	Good for comparing binary pattern, like finding similar genetic codes or error detection
Jaccard Similarity	Comparing two recipe ingredients lists to see how similar they are	Best for Comparing sets of items, like finding similar shopping carts or matching product categories

Going back to our example

With 23ai, we can now ***embed*** the text into Vectors

ID	animal_name	Vector
1	Golden Retriever	[1, 22, 333, 4445]
2	Poodle	[1, 22, 333, 4448]
3	Wolf	[1, 22, 333, 4000]
4	Cat	[1, 22, 555, 6666]
5	Lion	[1, 22, 555, 7777]
6	Tiger	[1, 22, 555, 8888]

We can now Search with Dog, we will embed the search criteria's such as

[1,22,333,4440]

```
select *
from  animal
order by vector_distance ( vector_column,
:query_vector )
fetch first 3 rows only
```

it will return Golden Retriever, Poodle & Wolf

The Ideal datastore for AI Application with Vector's Operation

Key Requirements for Enterprise AI Vector Search

- Search on business data plus vector data
- Mission-critical fault tolerance
- Data security
- Performance (millisecond latency)
- Simplicity of solution
- Completeness of solution
- Sophisticated query support
- Massive scale

AI Application's need

Generate vector embeddings from unstructured data

Store vectors in table columns using new VECTOR type

Build approximate vector indexes on VECTOR columns

Perform AI Vector Search on VECTOR columns using SQL

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_VECTOR.load_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 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

Alternatively, you can simply specify the column as a VECTOR

```
CREATE TABLE Support_Incidents(
      id number,
      incident_text CLOB,
      img_vec 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 Indexes

Approximate Vector Indexes – Explained

Imagine you're trying to find the 5 most similar paintings to your favorite artwork in a massive museum with millions of paintings:

1) Exact Search (Traditional Way with SQL Index):
   - You would need to look at EVERY single painting
   - It's 100% accurate because you checked everything
   - But it takes forever (very slow)

2) Vector Indexes (Smart Modern Way with Vector Index):

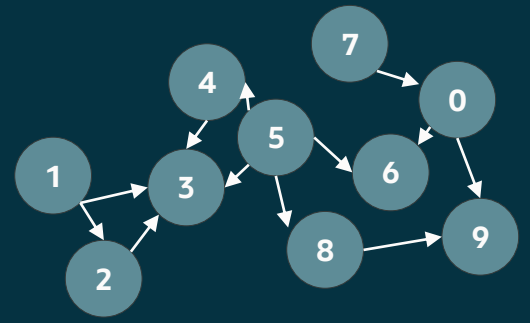
A. Neighbor Graph Index (HNSW - Hierarchical Navigable Small Worlds):

- Like having a smart museum guide who knows shortcuts
- Example: "If you like this painting, there's a similar one right next door"
- Very fast but keeps everything in memory (like the guide remembering everything)
- Great for smaller collections where speed is crucial

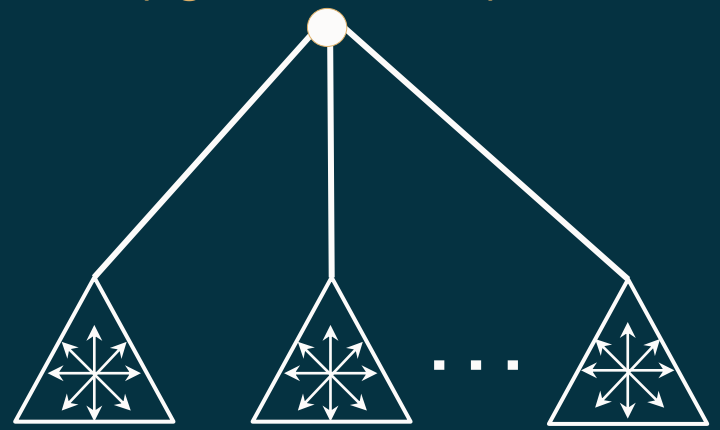
B. Neighbor Partition Index (IVF - Inverted File Index):

- Like organizing paintings into themed rooms
- Example: All landscape paintings in one room, portraits in another
- Good for huge collections
- Might miss some matches but handles massive amounts of art

Graph Vector Index
(e.g. **HNSW** Index)



Partition Vector Index
(e.g. **IVF_FLAT** index)



Real-World Analogy:

It's like asking friends for restaurant recommendations:

You don't ask EVERY person in the city (exact search)

but instead:

You ask a few knowledgeable friends who know your taste (approximate search)

They give you top 5 suggestions (Top-K results)

The suggestions might not be THE absolute best matches, but they're good enough and you get them much faster

How Vector Index helps in real cases

Think of Netflix recommendations:

- They don't compare your watching history with EVERY single movie they have
- Instead, they quickly look at groups of similar movies
- You get good recommendations in seconds
- Rather than perfect recommendations in hours

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

- 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

Querying Vectors

SQL 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>)`

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 – With Attribute Filters

Vector similarity search queries can easily be combined with relational filters and 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 before performing an index search on just those passing rows

INFILTER: For medium selectivity, apply the filter as the index is being searched

POST FILTER: For low selectivity, perform an index search first and then apply the filter on those top rows matched

Vector – 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 status:

