How to query your data using natural language Introduction to AI features in Oracle 23ai

Andrzej Nowicki

Voxxed Days, CERN, 2025

Andrzej Nowicki 🇵🇱

12 years of Oracle DB experience
Database Engineer @ CERN since 2020

andrzejnowicki

andrzej.nowicki@cern.ch

www.andrzejnowicki.pl

IT @ CERN

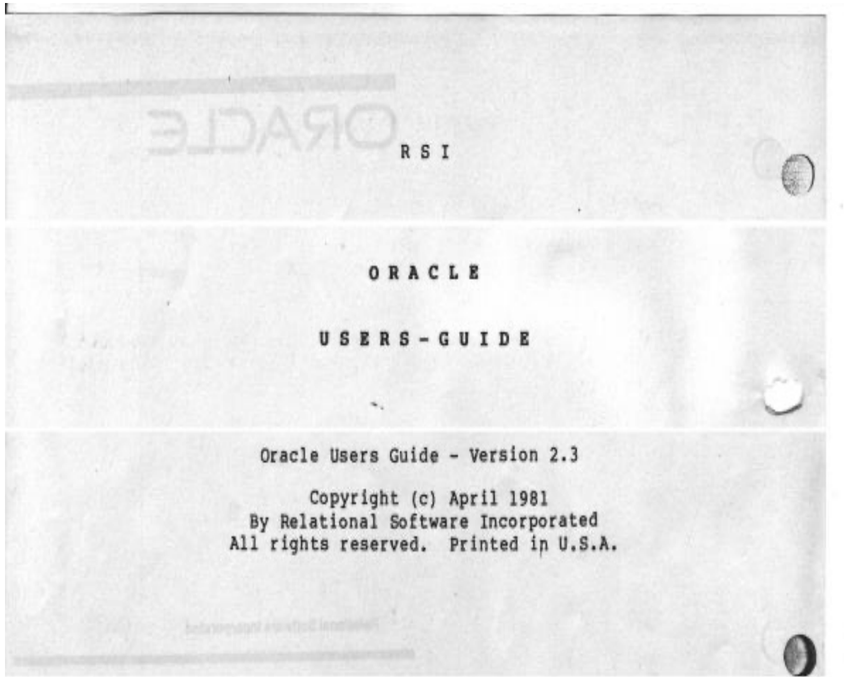
Databases at CERN

Oracle since 1982

- 105 Oracle databases, more than 11.800 Oracle accounts
- RAC, Active Data Guard, GoldenGate, OEM, RMAN, APEX, Cloud…
- Complex environment

Database on Demand (DBoD) since 2011

- ≈600 MySQL, ≈400 PostgreSQL, ≈200 InfluxDB
- Automated backup and recovery services, monitoring, clones, replicas
- HA MySQL clusters (Proxy + primary replica)


RSI

ORACLE

USERS-GUIDE

Oracle Users Guide - Version 2.3

Copyright (c) April 1981
By Relational Software Incorporated
All rights reserved. Printed in U.S.A.

