

# "Search and you shall find."

## Exploring the Oracle 26ai Search Landscape

Presented on 2<sup>nd</sup> October 2026

at

### **SOUG Day Zürich 2026**

in Zürich, Switzerland

by

Niall Mc Phillips - Long Acre sàrl

*niall.mcphillips@longacre.ch*

*@Niall\_McP*

*@niallmcp.bsky.social*

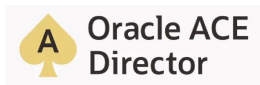


1

### About me: **Niall Mc Phillips**

Owner - Long Acre sàrl

Irish 🇮🇪 / 🇨🇭 Swiss living in Geneva, Switzerland.



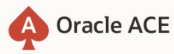
- Oracle Developer
- demi-DBA (was also a DBA from Oracle V6 to 11g)
- Developing web applications with PL/SQL and Oracle DB since 1995
- Developing with APEX since 2004 (HTML DB 1.5)

**Bluesky:** *@niallmcp.bsky.social*

Twitter/X: *@Niall\_McP*

e-mail: *niall.mcphillips@longacre.ch*

5



# The Oracle ACE Program

400+ technical experts helping peers globally



- The Oracle ACE Program recognizes and rewards community members for their technical and community contributions to the Oracle community
- 3 membership levels: Director, Pro, and Associate
- Nominate yourself or a colleague at [ace.oracle.com/nominate](https://ace.oracle.com/nominate)
- Learn more at [ace.oracle.com](https://ace.oracle.com)

✉ [aceprogram\\_ww@oracle.com](mailto:aceprogram_ww@oracle.com)

Facebook.com/OracleACEs

X @oracleace

in Oracle ACE Program Group



6

# SYMPOSIUM<sup>L2</sup>

Created by the community, to support the community

Sharing of reliable knowledge  
Supporting the various user groups and individuals



<https://sym42.org/>

7

like



9

## Searching with “like”



This is the basic “naïve” textual search that can work for very small datasets.

- it will not use an index if there is a wildcard at the start of the search string  
`where mytext like '%dog%'`
- it is case-sensitive
  - `where lower(mytext) like '%dog%'`



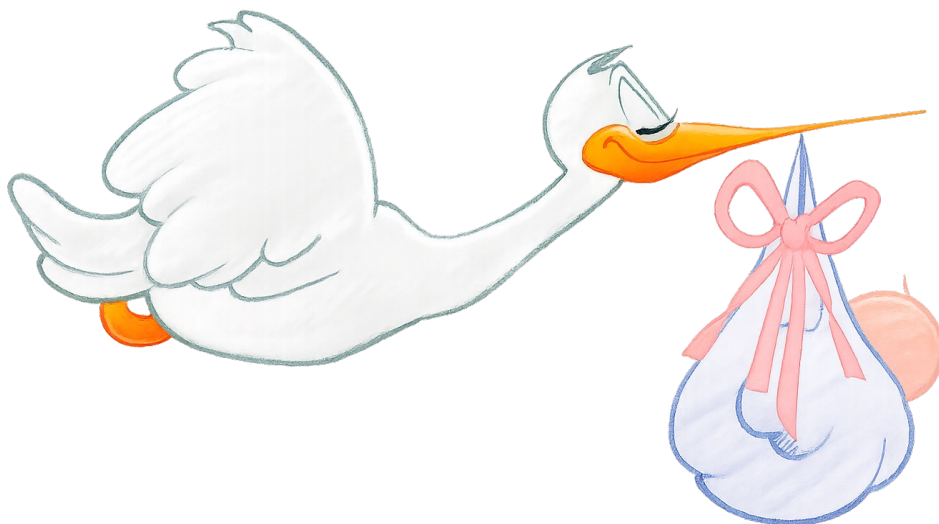
10

Oracle Text  
a.k.a. Search Indexes



11

**Where do Search Indexes come from?**



  
long acre  
solutions today

12

## Where do Search Indexes come from?

Oracle8 (1997) - Oracle ConText

Oracle8i (1999) - Oracle Intermedia Text.

Oracle9i (2001) - Oracle Text

Oracle23c (2023) - Search Indexes

13

## Search Indexes - Built on a solid foundation



  
long acre  
solutions today

14

## Using Search Indexes

- integral part of all Oracle database editions
- *really fast* and quite easy to start using
- just create an index and start searching
- index varchar2, XML, JSON, clobs and blobs (e.g., pdfs)
- uses the “contains” clause for querying
- allows AND/OR and more complex logic
- + many more advanced features...



15

## Creating a simple Search Index

```
create search index indexname  
on tablename (columnname);
```



16

## Creating a simple Search Index - examples

```
create search index si_judgments on judgments(description);
```

```
Index SI_JUDGMENTS created.
```

```
create search index si_hist_events on hist_events (description);
```

```
Index SI_HIST_EVENTS created.
```



17

## Searching with contains

```
select * from tablename
```

```
where
```

```
contains(searchcolumn, 'searchtext') > 0;
```



18

## Scoring search results

- The **score** of a search result gives an idea of the relevance of the result. High score indicates a higher relevance.
- Scores are always in the 1 to 100 range
- Scores have absolutely no meaning outside of their own query and cannot be compared between different queries, sub-queries or datasets.