```
SELECT id, name, photo
FROM   Customers c JOIN status s ON (c.status_id = s.id)
WHERE  c.city = 'San Francisco' AND s.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

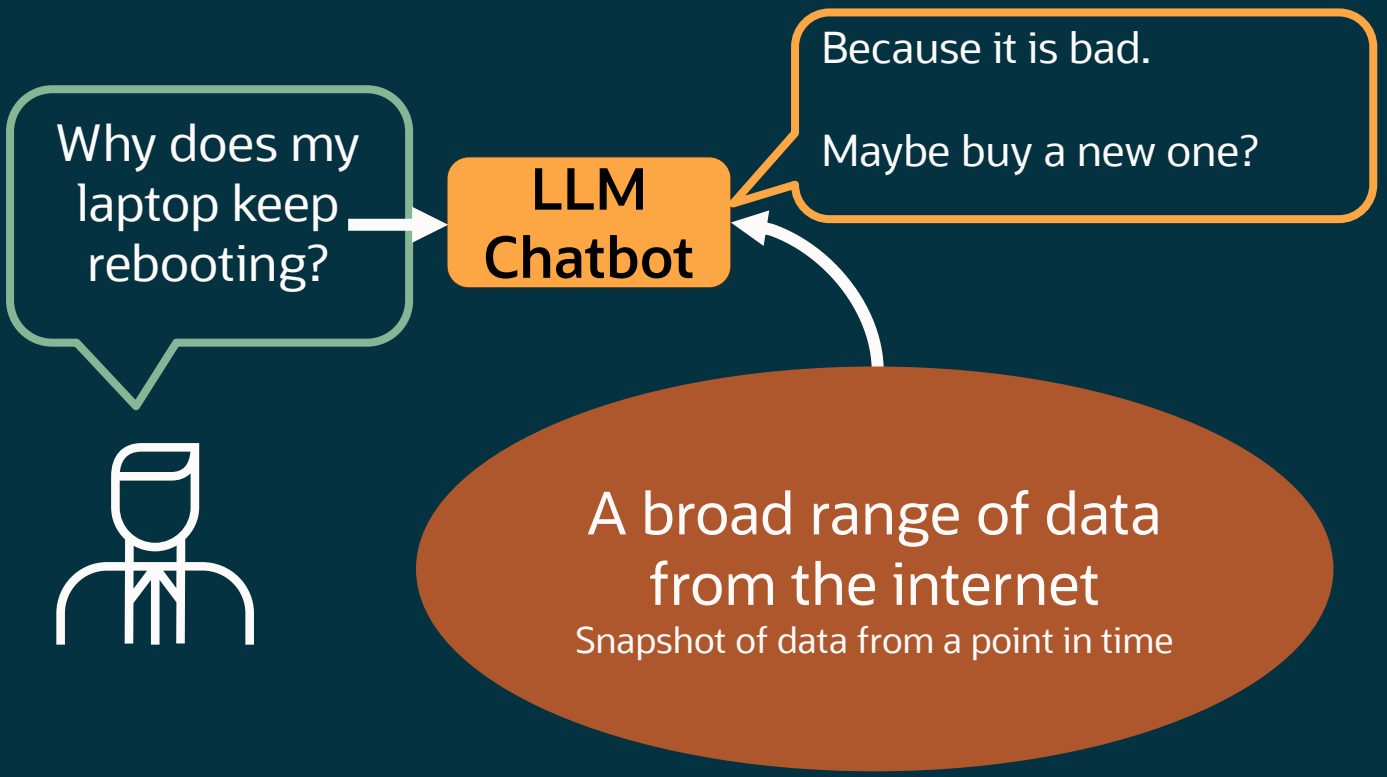
Trying the Vector Search with LiveSQL

https://livesql.oracle.com/ords/f?p=590:1000