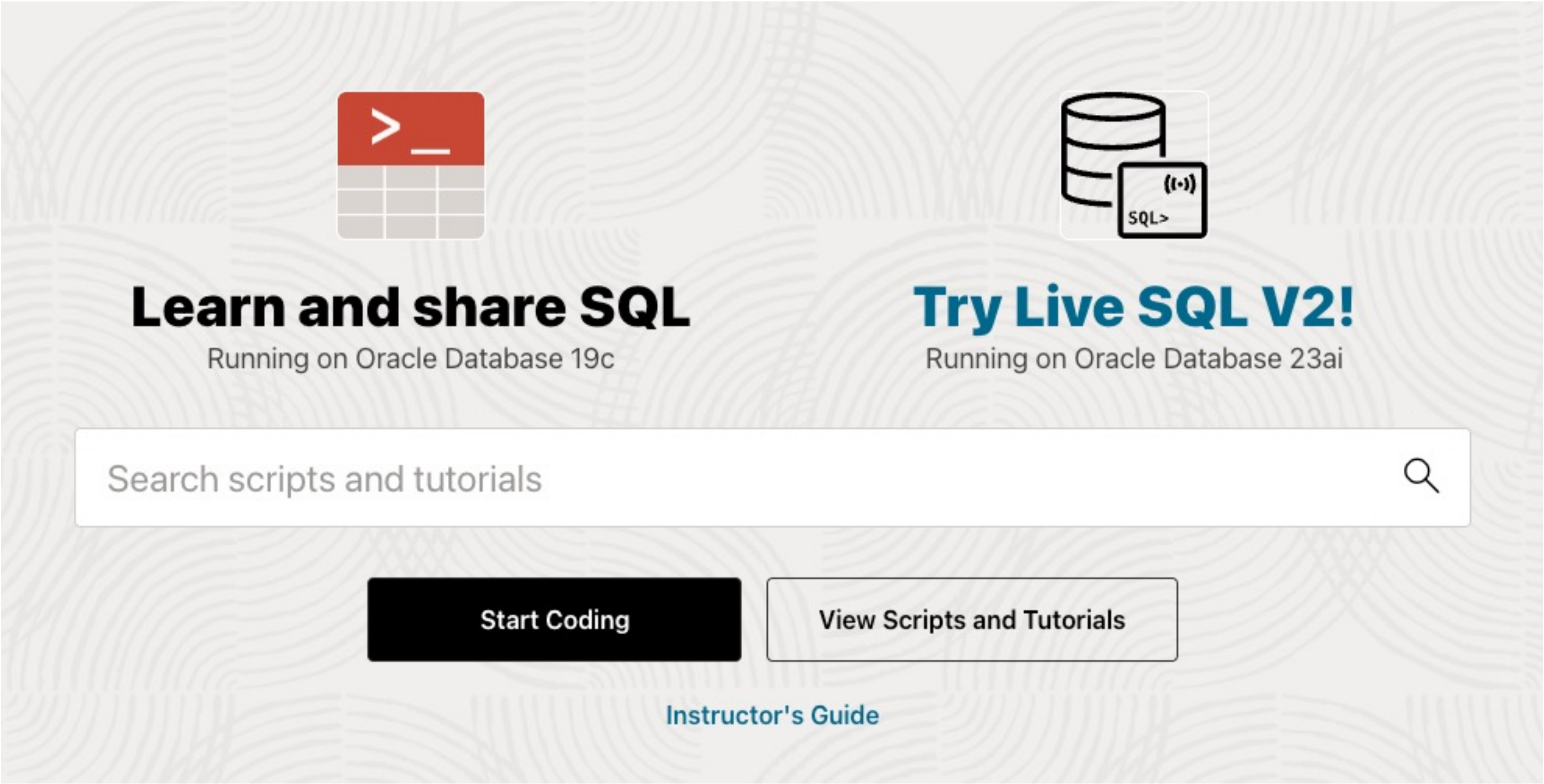
Size of the database environment

	Total size
Oracle	≈5 PB
DBoD (MySQL, PostgreSQL, InfluxDB)	≈150 TB
Backups	≈3 PB

Oracle 23ai

Where to test 23ai?

https://livesql.oracle.com ?



Where to test 23ai?

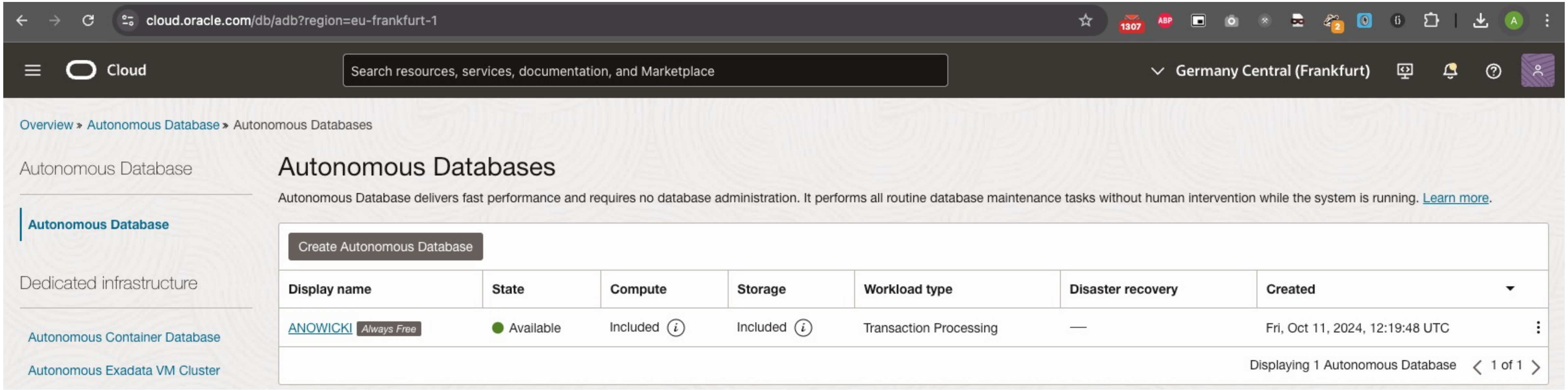
I'll spin up an Autonomous Database in the Oracle Cloud…

Autonomous Database with Oracle Database 23ai in the Paid tier is available in all commercial public cloud regions ***except* the following regions**: Colombia Central: Bogota (BOG), Saudi Arabia Central: Riyadh (RUH), Singapore West: Singapore (XSP), and Spain Central: Madrid (MAD).

Always Free Autonomous Database with Oracle Database 23ai **is available in the regions**: US West: Phoenix (PHX), US East: Ashburn (IAD), UK South: London (LHR), France Central: Paris (CDG), Australia East: Sydney (SYD), India West: Mumbai (BOM), Singapore (SIN), and Japan East: Tokyo (NRT).

https://docs.oracle.com/en-us/iaas/autonomous-database-serverless/doc/autonomous-always-free-23ai.html

Where to test 23ai?



Where to test 23ai?

Oracle Exadata Cloud@Customer
OCI Exadata Database Service
OCI Base Database Service

Oracle Database 23ai Free – https://www.oracle.com/database/free/get-started/

Available as: Docker image, VM VirtualBox, rpm for OEL & RHEL.

ARM version was released in November 2024:
https://blogs.oracle.com/database/post/announcing-oracle-database-23ai-free-container-images-for-armbased-apple-macbook-computers

ONNX model loading is
not supported on ARM yet

SELECT AI

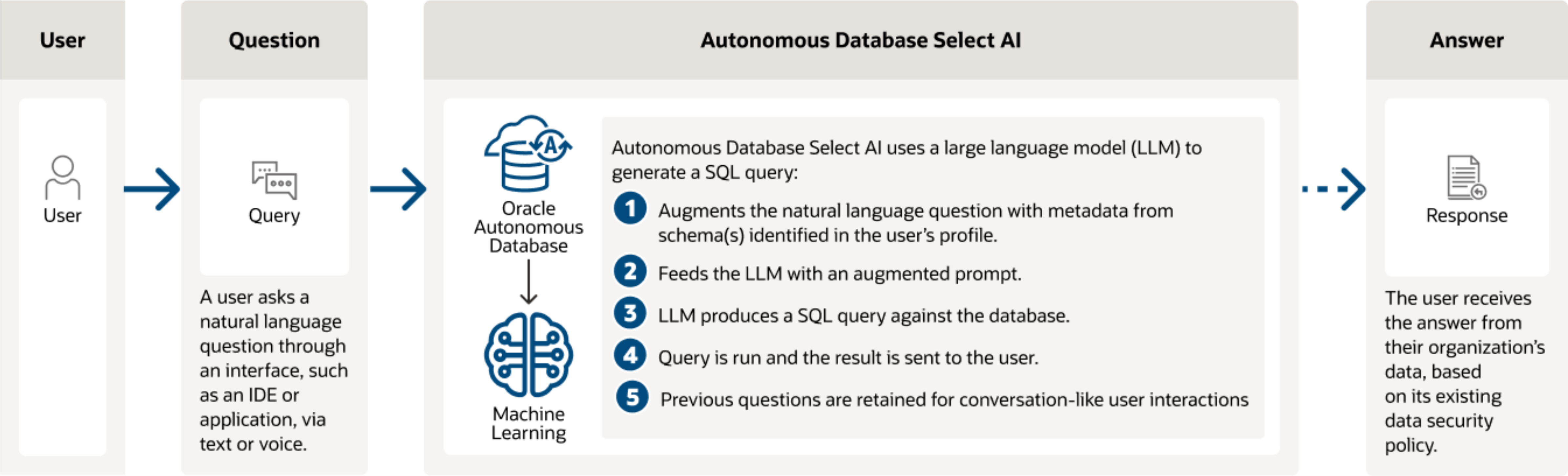
```
SQL> SELECT AI What are total sales of tom hanks movies
70,318.23

SQL> SELECT AI showsql What are total sales of tom hanks movies
SELECT SUM(sales_sample.list_price) AS total_sales
FROM moviestream.sales_sample
JOIN moviestream.movie ON sales_sample.movie_id = movie.movie_id
WHERE movie.cast LIKE '%Tom Hanks%'
```