19

## Scoring search results - syntax

```
select score (1), t.* from tablename t  
where  
contains(searchcolumn, 'searchtext', 1) > 0  
order by 1 desc;
```

Note that the (1) in score(1) matches the ,1) in the contains clause



20

## Oracle Text operator grammar and syntax



22

## Oracle Text operator grammar and syntax Principal operators

- ACCUM , at least one term
- AND &
- OR |
- NOT ~



23

## Some other operators

EQUIValence (=)

NEAR (;)

MINUS (-)

stem (\$)

Fuzzy

soundex (!)

and many more...

*full details in Oracle Text Reference at:*

*<https://docs.oracle.com/en/database/oracle/oracle-database/23/ccref/index.html#Oracle%C2%AE-Text>*



## Demo

- **Examples of searches with  
CONTAINS**



## Escaping terms entered

search for

- Africa and Near East
- “Near” is also an operator so we escape the search words using curly brackets {}

**{Africa}&{Near East}**



26

## Preparing text for search

- It can quickly become quite complex to parse and prepare the search text that users enter
- Normally some type of pre-processing is required for real-world scenarios



27

## Pre-processing user-input text for Google-like searches

Baseline principles:

- End-users should not need to know or understand Search Index grammar
- Everyone wants their searches to work “just like Google”



28

## Pre-processing user-input text for Google-like searches

- Roger Ford of Oracle has made a great PL/SQL package freely available on GitHub.

PARSER package:

<https://github.com/RogerFord/oracle-db-examples/tree/main/text/query-parser>



29

## The PARSER package

*We will use the  
parser.simpleSearch function to transform  
“Google-like” syntax into Oracle Text syntax.*

*e.g. “Ad Hoc Committee” becomes  
({Ad Hoc Committee})*



30

## PARSER examples

assessment damages becomes  
({assessment},{damages})

+assessment +damages becomes  
({assessment}&{damages})

+assessment -damages becomes  
({assessment}) ~{damages}



31

## DEMO – Preprocessing using parser



32

## Stoplists

Stoplists are lists containing words “stopwords” that should be ignored when searching.

i.e. frequently occurring words such as “the”, “also”, “their”, ...



33

## Indexing BLOB columns

Sample table JUDGMENT\_DOCUMENTS

- Blob column FILE\_CONTENT contains PDF files for each judgment

| COLUMN_NAME   | DATA_TYPE          |
|---------------|--------------------|
| JUDGMENT_NO   | NUMBER(32,4)       |
| FILENAME      | VARCHAR2(256 BYTE) |
| LANGUAGE_CODE | VARCHAR2(2 BYTE)   |
| FILE_CONTENT  | BLOB               |



## Indexing PDF files stored in BLOB columns

```
create search index txt_judgment_documents$1  
on judgment_documents(file_content);
```



## Searching the BLOB documents

BLOB Searches are the same as with any other column

```
select score(1) as the_score,  
       j.*  
  from judgments j  
 inner join judgment_documents jd  
        on (jd.judgment_no = j.judgment_no)  
 where contains(jd.file_content, :P8_SEARCHTEXT, 1) >  
        0  
 order by 1 desc;
```



36

## Oracle Text – 26ai Ubiquitous Search



40

## 26ai - Ubiquitous Search



41

## 26ai Ubiquitous Search

***With 26ai Ubiquitous Search we can search  
across multiple, unrelated data sources***

42

## 26ai Ubiquitous Search

***DBMS\_SEARCH***



43

## 26ai Ubiquitous Search – Creating an index

First, we'll create the index:

***dbms\_search.create\_index('UB1SEARCH');***



44

## 26ai Ubiquitous Search – adding data sources

Next, let's add two completely unrelated data sources:

```
dbms_search.add_source('UB1SEARCH',  
  'ANIMAL_FACTS');
```

```
dbms_search.add_source('UB1SEARCH', 'HIST_EVENTS');
```



45

## 26ai Ubiquitous Search – describe the table created

```
SQL> desc ub1search
```

| <b>Name</b>     | <b>Null?</b>    | <b>Type</b>            |
|-----------------|-----------------|------------------------|
| -----           | -----           | -----                  |
| <b>METADATA</b> | <b>NOT NULL</b> | <b>JSON</b>            |
| <b>DATA</b>     |                 | <b>JSON</b>            |
| <b>OWNER</b>    |                 | <b>VARCHAR2 (128)</b>  |
| <b>SOURCE</b>   |                 | <b>VARCHAR2 (128)</b>  |
| <b>KEY</b>      |                 | <b>VARCHAR2 (1024)</b> |



46

## 26ai Ubiquitous Search – explore the table created

*Let's take a look at the metadata*

```
select metadata from ublsearch
where contains(data, 'brain') > 0;
```