https://blog.rishoradev.com/2024/05/23/oracle-database-23ai-vector-search-in-action/

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



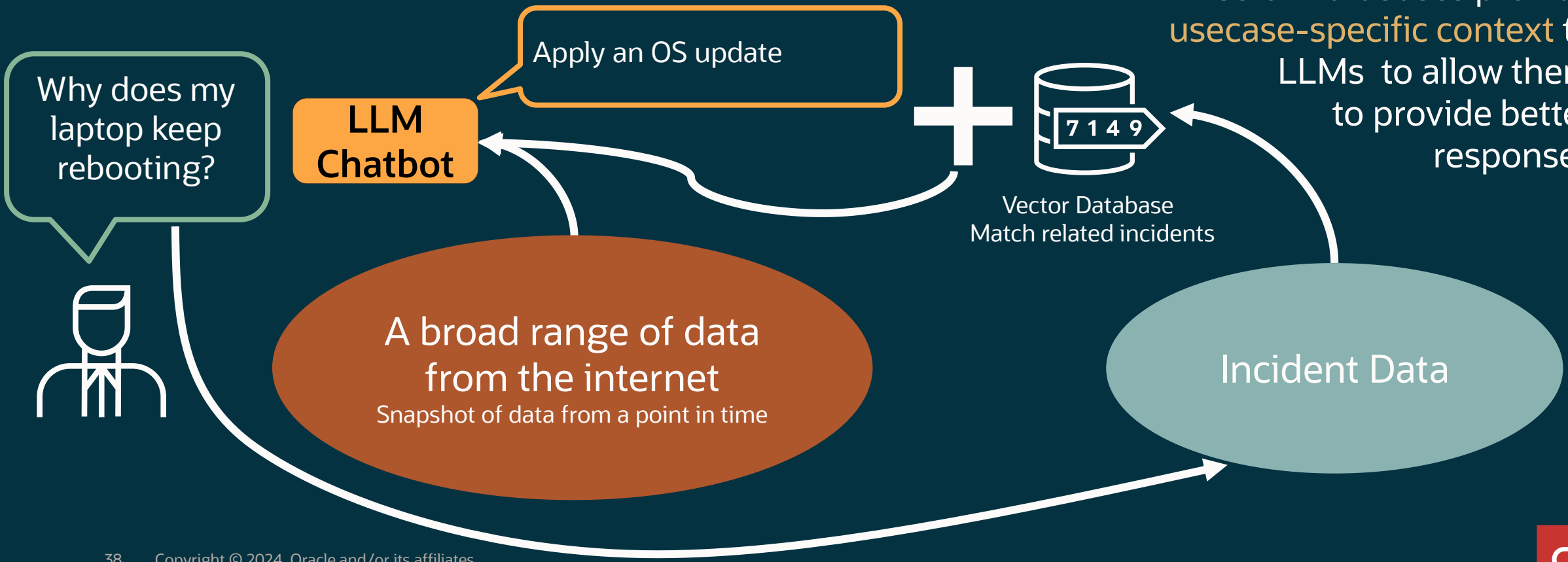
Role of Vector Databases in Generative AI

Provide enterprise content to enhance LLM interactions (retrieval augmentation)
Avoid having to train LLMs on sensitive enterprise data (not secure, expensive)