SELECT AI

is only available in the cloud

forget what I said about docker

Select AI



https://www.oracle.com/autonomous-database/select-ai/

AI features – credentials

```
SQL> BEGIN
     DBMS_CLOUD.CREATE_CREDENTIAL (
       credential_name => 'OCI_CRED',
       user_ocid       => 'ocid1.user.oc1...',
       tenancy_ocid    => 'ocid1.tenancy.oc1.....',
       private_key     => 'MII......=',
       fingerprint     => '44:.......');
END;
/
```

AI features – profile

```
SQL> BEGIN
     DBMS_CLOUD_AI.create_profile(
      'OCI',
      '{"provider": "oci",
        "credential_name": "OCI_CRED",
        "oci_compartment_id": "ocid1.compartment.oc1......",
        "region" :"eu-frankfurt-1",
        "object_list": [{"owner": "ADMIN", "name": "employees"}]
       }');
END;
/
```

The object's data is not shared to the LLM.
Only metadata (column definitions, etc.)

```
SQL> select  * from employees;

        ID NAME           SALARY
---------- ---------- ----------
        21 Christi           100
        22 Andrzej           101
        23 Anna              999
```

AI features – examples

```
SQL> EXEC DBMS_CLOUD_AI.set_profile('OCI');

SQL> select ai chat how many people live in Poland;

RESPONSE
--------------------------------------------------------------------------------
As of 2021, the estimated population of Poland is approximately 38.6 million people

SQL> select ai how many employees do we have;

EmployeeCount
-------------
            3
```

AI features – examples

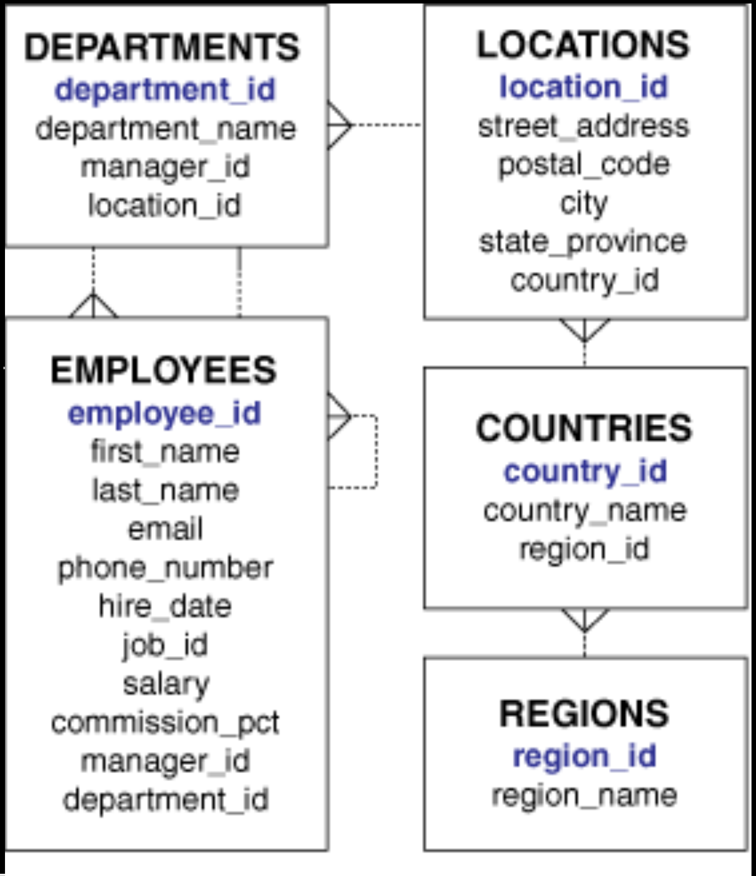
```
SQL> select ai showsql how many employees do we have;

RESPONSE
-------------------------------------------------------------------
SELECT COUNT(c."ID") AS "EmployeeCount"
FROM "ADMIN"."EMPLOYEES" e


SQL> select ai narrate how many employees do we have;

RESPONSE
--------------------
We have 3 employees.
```

DEMO

```
SQL> EXEC DBMS_CLOUD_AI.set_profile('OCI_HR_DEFAULT');

PL/SQL procedure successfully completed.

SQL> select ai showsql in which regions do we have departments?;

RESPONSE
--------------------------------------------------------------------------------------------------------------------------------
SELECT DISTINCT T2."REGION_NAME" FROM "HR"."DEPARTMENTS" T1 INNER JOIN "HR"."REGIONS" T2 ON T1."LOCATION_ID" = T2."REGION_ID"
```

in which regions

```
SQL> EXEC DBMS_CLOUD_AI.set_profile('OCI_HR_COHERE');

PL/SQL procedure successfully completed.

SQL> select ai showsql in which regions do we have departments?;

RESPONSE
--------------------------------------------------------------------------------------------
------------------
SELECT r."REGION_NAME" AS "Region", COUNT(DISTINCT d."DEPARTMENT_ID") AS "Number of Departments"
FROM "HR"."REGIONS" r
LEFT JOIN "HR"."DEPARTMENTS" d ON r."REGION_ID" = d."MANAGER_ID"
GROUP BY r."REGION_NAME"
```