47

## 26ai Ubiquitous Search – table created

*Let's take a look at the metadata*

```
{ "OWNER": "NIALL", "SOURCE": "HIST_EVENTS", "KEY": { "HIST_EVENT_ID": 4169 } }
{ "OWNER": "NIALL", "SOURCE": "HIST_EVENTS", "KEY": { "HIST_EVENT_ID": 34276 } }
{ "OWNER": "NIALL", "SOURCE": "HIST_EVENTS", "KEY": { "HIST_EVENT_ID": 24285 } }
{ "OWNER": "NIALL", "SOURCE": "HIST_EVENTS", "KEY": { "HIST_EVENT_ID": 24487 } }
. . . .
{ "OWNER": "NIALL", "SOURCE": "ANIMAL_FACTS", "KEY": { "ANIMAL_FACT_ID": 4193 } }
{ "OWNER": "NIALL", "SOURCE": "ANIMAL_FACTS", "KEY": { "ANIMAL_FACT_ID": 4228 } }
{ "OWNER": "NIALL", "SOURCE": "ANIMAL_FACTS", "KEY": { "ANIMAL_FACT_ID": 4230 } }
```



48

## 26ai Ubiquitous Search – getting to the data

*We can extract the keys using JSON\_TABLE*

```
...  
  from ublsearch us  
  cross join json_table(us.metadata, '$.KEY[*]'  
    columns (animal_fact_id number path '$.ANIMAL_FACT_ID',  
             hist_event_id  number path '$.HIST_EVENT_ID')) j  
...
```



49

## 26ai Ubiquitous Search – getting to the data

*Joining with the underlying tables...*

```
...  
  from ublsearch us  
  cross join json_table(us.metadata, '$.KEY[*]'  
    columns (animal_fact_id number path '$.ANIMAL_FACT_ID',  
             hist_event_id  number path '$.HIST_EVENT_ID')) j  
  left outer join hist_events h on (h.id = j.hist_event_id)  
  left outer join animal_facts af on (af.animal_fact_id =  
    j.animal_fact_id)  
...
```



50

## 26ai Ubiquitous Search – displaying the data

*Join with the underlying tables...*

```
...
select case
    when j.animal_fact_id is not null then 'Animal Fact'
    else 'Historical Event'
end as result_type,
coalesce(j.animal_fact_id, j.hist_event_id) as theId,
coalesce(h.description, af.text) as theText
from ublsearch us
cross join ... (see in previous slide)
```



51

## 26ai Ubiquitous Search – displaying the data

*Creating a view for other developers*

```
create or replace view vw_ubsearch as
select case
    when j.animal_fact_id is not null then 'Animal Fact'
    else 'Historical Event'
end as result_type,
coalesce(j.animal_fact_id, j.hist_event_id) as theId,
coalesce(h.description, af.text) as theText,
us.data
from ublsearch us
cross join json_table(us.metadata, '$.KEY[*]'
    columns (animal_fact_id number path '$.ANIMAL_FACT_ID',
             hist_event_id number path '$.HIST_EVENT_ID')) j
left outer join hist_events h on (h.hist_event_id = j.hist_event_id)
left outer join animal_facts af on (af.animal_fact_id = j.animal_fact_id)
```



52

## 26ai Ubiquitous Search

*Demo*



53

## AI Vector Search



54

## What is AI Vector Search

**AI Vector Search** enables you to search structured and unstructured data based on its semantics or meaning, in addition to its values.



55

## Vectors

Vectors are derived from the ***semantic content*** of your data, rather than the underlying words or image parts.

For example, a search for the term ***architecture*** could also find results such as *buildings, construction* or even *a famous architect*.

In this respect they are quite different to other search options such as Oracle Search Indexes.



56

## Vectors

From a simplistic point of view, a vector is simply a list of numbers.

The number of dimensions that a vector has is how many numbers it contains.

*[0.8763, 1.8476, 0.2977, 2.8686, ..... ]*



57

## Vector Similarity

Vector similarity is calculated using the mathematical distance between two vectors

*[0.8763, 1.8476, 0.2977, 2.8686, ..... ]*

*[0.7836, 1.6748, 0.7729, 2.6868, ..... ]*



58

## Where do I get my vectors – Vector Embeddings

The creation / calculation of a vector from data is called ***EMBEDDING***

There are many different embedding models, including those provided by OpenAI, Cohere and many open-source models.

The model you choose will be influenced by the data that you are vectorising: text, images, video, sound, ....



59

## Where do I get my vectors – Vector Embeddings

For the purposes of today's presentation, we'll be using an OpenAI model called: *text-embedding-3-large*

This model returns a vector of dimension 3000



60

# Vector Embeddings – Using the OpenAI API

## Calling the OpenAI API from Postman - Authentication

HTTP OPENAI / Embedding

POST https://api.openai.com/v1/embeddings

Params Authorization Headers (12) Body Scripts Tests Settings

Auth Type

Bearer Token

Token

{{apikey}}

The authorization header will be automatically generated when you send the request. Learn more about [Bearer Token](#) authorization.