Better business outcomes

Role of Vector Databases in Generative AI

When augmented with enterprise information they provide better answers
Known as Retrieval Augmented Generation (RAG)

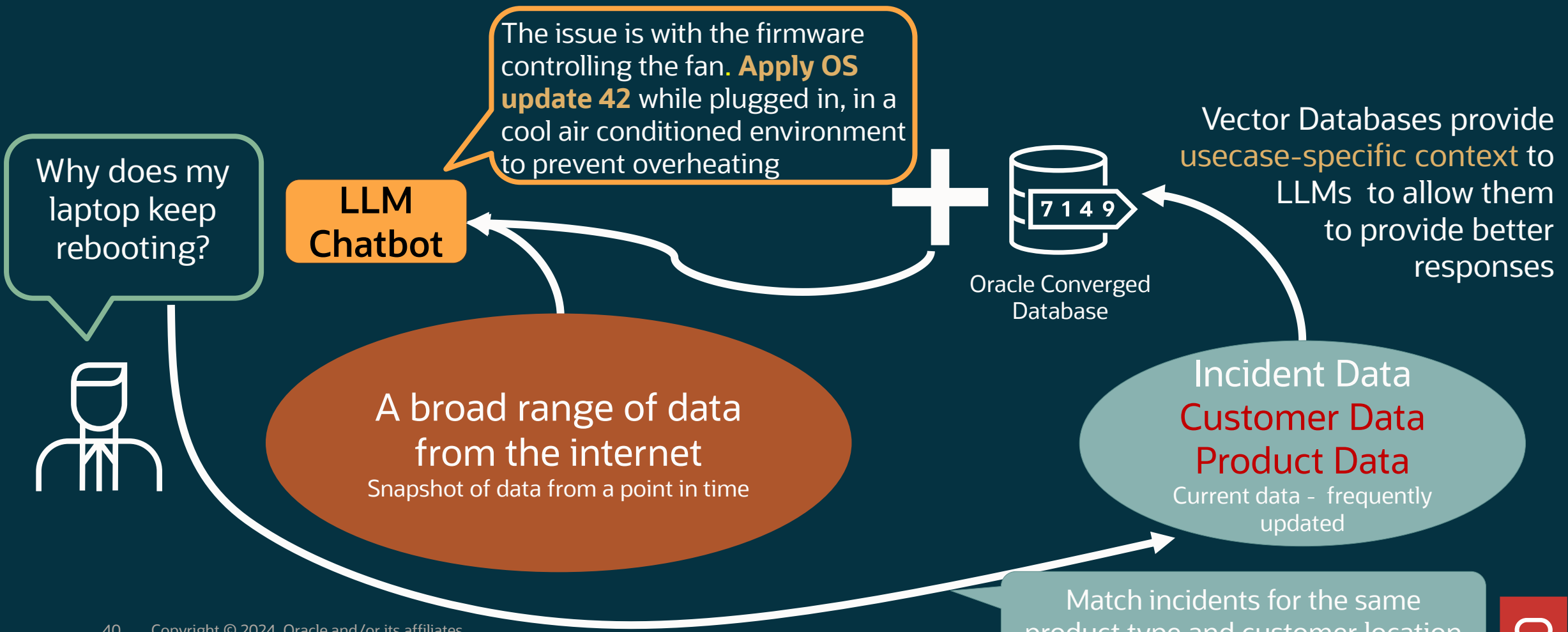


Role of Converged Oracle Database in Generative AI

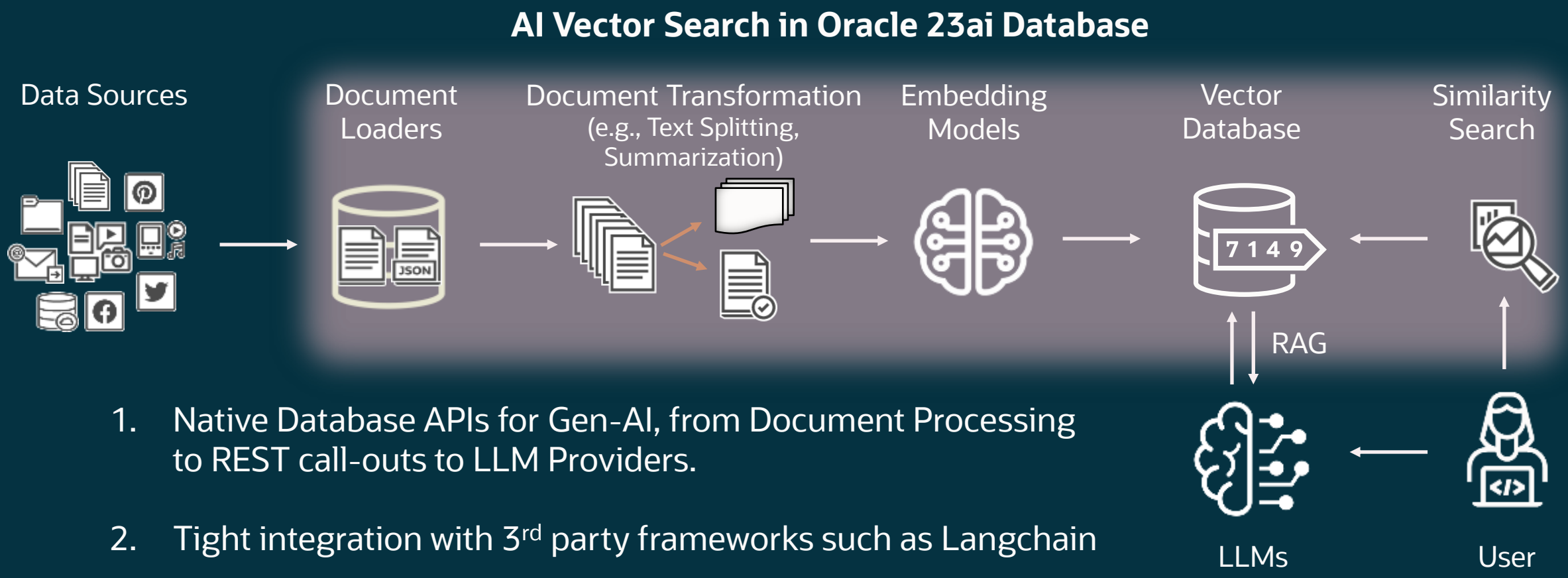
Oracle Converged Database has support for vectors in addition to Relational, JSON, Text, etc.
No need for data movement, avoids the cost, complexity, and security risk of multiple systems
Easily combine business data and vector data for ultra-sophisticated interactions with LLMs

Role of Converged Oracle Database in Generative AI

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



AI Vector Search powers Complete Gen AI pipeline



1. Native Database APIs for Gen-AI, from Document Processing to REST call-outs to LLM Providers.

2. Tight integration with 3rd party frameworks such as Langchain

Real Life Application with AI Vector Search

AI Vector Use Cases Encountered in the Limited Availability Program

Similarity Search

Find Similar
Support Tickets

Find Similar
Insurance Claims

Find Similar
Products

Detect manufacturing
anomalies

Find Promotions to offer
during final purchase

Text search using
semantic similarity

AI Vector Use Cases Encountered in the Limited Availability Program

Generative AI (RAG)