AI features – flexibility

```
SQL> BEGIN
  DBMS_CLOUD_AI.create_profile(
   'OCI_HR_COHERE',
   '{"provider": "oci",
     "model": "cohere.command-r-08-2024",
     "credential_name": "OCI_CRED_",
     "oci_compartment_id": "...",
     "region" :"eu-frankfurt-1",
     "conversation": "true",
     "object_list": [{"owner": "HR"}]
    }');
END;
/
```

```
SQL> BEGIN
  DBMS_CLOUD_AI.create_profile(
   'OCI_HR_LLAMA',
   '{"provider": "oci",
     "model": "meta.llama-3.1-70b-instruct",
     "credential_name": "OCI_CRED_",
     "oci_compartment_id": "...",
     "region" :"eu-frankfurt-1",
     "conversation": "true",
     "object_list": [{"owner": "HR"}]
    }');
END;
/
```

Multiple providers supported:
OpenAI, Cohere, Azure OpenAI, OCI GenAI, Google, Anthropic, Hugging Face

Multiple models supported:
Llama, GPT, Cohere Command, Gemini, Claude

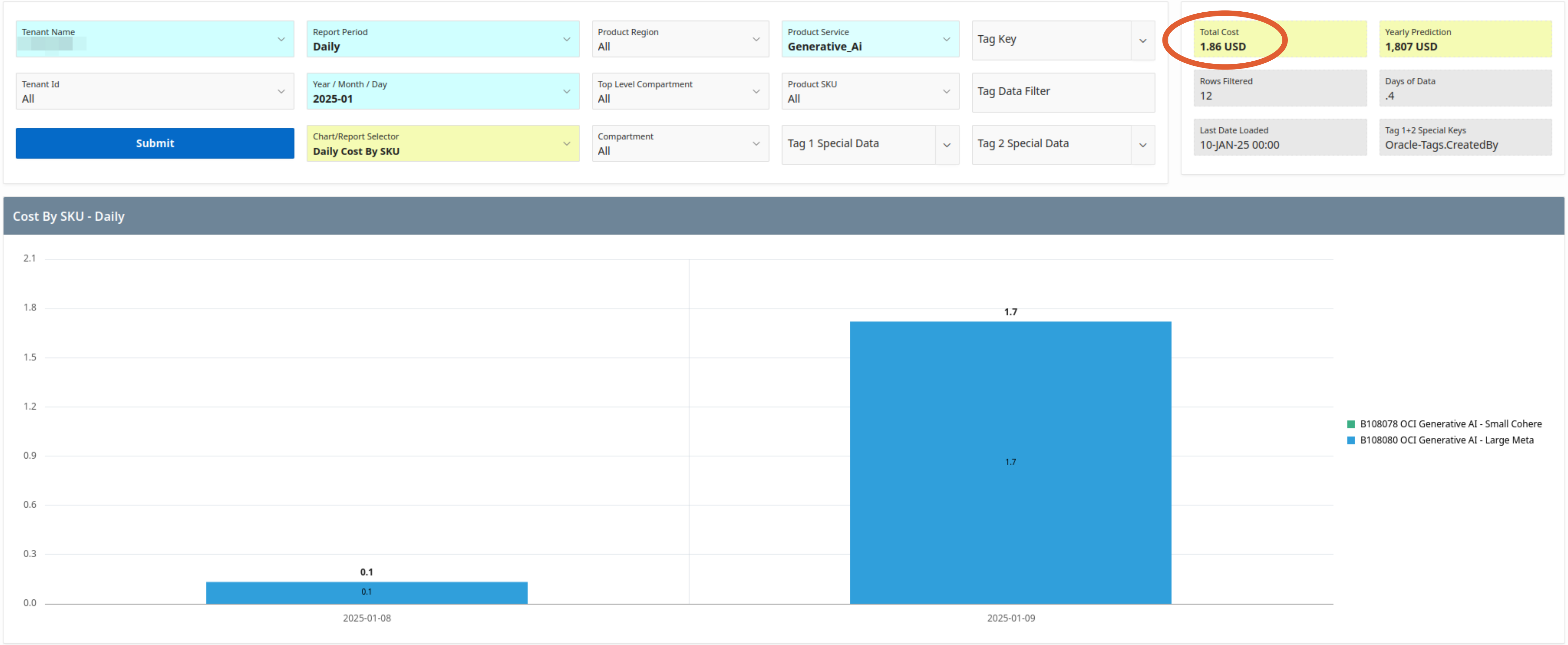
AI features – flexibility

```
SQL> BEGIN
  DBMS_CLOUD_AI.create_profile(
   'OCI_HR',
   '{"provider": "oci",
     "credential_name": "OCI_CRED_",
     "oci_compartment_id": "...",
     "region" :"eu-frankfurt-1",
      "conversation": "true",
     "object_list": [{"owner": "HR"}]
    }');
END;
/
```

Up to 10 past prompts can be included in your prompt

```
> select ai in which regions do we have departments?;

no rows selected

> select ai in which regions, countriedo we have departments?;

REGION_NAME                COUNTRY_NAME
-------------------------- ----------------------------------------
Europe                     Germany
Americas                   Canada
...

> select ai in which regions, do we have departments?;

REGION_NAME
---------------------------
Europe
Americas
```

Select AI – costs?



~~Let's build a simple beer recommendation system~~

VECTORS

VECTORS

In AI, a vector is an ordered list of numbers (scalars) that can represent a point in a multidimensional space. Mathematically, a vector is often written as:

$$\mathbf{v} = (v_1, v_2, \ldots, v_{n-1}, v_n)$$

n is the dimensionality of the vector.

EMBEDDINGS

Embeddings are numerical representations of real-world objects that machine learning (ML) and artificial intelligence (AI) systems use to understand complex knowledge domains like humans do.

For example, a bird-nest and a lion-den are analogous pairs, while day-night are opposite terms. Embeddings convert real-world objects into complex mathematical representations that capture inherent properties and relationships between real-world data.