61

# Vector Embeddings – Using the OpenAI API

## Calling the OpenAI API from Postman specify text to be vectorised and which model to use

HTTP OPENAI / Embedding

Save Share

POST https://api.openai.com/v1/embeddings

Send

Params Authorization Headers (12) Body Scripts Tests Settings

Cookies

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL JSON

Beautify

```
1 {  
2   "input": "Geneva is situated at the end of the lake that shares its name 'Lake Geneva',  
3   "model": "text-embedding-3-large"  
4 }
```



62

## Vector Embeddings – Using the OpenAI API

### Calling the OpenAI API from Postman - Response

Body Cookies (2) Headers (25) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "object": "list",
3   "data": [
4     {
5       "object": "embedding",
6       "index": 0,
7       "embedding": [
8         -0.025723679,
9         -0.001198068,
10        -0.010066863,
11         0.01498899,
12        -0.035073247,
13        -0.04261721,
14         0.0186002,
15         0.0117920805,
16         0.0029263776,
17        -0.027826095.
```



63

## Let's look at the new data type: VECTOR

Since 23ai VECTOR is a new data type fully integrated into the database.

```
create table my_tab
(my_id integer,
my_text varchar2(500),
my_vector vector);
```



64

## VECTOR data type

| Possible Declaration Format                                                                                | Explanation                                                                                                                                                                        |
|------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>VECTOR(*, *)</code>                                                                                  | Vectors can have an arbitrary number of dimensions and formats. <code>VECTOR</code> and <code>VECTOR(*, *)</code> are equivalent.                                                  |
| <code>VECTOR(number_of_dimensions, *)</code><br>equivalent to<br><code>VECTOR(number_of_dimensions)</code> | Vectors must all have the specified number of dimensions or an error is thrown. Every vector will have its dimensions stored without format modification.                          |
| <code>VECTOR(*, dimension_element_format)</code>                                                           | Vectors can have an arbitrary number of dimensions, but their format will be up-converted or down-converted to the specified dimension element format (INT8, FLOAT32, or FLOAT64). |

A vector can be NULL but its dimensions cannot (for example, you cannot have a VECTOR with a NULL dimension such as [1.1, NULL, 2.2]).



65

## Embeddings – DBMS\_VECTOR

Example using the DBMS\_VECTOR package

- Create credentials for a 3rd party embedding API



66

## Embeddings – create credentials

```
declare
  jo json_object_t;
begin
  jo := json_object_t();
  jo.put('access_token', 'sk-*****'); -- api key
  dbms_vector.create_credential(
    credential_name => 'NIALLOPENAI_CRED',
    params          => json(jo.to_string));
end;
/
```



67

## Embeddings – create credentials

```
SQL>
SQL> set echo on
SQL> declare
  2   jo json_object_t;
  3   begin
  4     jo := json_object_t();
  5     jo.put('access_token', embeddings.getOpenaiAPIKey());
  6     dbms_vector.create_credential(
  7       credential_name => 'NIALLOPENAI_CRED',
  8       params          => json(jo.to_string));
  9   end;
 10  /
```

retrieve the API  
key from table

PL/SQL procedure successfully completed.



68

## Embeddings – *utl\_to\_embedding* function

- Create credentials for a 3rd party embedding API
- Use the *dbms\_vector.utl\_to\_embedding* function to embed and return a vector

```
DBMS_VECTOR.UTL_TO_EMBEDDING  
  (DATA IN CLOB,  
   PARAMS IN JSON default NULL )  
  return VECTOR
```



69

## Embeddings – *utl\_to\_embedding* function example parameters

```
{  
  "provider": "openai",  
  "credential_name": "NIALL_OPENAI_CRED",  
  "url":  
    "https://api.openai.com/v1/embeddings",  
  "model": "text-embedding-3-large"  
}
```



70

## Embeddings – *utl\_to\_embedding* function usage example

```
...
l_params := '{"provider": "openai",
             "credential_name": "NIALL_OPENAI_CRED",
             "url": "https://api.openai.com/v1/embeddings",
             "model": "text-embedding-3-large"}';

...
l_vector := dbms_vector.utl_to_embedding
             (data    => 'Grüezi mitenand!',
              params => json(l_params) );

...
```



71

## Embeddings – *utl\_to\_embedding* function usage example

```
SQL> declare
2   v_input  clob;
3   v_vector vector;
4   begin
5     v_input := 'Grüezi mitenand!';
6     v_vector := dbms_vector.utl_to_embedding
7                 (data    => v_input,
                  params => json(embeddings.getOpenaiAPIEmbeddingParams()));
8
9     dbms_output.put_line(v_input);
10    dbms_output.put_line(embeddings.serializeMyVector(v_vector));
11  end;
12  /
Grüezi mitenand!
[-3.03840637E-003,3.75061035E-002,-1.86462402E-002,3.78723145E-002,2.1774292E-002,-5

PL/SQL procedure successfully completed.
```



72

## Embeddings – table of sentences

We will use a table called SENTENCES which contains 150 ChatGPT-generated sentences. 100 about cities in Europe and 50 about animals.