Document Classification

Support Chatbot

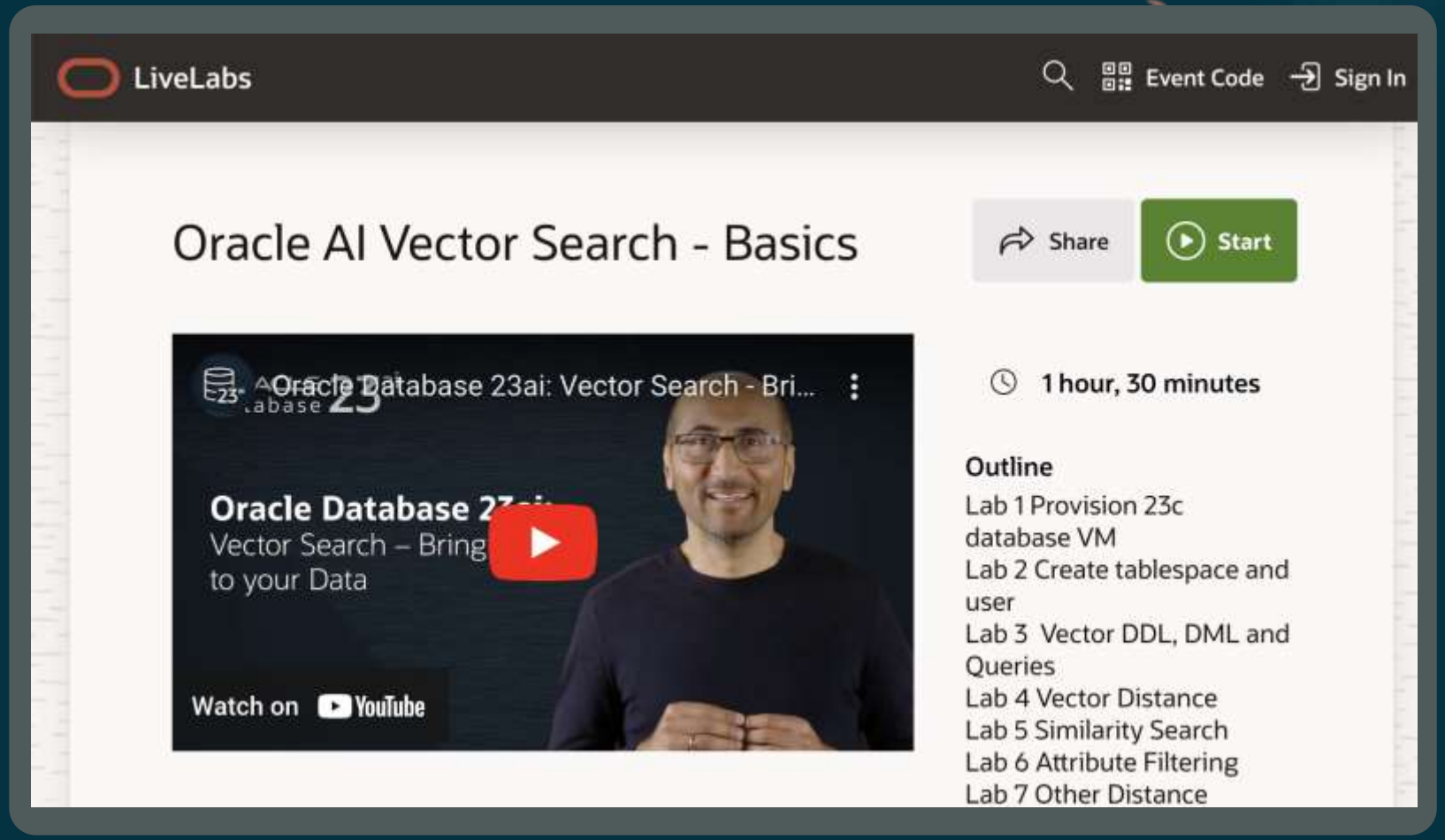
Identification of PII in documents

Online travel agent chatbot

Customer call classification

Natural language catalog search

Try
AI Vector Search
LiveLabs



https://livelabs.oracle.com/pls/apex/f?p=133:100:438449551367::::SEARCH:vector%20search

AI Vector Search Resources | Links

AI Vector Search LiveLabs :

https://livelabs.oracle.com/pls/apex/f?p=133:100:438449551367::::SEARCH:vector%20search

AI Vector Search User Guide :

https://docs.oracle.com/en/database/oracle/oracle-database/23/vecse/overview-ai-vector-search.html

AI Vector Search Blog :

https://blogs.oracle.com/database/post/oracle-announces-general-availability-of-ai-vector-search-in-oracle-database-23ai

Autonomous Database 23ai Container Image :

https://www.oracle.com/autonomous-database/free-trial/#free-container-image

Database 23ai Free :

https://www.oracle.com/database/free/get-started/

Oracle AI foundation associate certification

https://mylearn.oracle.com/ou/learning-path/become-an-oci-ai-foundations-associate-2024/140164