EMBEDDING MODEL

An embedding model is a type of machine learning model designed to map high-dimensional or complex data (such as text, images, or categorical data) into lower-dimensional continuous vector spaces, known as embeddings. These embeddings capture the essential information or meaning of the data while preserving relationships between different data points in the original space.

How to put it all together?

Embedding
model

Input

(movie, picture, text, etc.)

🚨
That's a simplification.
Normally you would cut the text in chunks and
embed each chunk separately

Embedding

Vector

"Citrusy, sweet aroma"

[0.329, 0.911, 0.21, 0.37, …]

How to put it all together?

“Citrusy, sweet aroma”	`[0.329, 0.911, 0.21, 0.37, …]`
“Grapefruity taste, sweet aroma”	`[0.317, 0.818, 0.11, 0.36, …]`
“Harsh, spicy, roasted”	`[ 0.11,  0.01, 0.91, 0.87, …]`

Similar input should result in similar embedding (vector) values.

We can calculate distance between vectors to find similarity.

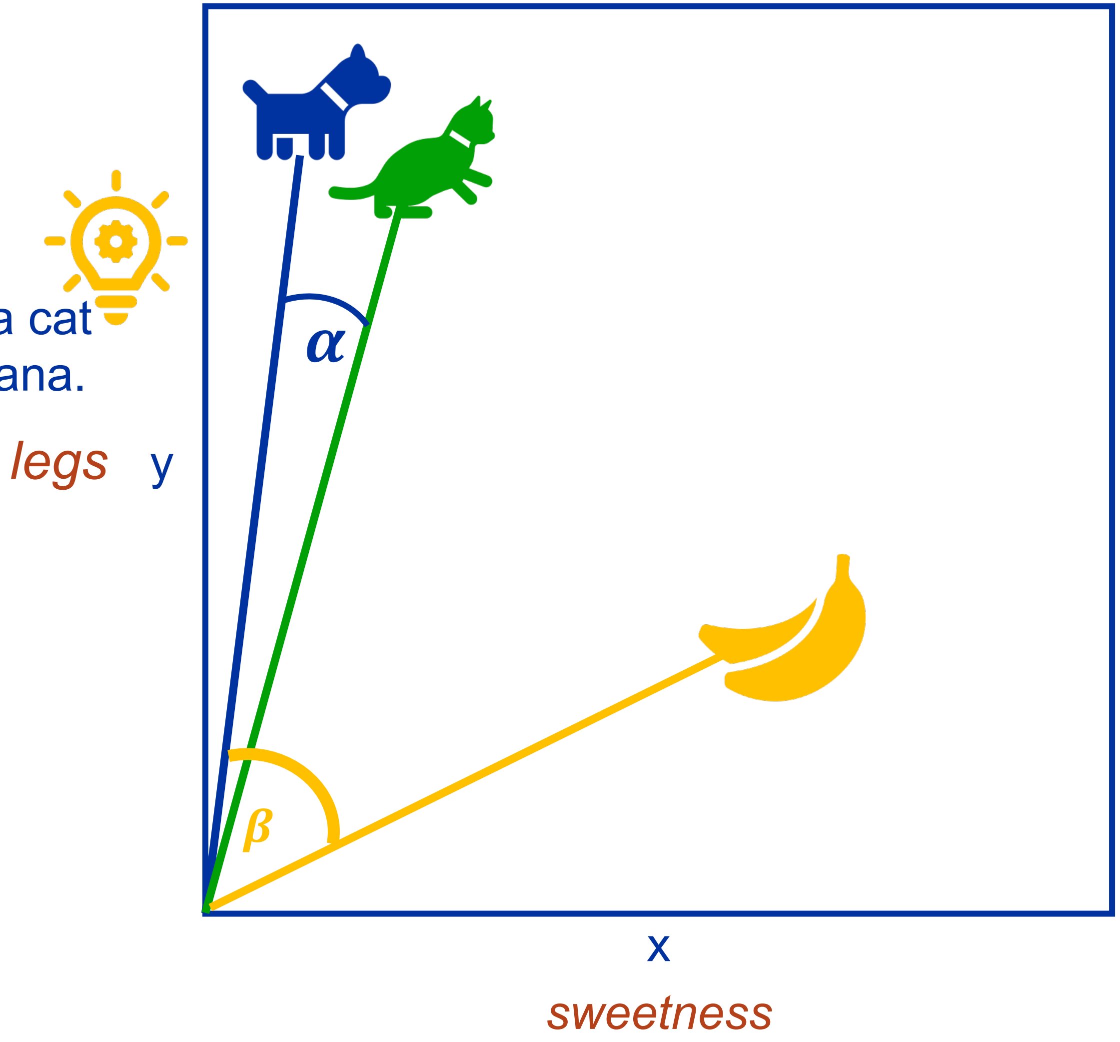
Our recommendation system will be based solely on similarity.

How to calculate similarity?

Cosine distance!

$\boldsymbol{\beta} > \boldsymbol{\alpha}$

A dog is more similar to a cat then it is similar to a banana.



Same thing hapens in the similarity search.

But we have 384 dimensions.