```
SQL> desc sentences
Name      Null?     Type
-----
ID         NUMBER
THE_TEXT   NOT NULL VARCHAR2(255)
THE_VECTOR VECTOR(*,*,*)
```

| ID | THE_TEXT                                                                                        | THE_VECTOR |
|----|-------------------------------------------------------------------------------------------------|------------|
| 1  | Ljubljana, Slovenia is admired for its charming city center and lively atmosphere.              | (null)     |
| 2  | Split, Croatia is famous for the ancient Diocletian's Palace and its Adriatic coast.            | (null)     |
| 3  | Ljubljana, Slovenia is recognized for its friendly people and green spaces.                     | (null)     |
| 4  | Sofia, Bulgaria is celebrated for its Orthodox churches and vibrant city life.                  | (null)     |
| 5  | Bucharest, Romania is known for its mix of historical and modern architecture.                  | (null)     |
| 6  | Belgrade, Serbia is admired for its fortresses and riverside setting.                           | (null)     |
| 7  | Sarajevo, Bosnia and Herzegovina is famous for its diverse culture and historical significance. | (null)     |
| 8  | Tirana, Albania is recognized for its colorful architecture and dynamic cultural scene.         | (null)     |
| 9  | Skopje, North Macedonia is noted for its eclectic blend of old and new buildings.               | (null)     |
| 10 | Pristina, Kosovo is admired for its youthful energy and growing arts scene.                     | (null)     |



73

## Embeddings - updating the embedding vector

update all rows in the sentences table to populate their vectors

```
update sentences s
```

```
  set s.the_vector =
```

```
    dbms_vector.utl_to_embedding
```

```
      (s.the_text,
```

```
       json(l_params)) ;
```



74

## We can now use the embedding

View our sentences table with its vectors populated

```
sql> select id, the_text, the_vector from sentences fetch first 10 rows only;
```

| ID | THE_TEXT                                                                                        | THE_VECTOR                                                                                                                                                                                                                                                                                                                                                        |
|----|-------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1  | Ljubljana, Slovenia is admired for its charming city center and lively atmosphere.              | [3.92498486E-002,3.8745027E-002,-1.384311961E-002,-4.54256348E-002,-3.0577866E-004,4.74944785E-002,-1.3237983E-002,-7.41957454E-003,-1.714117195E-003,-1.28046563E-003,-5.60169034E-002,7.53022218E-003,-2.580895729E-002,-1.25712929E-002,-2.69968752E-002,-2.40882835E-003,-3.20351236E-002,1.77446958E-002,-2.347207781E-002,1.52237155E-002,-2.10000000E-002] |
| 2  | Split, Croatia is famous for the ancient Diocletian's Palace and its Adriatic coast.            | [3.84311961E-002,-4.54256348E-002,-3.0577866E-004,4.74944785E-002,-1.3237983E-002,-7.41957454E-003,-1.714117195E-003,-1.28046563E-003,-5.60169034E-002,7.53022218E-003,-2.580895729E-002,-1.25712929E-002,-2.69968752E-002,-2.40882835E-003,-3.20351236E-002,1.77446958E-002,-2.347207781E-002,1.52237155E-002,-2.10000000E-002]                                  |
| 3  | Ljubljana, Slovenia is recognized for its friendly people and green spaces.                     | [3.92498486E-002,3.8745027E-002,-1.384311961E-002,-4.54256348E-002,-3.0577866E-004,4.74944785E-002,-1.3237983E-002,-7.41957454E-003,-1.714117195E-003,-1.28046563E-003,-5.60169034E-002,7.53022218E-003,-2.580895729E-002,-1.25712929E-002,-2.69968752E-002,-2.40882835E-003,-3.20351236E-002,1.77446958E-002,-2.347207781E-002,1.52237155E-002,-2.10000000E-002] |
| 4  | Sofia, Bulgaria is celebrated for its Orthodox churches and vibrant city life.                  | [3.92498486E-002,3.8745027E-002,-1.384311961E-002,-4.54256348E-002,-3.0577866E-004,4.74944785E-002,-1.3237983E-002,-7.41957454E-003,-1.714117195E-003,-1.28046563E-003,-5.60169034E-002,7.53022218E-003,-2.580895729E-002,-1.25712929E-002,-2.69968752E-002,-2.40882835E-003,-3.20351236E-002,1.77446958E-002,-2.347207781E-002,1.52237155E-002,-2.10000000E-002] |
| 5  | Bucharest, Romania is known for its mix of historical and modern architecture.                  | [3.92498486E-002,3.8745027E-002,-1.384311961E-002,-4.54256348E-002,-3.0577866E-004,4.74944785E-002,-1.3237983E-002,-7.41957454E-003,-1.714117195E-003,-1.28046563E-003,-5.60169034E-002,7.53022218E-003,-2.580895729E-002,-1.25712929E-002,-2.69968752E-002,-2.40882835E-003,-3.20351236E-002,1.77446958E-002,-2.347207781E-002,1.52237155E-002,-2.10000000E-002] |
| 6  | Belgrade, Serbia is admired for its fortresses and riverside setting.                           | [3.92498486E-002,3.8745027E-002,-1.384311961E-002,-4.54256348E-002,-3.0577866E-004,4.74944785E-002,-1.3237983E-002,-7.41957454E-003,-1.714117195E-003,-1.28046563E-003,-5.60169034E-002,7.53022218E-003,-2.580895729E-002,-1.25712929E-002,-2.69968752E-002,-2.40882835E-003,-3.20351236E-002,1.77446958E-002,-2.347207781E-002,1.52237155E-002,-2.10000000E-002] |
| 7  | Sarajevo, Bosnia and Herzegovina is famous for its diverse culture and historical significance. | [3.92498486E-002,3.8745027E-002,-1.384311961E-002,-4.54256348E-002,-3.0577866E-004,4.74944785E-002,-1.3237983E-002,-7.41957454E-003,-1.714117195E-003,-1.28046563E-003,-5.60169034E-002,7.53022218E-003,-2.580895729E-002,-1.25712929E-002,-2.69968752E-002,-2.40882835E-003,-3.20351236E-002,1.77446958E-002,-2.347207781E-002,1.52237155E-002,-2.10000000E-002] |
| 8  | Tirana, Albania is recognized for its colorful architecture and dynamic cultural scene.         | [3.92498486E-002,3.8745027E-002,-1.384311961E-002,-4.54256348E-002,-3.0577866E-004,4.74944785E-002,-1.3237983E-002,-7.41957454E-003,-1.714117195E-003,-1.28046563E-003,-5.60169034E-002,7.53022218E-003,-2.580895729E-002,-1.25712929E-002,-2.69968752E-002,-2.40882835E-003,-3.20351236E-002,1.77446958E-002,-2.347207781E-002,1.52237155E-002,-2.10000000E-002] |
| 9  | Skopje, North Macedonia is noted for its eclectic blend of old and new buildings.               | [3.92498486E-002,3.8745027E-002,-1.384311961E-002,-4.54256348E-002,-3.0577866E-004,4.74944785E-002,-1.3237983E-002,-7.41957454E-003,-1.714117195E-003,-1.28046563E-003,-5.60169034E-002,7.53022218E-003,-2.580895729E-002,-1.25712929E-002,-2.69968752E-002,-2.40882835E-003,-3.20351236E-002,1.77446958E-002,-2.347207781E-002,1.52237155E-002,-2.10000000E-002] |
| 10 | Pristina, Kosovo is admired for its youthful energy and growing arts scene.                     | [3.92498486E-002,3.8745027E-002,-1.384311961E-002,-4.54256348E-002,-3.0577866E-004,4.74944785E-002,-1.3237983E-002,-7.41957454E-003,-1.714117195E-003,-1.28046563E-003,-5.60169034E-002,7.53022218E-003,-2.580895729E-002,-1.25712929E-002,-2.69968752E-002,-2.40882835E-003,-3.20351236E-002,1.77446958E-002,-2.347207781E-002,1.52237155E-002,-2.10000000E-002] |

10 rows selected.



## New Vector function – VECTOR\_DISTANCE

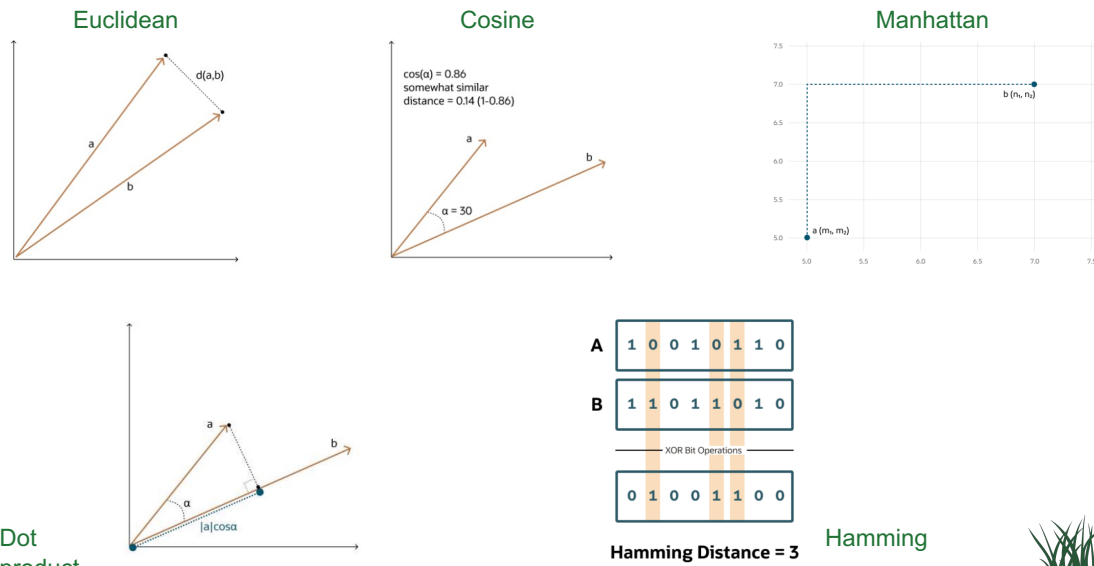
VECTOR\_DISTANCE calculates the distance between two vectors

### **vector\_distance (vector1, vector2, method)**

returns a number containing the distance between vectors 1 and 2, calculated according the the method specified in the 3<sup>rd</sup> parameter.



## Vector distance measurement methods



77

## VECTOR\_DISTANCE – Example