There are some limitations of the similarity

higher number = more similar

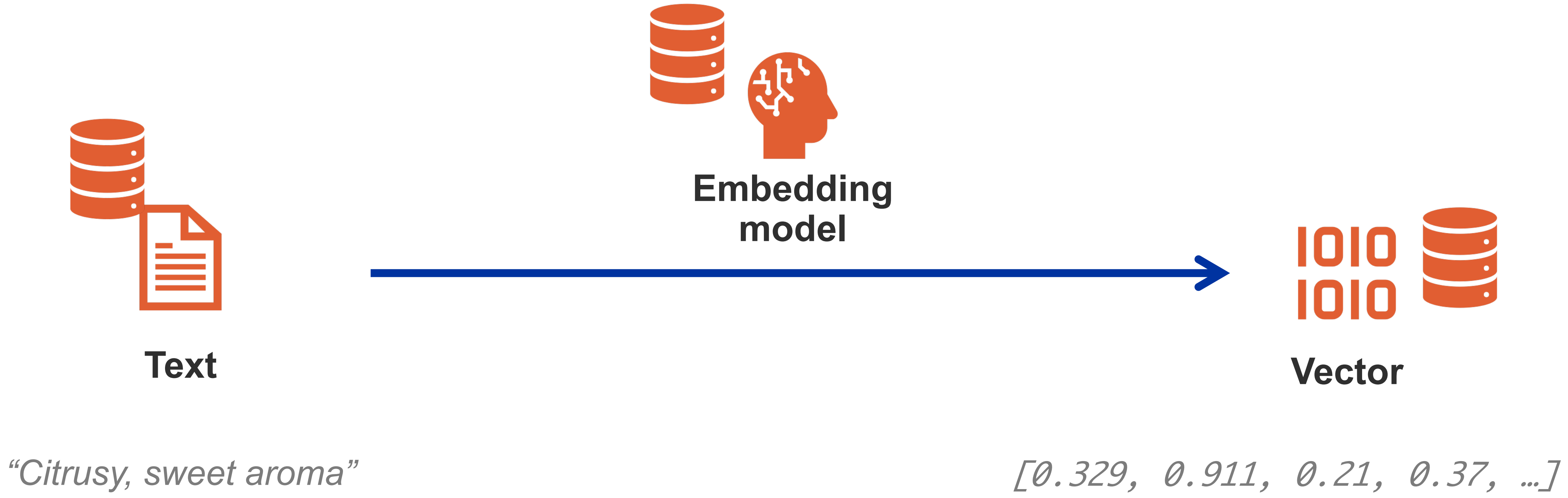
“healthy” vs “unhealthy”	`0.6788`
“healthy” vs “not healthy”	`0.8208`
“dog” vs “banana”	`0.2532`
“I like beer” vs “Table partitioning is an amazing feature of RDBMS”	`0.0311`
“I like beer” vs “I like indexes in databases”	`0.2238`
“I like to index my data” vs “I like indexes”	`0.7497`

Healthy vs Unhealthy are similar because both are adjectives, related to the health status

The “opposite” is not well defined. What is the opposite of ”king”? Queen? Prince? Poor man? 🍌?

Enough theory

Oracle 23ai – everything can be stored in the db



Let’s design a simple beer recommendation system

🍻 dataset

```
SQL> select beer_name, info from beers sample (1);

BEER_NAME          INFO
----------------- ---------------------------------------------------------------
Dreadnaught IPA   An Imperial India Pale Ale with an intense citrus hop aroma, a
                  huge malt body and a crisp finish.100 IBU
Ubu Ale           Our famous English-style Strong Ale, deep garnet red in color,
                  with a smooth, hardy taste and a nice warm feeling to follow.
Jose Pimiento     Barrel-aged Blonde w/ dried chiles.
Outdoor           A hoppy pilsner hopped with Saaz, Fieldwork Dank Blend.
```

This amazing dataset is available on Kaggle under creative commons license CC BY 4.0:

https://www.kaggle.com/datasets/ruthgn/beer-profile-and-ratings-data-set

VECTOR data type

```
SQL> desc beers
 Name                  Null?    Type
 --------------------- -------- -----------------------------
 ID                    NOT NULL NUMBER(38)
 BEER_NAME                      VARCHAR2(128)
 INFO                           VARCHAR2(4000)

SQL> alter table beers add vector_l12_v2 vector;
SQL> desc beers

 Name                  Null?    Type
 --------------------- -------- -----------------------------
 ID                             NUMBER(38)
 BEER_NAME                      VARCHAR2(128)
 INFO                           VARCHAR2(4000)
 VECTOR_L12_V2                  CLOB VALUE
```

EMBEDDING (the most popular way)

```python
#!/bin/env python3

from sentence_transformers import SentenceTransformer

embedding_model = "sentence-transformers/all-MiniLM-L12-v2"

model = SentenceTransformer(embedding_model)

data = "rich blend of roasted barley"

embedding = list(model.encode(data))

print(embedding)
```

```
[-0.006417383, -0.022299055, -0.07196472, -0.038730085, 0.015408011,
0.011460664, 0.031957585, -0.14295837, -0.06265083, 0.047036696, 0.05393924,
-0.017266361, -0.060880985, -0.090641975, -0.018470088, 0.043274913,
0.10671821, -0.01918215, -0.017627805, 0.007417538, -0.094217524,
0.048147723, 0.007045083, -0.0059344354, 0.031551342, 0.0060908115, ...
```

Configuring EMBEDDING MODEL in a local db

```
SQL> exec dbms_vector.load_onnx_model(
             directory=>'model_dir',
             file_name => 'all_MiniLM_L12_v2.onnx',
             model_name => 'ALL_MINILM_L12_V2',
             metadata => JSON('{
                    "function" : "embedding",
                    "embeddingOutput" : "embedding",
                    "input": {"input": ["DATA"]}
                 } ')
             );
```

Configuring EMBEDDING MODEL in the cloud

```
DECLARE
      ONNX_MOD_FILE VARCHAR2(100) := 'all_MiniLM_L12_v2.onnx';
      MODNAME VARCHAR2(500) := 'ALL_MINILM_L12_V2';
      LOCATION_URI VARCHAR2(200) := 'https://adwc4pm.objectstorage.us-ashburn-1.oci.customer-oci.com/p/eLddQappgBJ7jNi6Guz9m9LOtYe2u8LWY19GfgU8flFK4N9YgP4kTlrE9Px3pE12/n/adwc4pm/b/OML-Resources/o/';
BEGIN
      DBMS_CLOUD.GET_OBJECT(
          credential_name => 'MY_CLOUD_CRED',
          directory_name => 'DATA_PUMP_DIR',
          object_uri => LOCATION_URI||ONNX_MOD_FILE);
      DBMS_VECTOR.LOAD_ONNX_MODEL(
          directory => 'DATA_PUMP_DIR',
          file_name => ONNX_MOD_FILE,
          model_name => MODNAME);
END;
```