```
select vector_distance
      (s.the_vector, P_SEARCH_VECTOR, COSINE) as vdist,
      s.the_text,
      s.id
  from sentences s
 order by vdist asc
 fetch first 8 rows only;
```



78

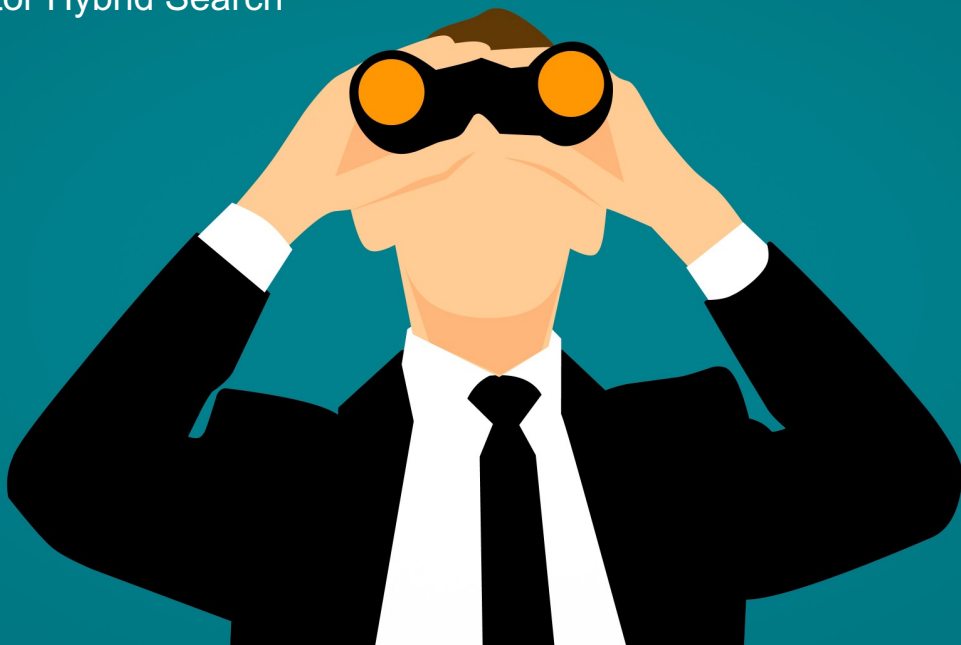
## AI Vector Search - DEMOS

Quick demo using SQL / APEX...



79

## AI Vector Hybrid Search



80

***Wouldn't it be nice if we could mix both Oracle Text Search and AI Vector Search together?***



81

## **AI Vector Hybrid Search**

**With AI Vector Hybrid Search we can retrieve both Oracle Text and AI Vector Search results with a single query**



82

## AI Vector Search

### Importing pre-trained models

Import pre-trained embedding ML models for vector generation within the DB

- Open Neural Network Exchange (ONNX) is an open format built to represent machine learning models
- <https://onnx.ai/>



83

## AI Vector Hybrid Search

- To use AI Vector Hybrid search, we first need to import an embedding model into the database
- ONNX models can be found on Hugging Face and other platforms
- To import a model use:  
`dbms_vector.load_onnx_model_cloud`
- The model used here: `all_MiniLM_L12_v2.onnx`



84

## AI Vector Hybrid Search

- Viewing models loaded in the DB

```
select *  
  from user_mining_models  
 where model_name='ALL_MINILM_L12_V2';
```

| MODEL_NAME        | MINING_FUNCTION | ALGORITHM | ALGORITHM_TYPE | CREATION_DATE | BUILD_DURATION | MODEL_SIZE |
|-------------------|-----------------|-----------|----------------|---------------|----------------|------------|
| ALL_MINILM_L12_V2 | EMBEDDING       | ONNX      | NATIVE         | 20-MAY-25     | 0              | 133322334  |



85

## AI Vector Hybrid Search

Getting embedding vectors using a model loaded in the DB

```
select  
  vector_embedding  
    (all_minilm_l12_v2  
      using 'Grüezi mitenand!' as data);
```

```
SQL> select vector_embedding (all_minilm_l12_v2  
                             2                               using 'Grüezi mitenand!' as data);
```

```
VECTOR_EMBEDDING(ALL_MINILM_L12_V2USING'GRÜEZIMITENAND!'ASDATA)
```

```
-----  
[-5.49018532E-002,8.5881941E-002,-3.00570447E-002,3.86604518E-002,1  
2,1.11143216E-002,-4.85252868E-003,-9.49634239E-003,6.25020266E-002
```

86

## AI Vector Hybrid Search

- Creating a hybrid vector index

```
create hybrid vector index
    sentences_hybrid_idx
on sentences(the_text)
parameters ('model all_minilm_l12_v2')
parallel 8;
```



87

## AI Vector Hybrid Search - query

- Search using the newly created index

```
select json_serialize(
    dbms_hybrid_vector.search(
        json(
            '{
                "hybrid_index_name" : "sentences_hybrid_idx",
                "search_text"       : "river"
            }'
        )
    )
returning clob pretty) as output;
```



88

## AI Vector Hybrid Search – query result

```
[
  {
    "rowid" : "AAArS1AAAAAGAl+ABJ",
    "score" : 56.97,
    "vector_score" : 61.67,
    "text_score" : 10,
    "vector_rank" : 1,
    "text_rank" : 1,
    "chunk_text" : "Budapest, Hungary is notable for its beautiful Danube River
and thermal baths.",
    "chunk_id" : "1"
  },
  {
    "rowid" : "AAArS1AAAAAGAl/AA0",
    "score" : 53.58,
    "vector_score" : 58.94,
    "text_score" : 0,
    "vector_rank" : 2,
    "text_rank" : 80,
    "chunk_text" : "Gothenburg, Sweden is recognized for its Dutch-style canals
and cultural events.",
    "chunk_id" : "1"
  },
  {
    "rowid" : "AAArS1AAAAAGAl/AA5",
    "score" : 53.16
```



89

## AI Vector Hybrid Search – query with json\_table

```
select *
from json_table(dbms_hybrid_vector.search(
    json('{"hybrid_index_name" : "sentences_hybrid_idx",
    "search_text" : "wool" }'),
    )
, '$[*]'
columns (score          varchar2(20) path '$.score',
         vector_score    varchar2(20) path '$.vector_score',
         text_score      varchar2(20) path '$.text_score',
         vector_rank     varchar2(20) path '$.vector_rank',
         text_rank       varchar2(20) path '$.text_rank',
         chunk_text      varchar2(200) path '$.chunk_text',
         chunk_id        varchar2(20) path '$.chunk_id') );
```

90

## AI Vector Hybrid Search – query with json\_table

| SCORE | VECTOR_SCORE | TEXT_SCORE | VECTOR_RANK | TEXT_RANK | CHUNK_TEXT                                                     |
|-------|--------------|------------|-------------|-----------|----------------------------------------------------------------|
| 66.05 | 71.85        | 8          | 1           | 1         | Sheep are valued for their wool, meat, and milk.               |
| 63.57 | 69.13        | 8          | 2           | 1         | Alpacas are kept for their soft, luxurious wool.               |
| 61.95 | 67.35        | 8          | 3           | 1         | Llamas are used as pack animals and for their wool.            |
| 53.51 | 58.86        | 0          | 4           | 80        | Sheepdogs are specifically bred and trained to help herd sheep |
| 52.1  | 57.31        | 0          | 5           | 80        | Cows are essential farm animals, providing milk, meat, and tea |
| 51.61 | 56.77        | 0          | 6           | 80        | Goats are known for their versatility, providing milk, meat, a |
| 51.49 | 56.64        | 0          | 7           | 80        | Rabbits are sometimes kept on farms for their meat and fur.    |
| 51.09 | 56.2         | 0          | 8           | 80        | Emus are large birds farmed for their meat, oil, and leather.  |
| 51.01 | 56.11        | 0          | 9           | 80        | Hedgehogs are admired for their unique appearance and quiet na |
| 50.65 | 55.71        | 0          | 10          | 80        | Rust is a municipality in the district of Ortenau in Baden-Wür |
| 50.28 | 55.31        | 0          | 11          | 80        | Nuuk, Greenland is recognized for its stunning arctic landscap |
| 50.26 | 55.29        | 0          | 12          | 80        | Dogs are often used on farms to herd livestock and provide pro |
| 50.09 | 55.1         | 0          | 13          | 80        | Tarantulas are kept by enthusiasts for their unique appearance |
| 49.96 | 54.96        | 0          | 14          | 80        | Peacocks are kept on farms and estates for their ornamental be |
| 49.93 | 54.92        | 0          | 15          | 80        | Cats are known for their ability to control rodent populations |
| 49.85 | 54.83        | 0          | 16          | 80        | Copenhagen, Denmark is famous for its colorful Nyhavn harbor a |
| 49.75 | 54.72        | 0          | 17          | 80        | Horses are essential in many agricultural tasks, from plowing  |
| 49.6  | 54.56        | 0          | 18          | 80        | Malmo, Sweden is known for its modern architecture and green s |
| 49.44 | 54.38        | 0          | 19          | 80        | Alpacas are also kept as pets and for their calm and friendly  |
| 49.25 | 54.17        | 0          | 20          | 80        | Horses are prized for their strength, speed, and companionship |

91

## AI Vector Hybrid Search – query with json\_table

Demo

92



93

## What we've seen...

Oracle Text / Search Indexes

Ubiquitous Search

AI Vector Search

AI Hybrid Vector Search

94