EMBEDDING using SQL

```
SQL> SELECT VECTOR_EMBEDDING(
                                    ALL_MINILM_L12_V2
                                    USING 'rich blend of roasted barley' as DATA
                                ) AS embedding;
EMBEDDING
----------------------------------------------------------------------------------
[-6.41739741E-003,-2.22990848E-002,-7.19647631E-002,-3.87300365E-002,1.54080233E
```

EMBEDDING PROCESS

```
update beers set vector_l12_v2 =
                          VECTOR_EMBEDDING(ALL_MINILM_L12_V2 USING info as data);
```

```
update beers set vector_l12_v2  = :embedded_value_calculated_outside_db
             where id = :2;
```

Embedding 3361 beer descriptions

Using a model in the database in Autonomous DB in Cloud (2 threads)	**~73s**
Embedding locally on Macbook M3 Pro (single threaded python code)	**~43s**
Embedding locally on Macbook M3 Pro (4 threads python code)	**~20s**

I used ChatGPT to parallelize my code

VECTOR INDEX (optional)

```
SQL> create vector index vector_index on beers(vector_l12_v2)
             ORGANIZATION INMEMORY NEIGHBOR GRAPH
             distance cosine
             with target accuracy 95;
```

```
SQL> show parameter vector_index_neighbor_graph_reload

NAME                                 TYPE        VALUE
------------------------------------ ----------- -----
vector_index_neighbor_graph_reload        string   OFF
```

VECTOR SEARCH

```
SQL>   select beer_name, info
       from beers
       order by vector_distance(
                     VECTOR_L12_V2,
                     :vector_calculated_outside_db,
                     cosine)
       fetch approximate first 5 rows only;

SQL>   select beer_name, info
       from beers
       order by vector_distance(
                     VECTOR_L12_V2,
                     VECTOR_EMBEDDING(ALL_MINILM_L12_V2 USING
                                      '&prompt' as data),
                     cosine)
       fetch approximate first 5 rows only;
```

VECTOR SEARCH

Prompt: 'lemon'

Sun Drift
Summon some sunshine with bright notes of citrus and black tea. A Brett-fermented ale with lemon zest and tea

Lemon Lager
Refreshingly cool taste produced with freshly squeezed lemon juice from Japanese Hiroshima Lemons, fermented and bottled as the perfect thirst-quencher, no matter what season.

Tocobaga Red Ale
Pours amber in color with notes of citrus and caramel. Citrus hop bitterness upfront with notes of caramel and an Amish bread sweetness. Citrus hop bitterness returns at the end for a long dry finish.75 IBU

Sorachi Ace
This is a saison featuring the rare Japanese-developed hop Sorachi Ace. The Sorachi Ace hop varietal is noted for its unique lemon zest/lemongrass aroma.

Femme Fatale Sudachi
A new version of Evil Twin?s classic brett fermented I.P.A. feauring Sudachi, an Asian citrus, for a nice citrusy note.

DEMO

What about real life usage?

How to put it all together?

Input (e.g. text) → **Chunking** → **Chunks** → **Embedding model** → **Vector**

Two households, both alike in dignity
(In fair Verona, where we lay our scene),
From ancient grudge break to new mutiny,
Where civil blood makes civil hands unclean
…

Two households, both alike in dignity	[0.329, 0.917, 0.211, 0.307, …]
(In fair Verona, where we lay our scene),	[0.129, 0.101, 0.561, 0.487, …]
From ancient grudge break to new mutiny,	[0.989, 0.091, 0.231, 0.962, …]
Where civil blood makes civil hands unclean	[0.439, 0.053, 0.513, 0.321, …]

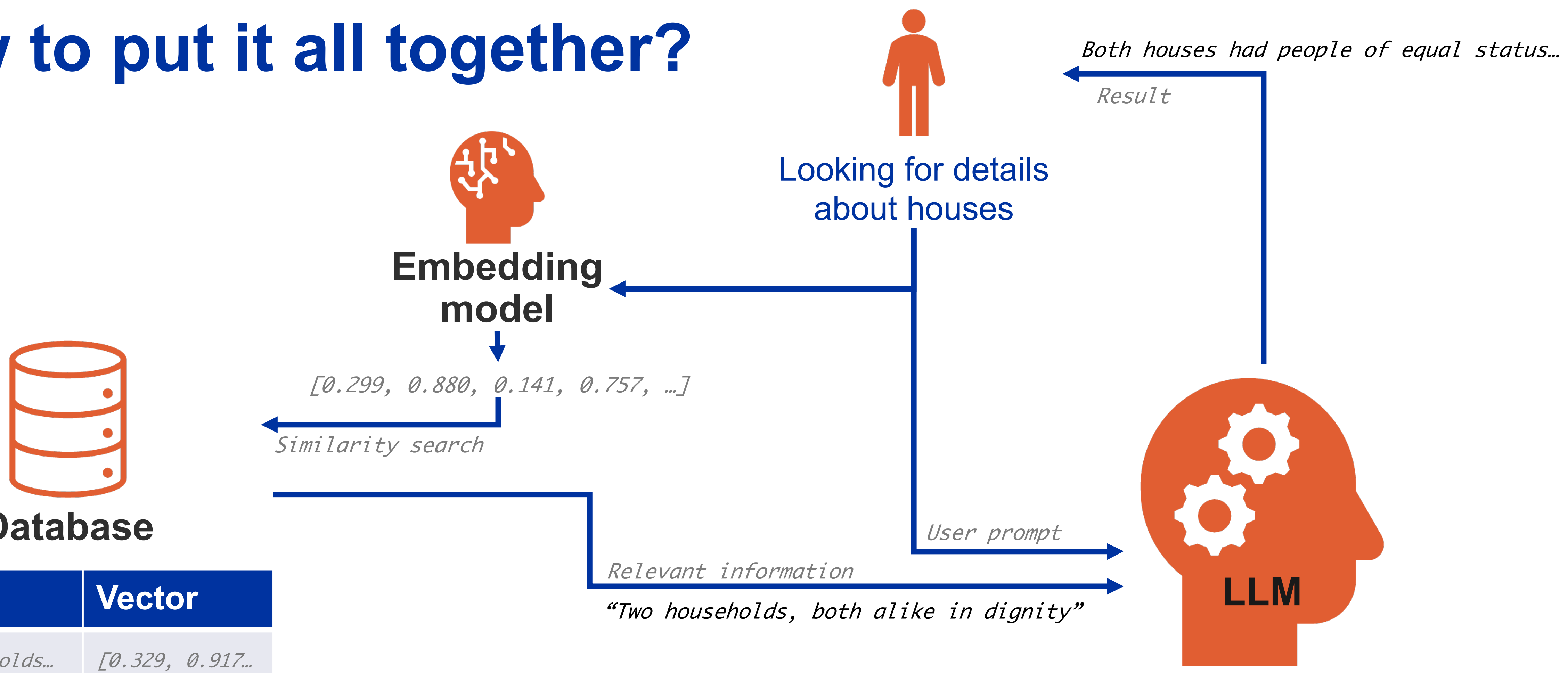
How to put it all together?

Chunks & Vector → Database

Two households, both alike in dignity	[0.329, 0.917, 0.211, 0.307, …]
(In fair Verona, where we lay our scene),	[0.129, 0.101, 0.561, 0.487, …]
From ancient grudge break to new mutiny,	[0.989, 0.091, 0.231, 0.962, …]
Where civil blood makes civil hands unclean	[0.439, 0.053, 0.513, 0.321, …]

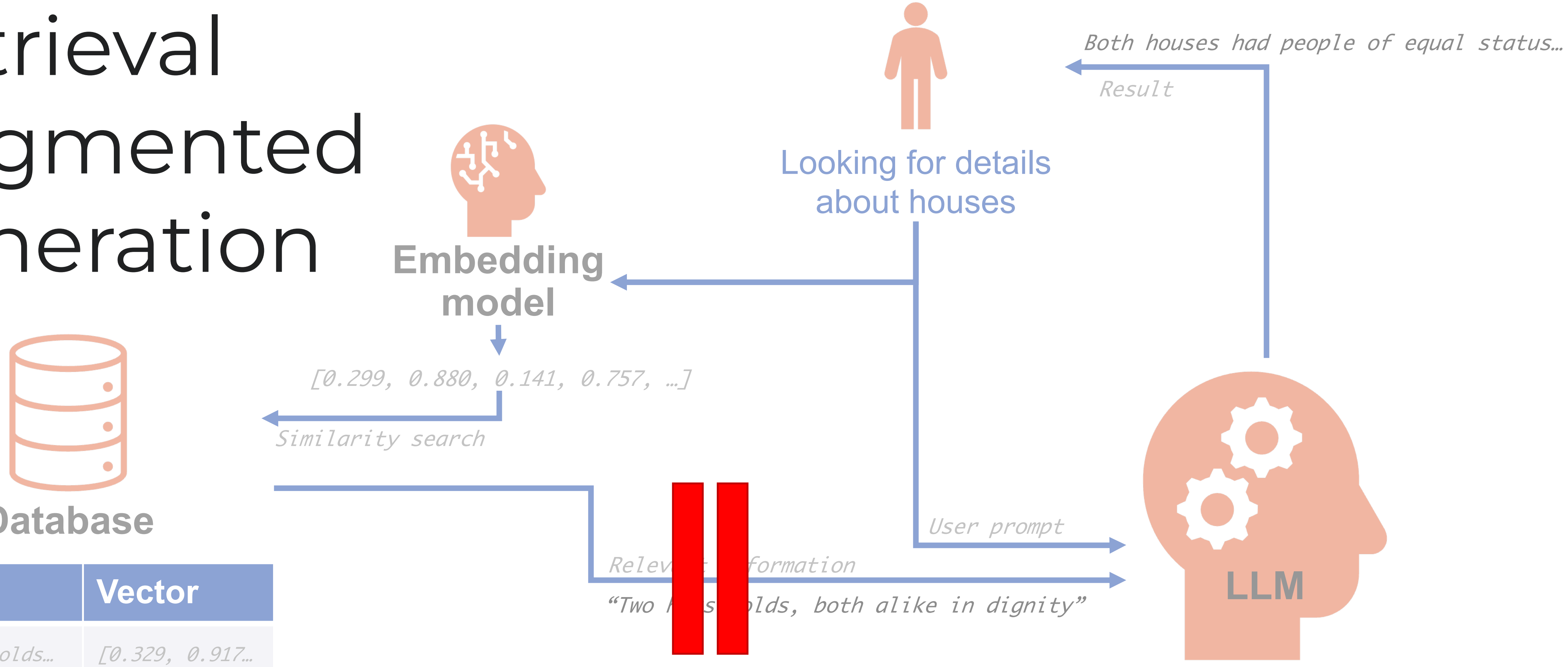
Text	Vector
Two households…	[0.329, 0.917…
(In fair Verona…	[0.129, 0.101…
From ancient…	[0.989, 0.091…
Where civil…	[0.439, 0.053…

How to put it all together?



Text	Vector
Two households…	*[0.329, 0.917…*
(In fair Verona…	*[0.129, 0.101…*
From ancient…	*[0.989, 0.091…*
Where civil…	*[0.439, 0.053…*

Retrieval **A**ugmented **G**eneration



Text	Vector
Two households…	*[0.329, 0.917…*
(In fair Verona…	*[0.129, 0.101…*
From ancient…	*[0.989, 0.091…*
Where civil…	*[0.439, 0.053…*

We would add a ReRank operation here
We can query from DB more information
Rank our information on relevance
Be selective in what we feed into the LLM

References

Oracle Docs	**https://docs.oracle.com/en/database/oracle/oracle-database/index.html**
Oracle Blogs	**https://blogs.oracle.com/database/**
LiveLabs	**https://apexapps.oracle.com/pls/apex/r/dbpm/livelabs/run-workshop?p210_wid=3831**
Beer dataset	**https://www.kaggle.com/datasets/ruthgn/beer-profile-and-ratings-data-set**

***Romeo and Juliet* by W. Shakespeare**


Remember to visit Science Gateway!
https://visit.cern

Thank you !

And remember… a dog is more similar to a cat then it is similar to a banana.

andrzejnowicki

andrzej.nowicki@cern.ch

www.andrzejnowicki.pl

Slides are available: