

# Oracle Database@ Google Cloud

Building GenAI Applications  
With Oracle Database@  
Google Cloud

Google Cloud



# Speakers



**Cody Fincher**  
Database Black Belts  
Oracle on Google Cloud

---

The magic of **GenAI**  
captured the world's  
imagination

Today organizations are  
looking to turn that  
excitement into  
**business results**



---

Your **AI** needs to be  
fueled by your  
business data

Your **business data**  
needs to be  
unlocked by AI



**85%**

companies experimenting or in  
limited production with GenAI

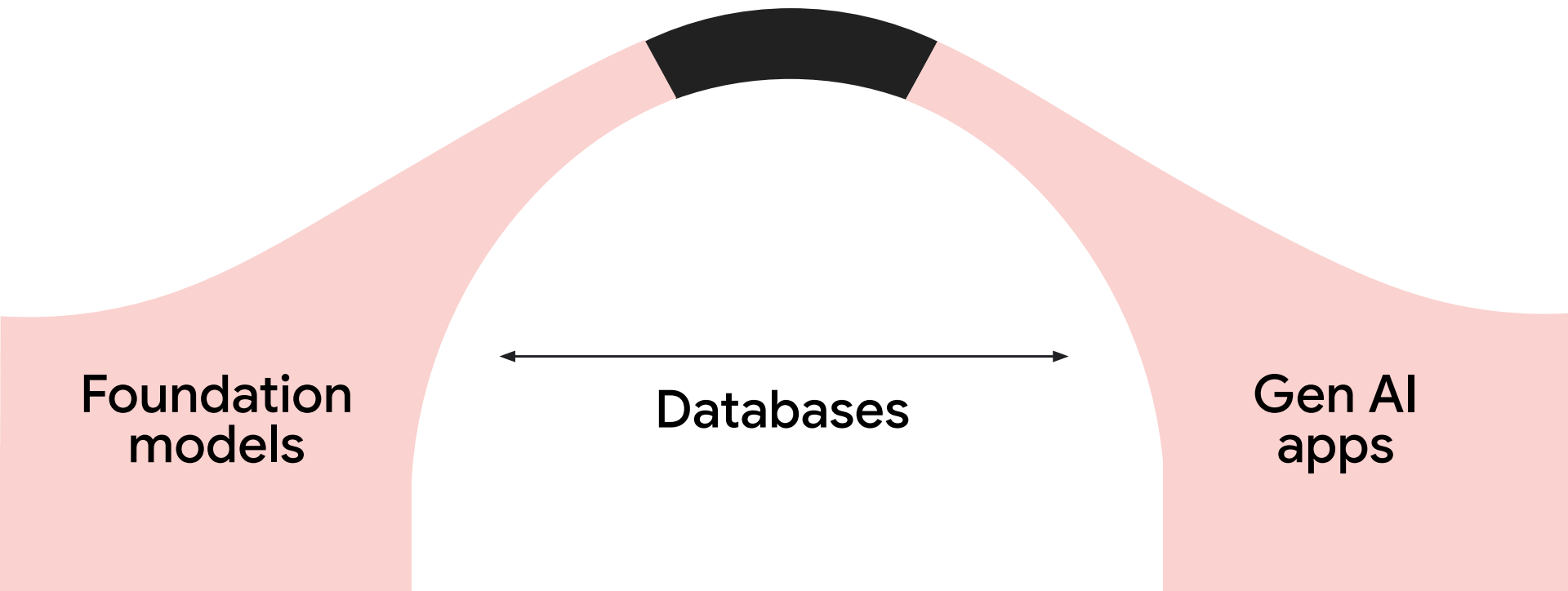
**5%**

companies that have GenAI  
applications implemented at scale

However,  
**existing data  
platforms** don't  
meet the needs  
of the **AI era**

Source: Wavestone, [2024 Data and AI Leadership Executive Survey](#)

# Operational data is key to exploiting the power of gen AI



Foundation  
models



Databases

Gen AI  
apps

# Accelerate modernization and innovation



Oracle database services and applications in Google Cloud



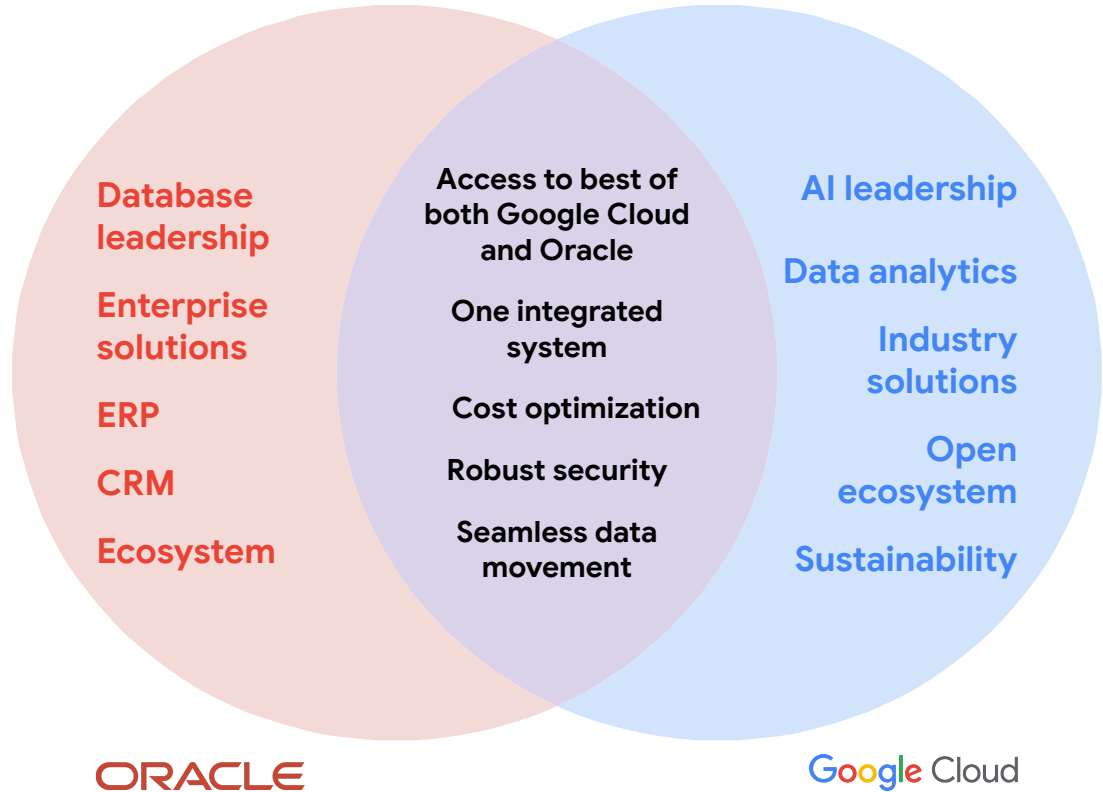
Flexible deployments on Google Cloud



OCI and Google Cross Cloud Interconnect between Google and Oracle

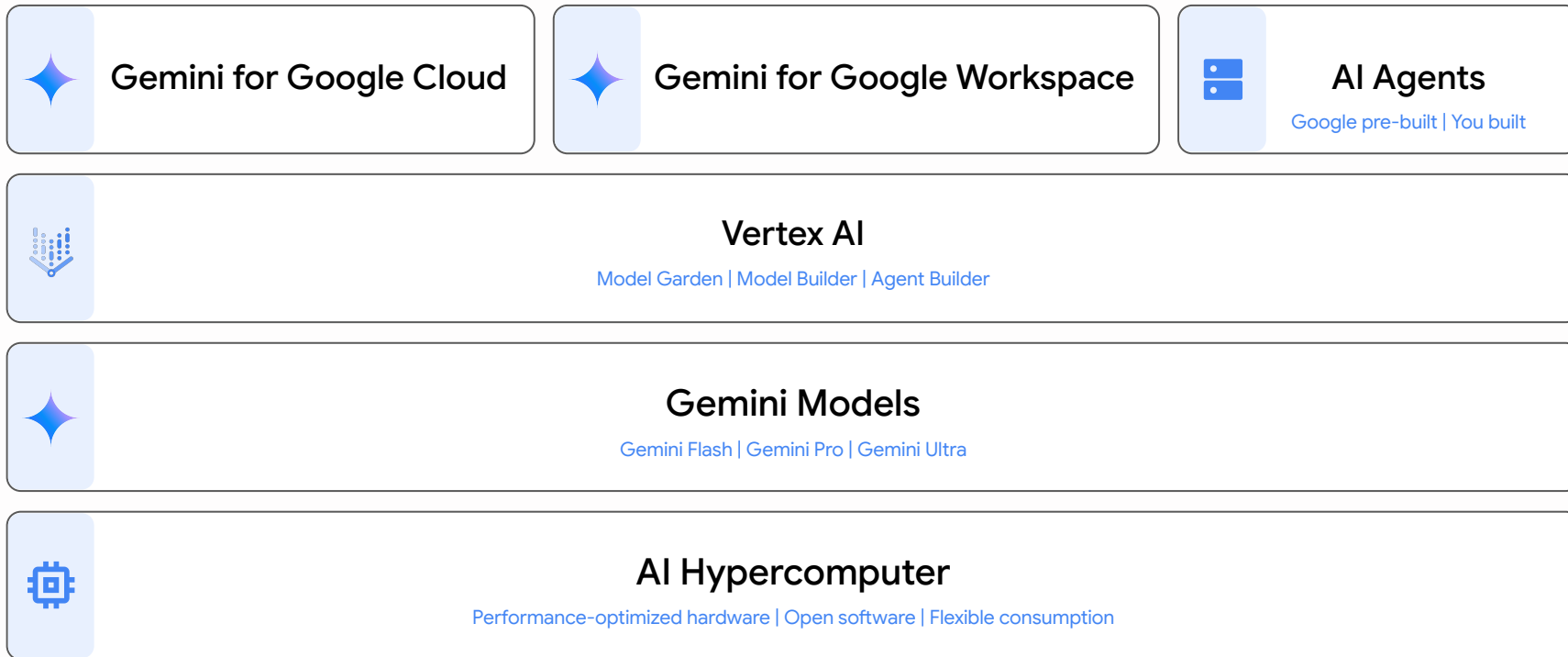
---

# A powerful combination of leading capabilities for modernization and innovation



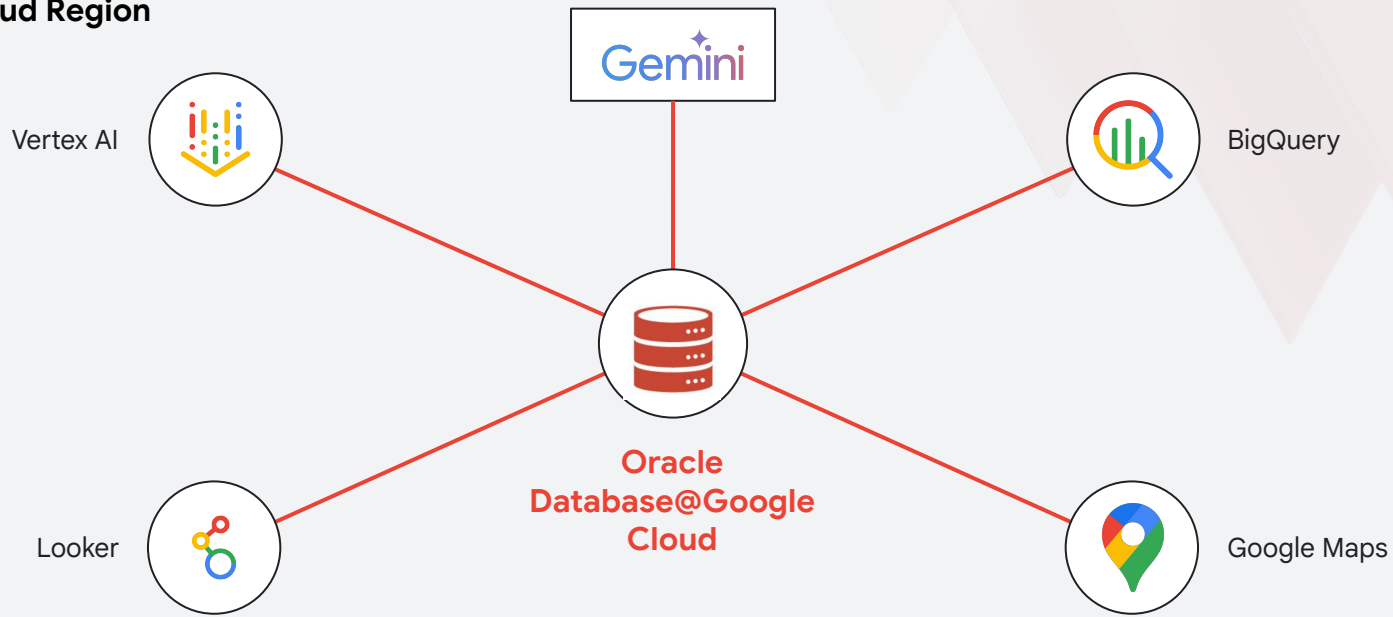
# Pioneering leader in AI

# Generative AI is built on a unified technology stack



# Google and Oracle unified vision

Google Cloud Region



# Google Cloud users benefit from an integrated **Data** ⇒ **Value** journey



ANY USER



**Data engineers**  
Clean, useful data



**ML engineers**  
Integrated intelligence



**Data scientists**  
Models that work



**Developers**  
Intelligent apps



**Data analysts**  
Query and analyze



**Business users**  
Insights Everywhere



**Consumers**  
Value

ANY DATA



Corporate and 3rd Systems



E-Commerce



SaaS Applications



IoT Devices



Social Media



Geospatial

ANY WORKLOAD

Multi Cloud Data Sources

DATA ENGINEERING



Dataproc



Dataflow

DATA ANALYSIS



BigQuery

MODEL DEVELOPMENT



Vertex AI

ML ENGINEERING



Vertex AI

INSIGHTS ACTIVATION



Vertex AI



Looker



Cloud Run



Cloud Composer



Vertex AI Pipelines

ANY INNOVATION



Decision Making



App Development



SaaS Applications



Exception Management



Operational Intelligence



Data Monetization

# Vertex AI



## AI Solution

Contact Center AI | Risk AI | Healthcare Data Engine | Search for Retail, Media and Healthcare

## Gemini Agents

## Build your own generative AI-powered agents

### Vertex AI Agent Builder

OOTB and custom Agents | Search  
Orchestration | Extensions | Connectors | Document Processors | Retrieval engines | Rankers | Grounding

### Vertex AI Model Builder

Prompt | Serve | Tune | Distill | Eval | Notebooks | Training | Feature Store | Pipelines | Monitoring

### Vertex AI Model Garden

Google | Open | Partner

Google Cloud Infrastructure (GPU/TPU) | Google Data Cloud

# Agent Builder

# Model Builder

## Model Garden

130+ Curated Models Including...



### Gemini 2.5 Pro / Flash

Multimodal reasoning across a wide range of tasks



### Imagen

Create and edit images from simple prompts



### Gemma

Family of lightweight, state-of-the-art open models

AI

### Claude on Vertex AI

Claude Haiku, Sonnet & Opus



kaggle

### Hugging Face & Kaggle Models

Deployable on Vertex AI



Meta

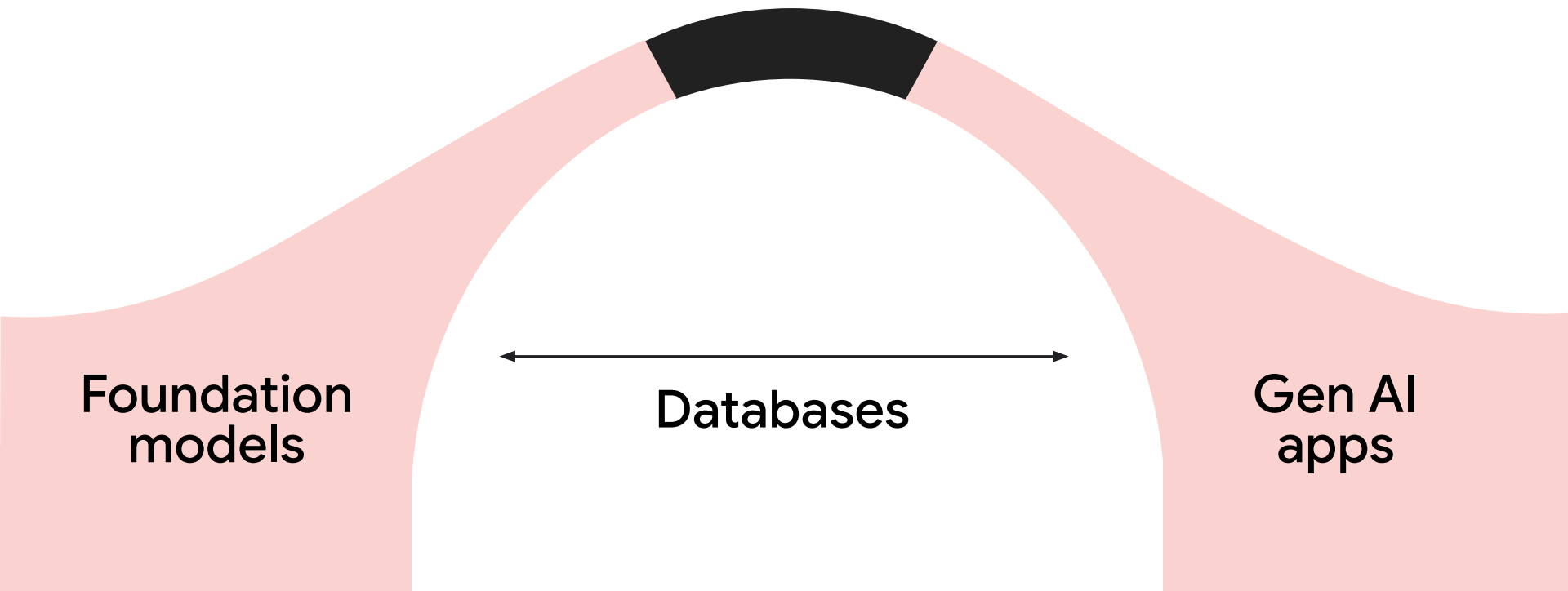
### Open Models on Vertex AI

Mixtral, Image Bind, DITO, Llama and more



# Connecting to your own Data

# Operational data is key to exploiting the power of gen AI



# What are Embeddings?

Numeric representations (vectors) of words and phrases that help define the way that the phrases relates to various concepts and other words.

## Traditional Search:

"espresso" matches only "espresso"

## Embedding Search:

"espresso" also finds:

- "strong coffee" (close in meaning)
- "concentrated brew" (similar concept)
- "intense flavor" (related characteristic)

# Vectorized Search

```
1 SELECT * FROM products  
2 WHERE description LIKE '%strong%'  
3   AND description NOT LIKE '%bitter%'
```

*\*This could further improved with concepts like metaphone, soundex, and full text search.*



```
1 SELECT * FROM products  
2 WHERE vector_distance(embeddings, :user_embedding) < 0.8
```

# Oracle Database Schema

```
1  -- Create product table with Oracle 23AI vector support
2  CREATE TABLE product (
3      id NUMBER GENERATED BY DEFAULT ON NULL AS IDENTITY PRIMARY KEY,
4      company_id NUMBER NOT NULL,
5      name VARCHAR2(255 CHAR) NOT NULL,
6      current_price NUMBER NOT NULL,
7      description VARCHAR2(2000 CHAR) NOT NULL,
8      embedding VECTOR(768, FLOAT32),
9      embedding_generated_on TIMESTAMP WITH TIME ZONE,
10     created_at TIMESTAMP WITH TIME ZONE DEFAULT ON NULL CURRENT_TIMESTAMP NOT NULL,
11     updated_at TIMESTAMP WITH TIME ZONE DEFAULT ON NULL FOR INSERT AND UPDATE CURRENT_TIMESTAMP NOT NULL,
12     CONSTRAINT fk_product_company FOREIGN KEY (company_id)
13         REFERENCES company(id) ON DELETE CASCADE
14 );
15
16 -- Create vector index for similarity search on products
17 CREATE VECTOR INDEX idx_product_embedding ON product(embedding)
18 ORGANIZATION NEIGHBOR PARTITIONS
19 DISTANCE COSINE
20 WITH TARGET ACCURACY 95;
```

# Populating Embeddings

```
1  async def create_embedding(self, text: str) -> list[float]:
2      """Create embeddings using Vertex AI."""
3      try:
4          # Use the native Vertex AI embedding model
5
6          model = TextEmbeddingModel.from_pretrained(self.embedding_model)
7          embeddings = await model.get_embeddings_async([text])
8
9          if embeddings and len(embeddings) > 0:
10             return cast("list[float]", embeddings[0].values)
11             # Fallback to mock embedding for development
12
13     except Exception:
14         # Log the error and fallback to mock embedding
15         logger.exception("Embedding generation failed, using fallback")
16         return [0.0] * 768 # Standard embedding dimension
17     else:
18         return [0.0] * 768
```

```
- Yummy Waffles: [37.7749, -122.4194]
- Pancakes: [0.2, -0.5, 0.8, ...]
```

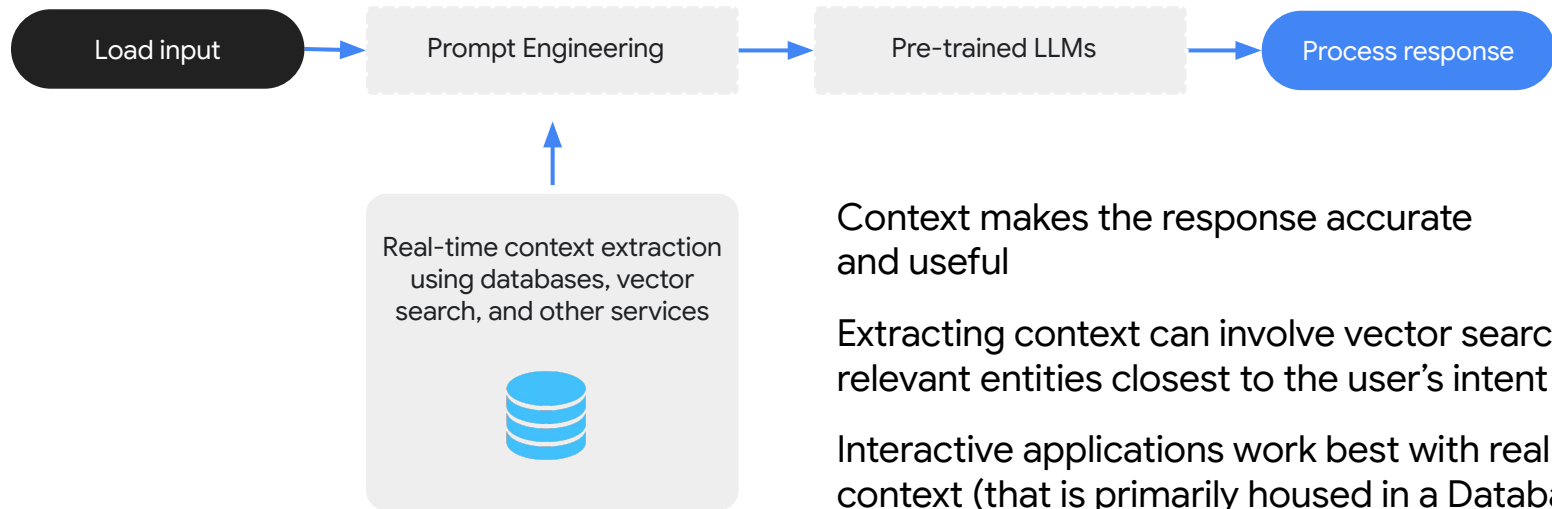
```
1  UPDATE product
2  SET embedding = :embedding,
3     embedding_generated_on = SYSTIMESTAMP
4  WHERE id = :id
```

# What If I have many vectors?

```
1  async def submit_batch_embedding_job(
2      self, input_path: str, output_path: str, job_display_name: str | None = None
3  ) -> BatchPredictionJob:
4      """Submit a batch prediction job for embeddings."""
5      if job_display_name is None:
6          job_display_name = f"bulk-embeddings-{uuid.uuid4().hex[:8]}"
7
8      # Input and output URIs
9      input_uri = f"gs://{self.bucket_name}/{input_path}"
10     output_uri = f"gs://{self.bucket_name}/{output_path}"
11
12     await logger.ainfo(f"Submitting batch job: {job_display_name}")
13     await logger.ainfo(f"Input URI: {input_uri}")
14     await logger.ainfo(f"Output URI: {output_uri}")
15
16     # Create batch prediction job
17     job = aiplatform.BatchPredictionJob.create(
18         job_display_name=job_display_name,
19         model_name=f"publishers/google/models/{self.embedding_model}",
20         instances_format="jsonl",
21         predictions_format="jsonl",
22         gcs_source=input_uri,
23         gcs_destination_prefix=output_uri,
24     )
25
26     await logger.ainfo(f"Batch job submitted with ID: {job.resource_name}")
27     return job
```

# Grounding an LLM with RAG

Retrieval Augmented Generation



# System Prompt

You are a helpful coffee expert for Cymbal Coffee. Give quick, friendly recommendations and advice.

Keep responses SHORT and conversational - this is a chat interface:

- 1-3 sentences max unless they ask for details
- Be direct and helpful
- Focus on practical recommendations
- No bullet points or long explanations
- Sound natural and friendly like you're talking to a customer at the counter

# Persona

```
"persona": "Coffee Novice"

"description": "New to coffee, needs simple explanations"

"language_style": "Simple, friendly, encouraging, avoid jargon"

"focus_areas": [
    "basic coffee types",
    "simple brewing methods",
    "starter recommendations",
    "easy-to-understand comparisons",
],

"example_responses":
    "brewing": "Let me explain this in simple terms..."
    "beans": "Think of coffee beans like different types of apples..."
    "recommendation": "For someone new to coffee, I'd suggest starting with..."

"system_prompt_addon": "You are helping someone new to coffee in a friendly chat.
Keep it SIMPLE and SHORT. Use everyday language, no coffee jargon. Be encouraging.
Give 1-2 sentence answers unless they ask for more. Think of it like helping a
friend who's never had good coffee before."
```



“I need something bold”  
“Caffeine please”  
“How’s it going?”

# Intent Exemplars

### Intent Detection

ORACLE VECTOR SEARCH QUERY

```
SELECT intent_type,  
       VECTOR_DISTANCE(embedding,  
                       :query_embedding, COSINE) AS similarity  
FROM intent_exemplars  
WHERE VECTOR_DISTANCE(embedding,  
                      :query_embedding, COSINE) < 0.3  
ORDER BY similarity  
FETCH FIRST 1 ROW ONLY
```

DETECTION RESULTS

Intent	PRODUCT_RAG
Type	Product Search
Processing Time	2.3ms

WHAT THIS MEANS

Your query was classified as a product search, which triggers vector similarity search against our coffee product database using Oracle 23AI.



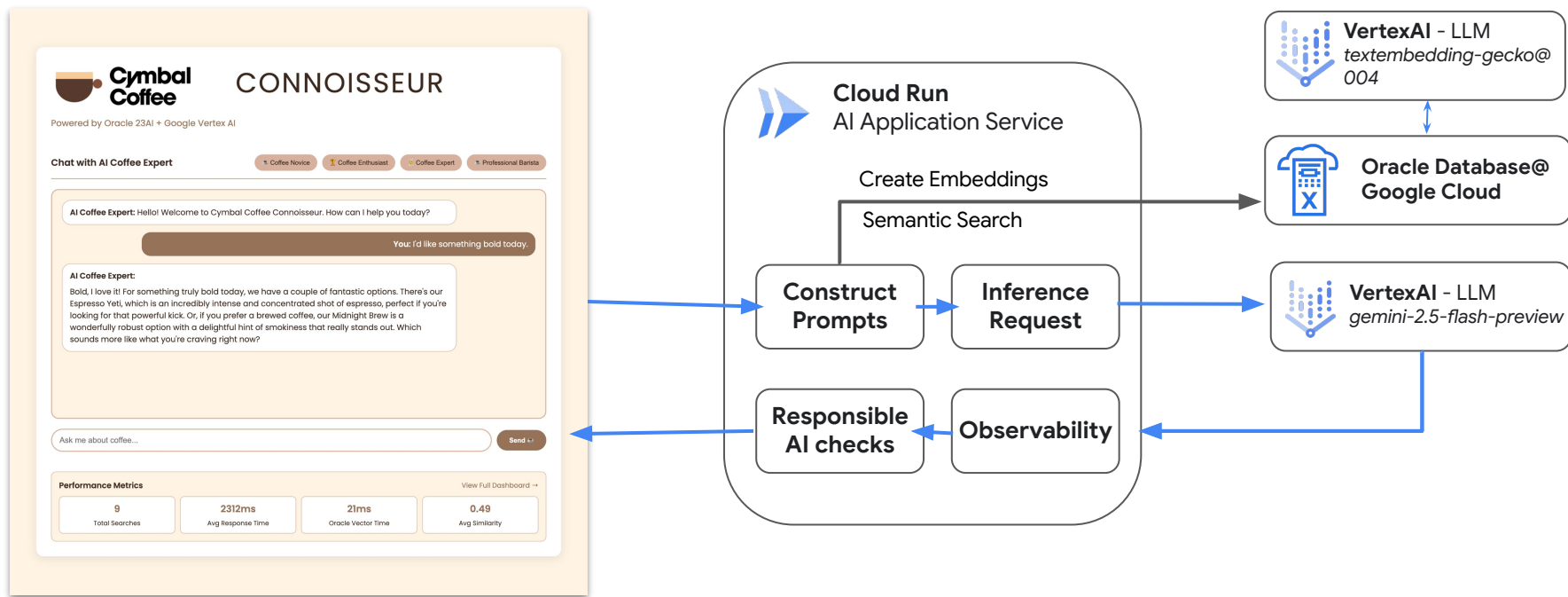
# Vector Similarity

```
1  ### Product Vector Search
2
3  ```sql
4  -- Find relevant products using semantic search
5  SELECT
6      p.id,
7      p.name,
8      p.description,
9      p.current_price,
10     c.name as company_name,
11     1 - VECTOR_DISTANCE(p.embedding, :query_vector, COSINE) as relevance_score
12 FROM product p
13 JOIN company c ON p.company_id = c.id
14 WHERE p.embedding IS NOT NULL
15 ORDER BY VECTOR_DISTANCE(p.embedding, :query_vector, COSINE)
16 FETCH FIRST 10 ROWS ONLY;
17 ```
```

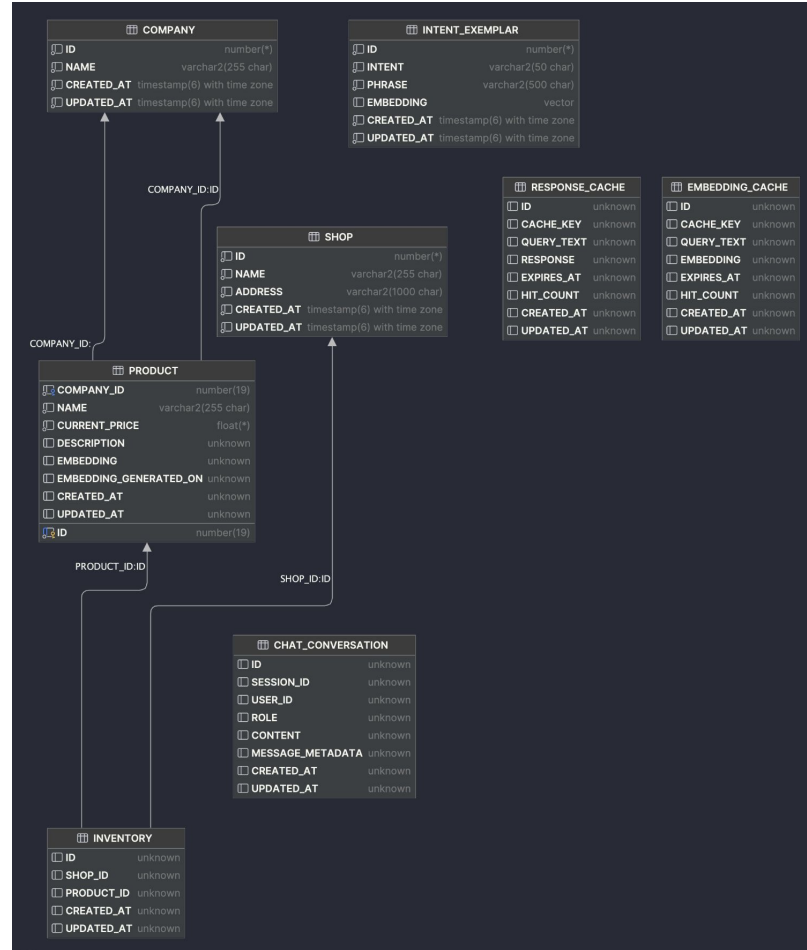
# It's All SQL!

```
1 # Join in vector distance with spatial in a single query
2
3 ```sql
4 -- Find coffee that's similar AND in stock nearby
5 SELECT p.name, p.price, s.address, i.quantity,
6        VECTOR_DISTANCE(p.embedding, :taste_vector, COSINE) as match_score
7 FROM products p
8 JOIN inventory i ON p.id = i.product_id
9 JOIN shops s ON i.shop_id = s.id
10 WHERE VECTOR_DISTANCE(p.embedding, :taste_vector, COSINE) < 0.7
11        AND ST_Distance(s.location, :user_location) < 5000 -- Within 5km
12        AND i.quantity > 0
13 ORDER BY match_score, distance
14 FETCH FIRST 10 ROWS ONLY;
15 ```
```

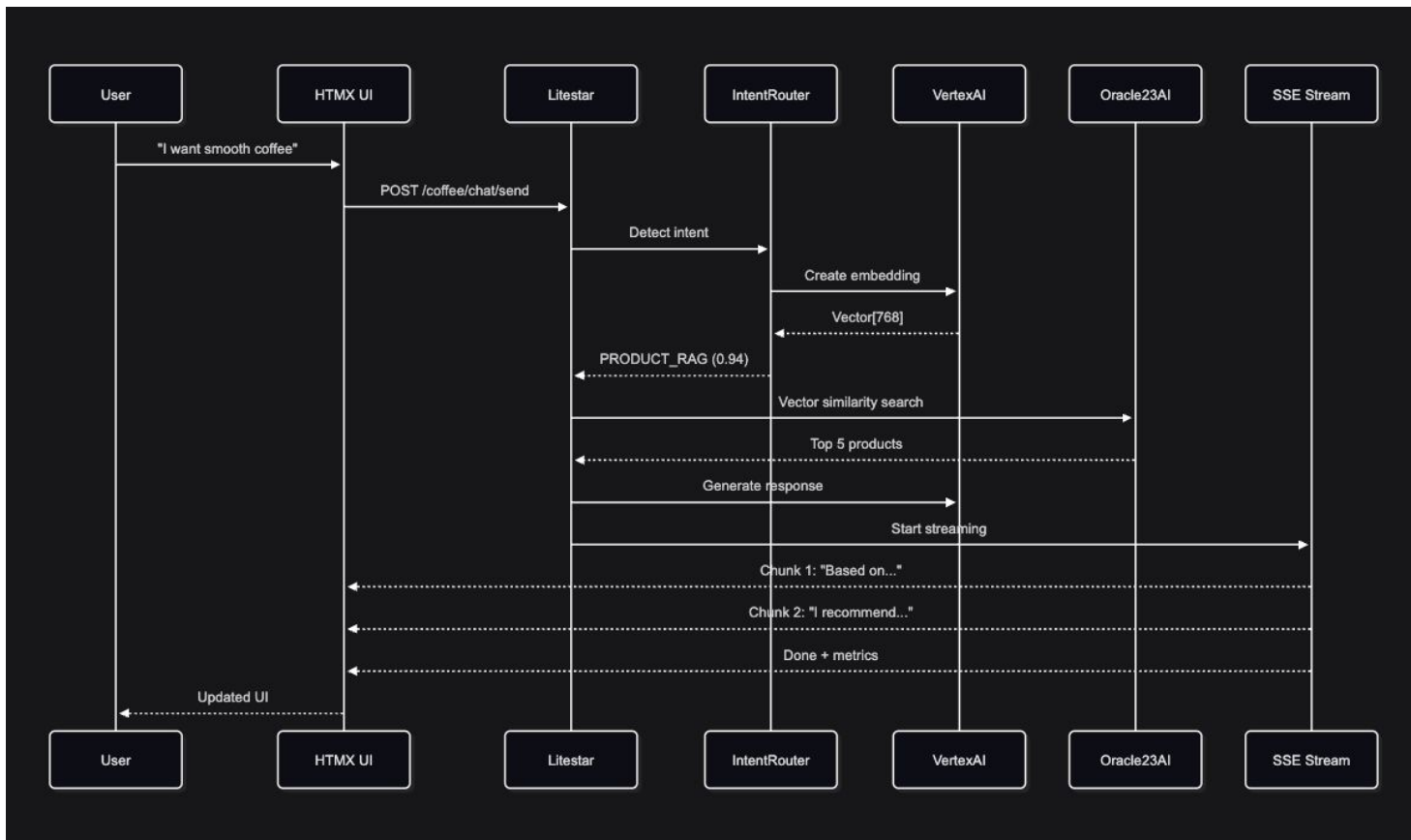
# RAG-capable GenAI Application with Oracle DB



# A Practical Example



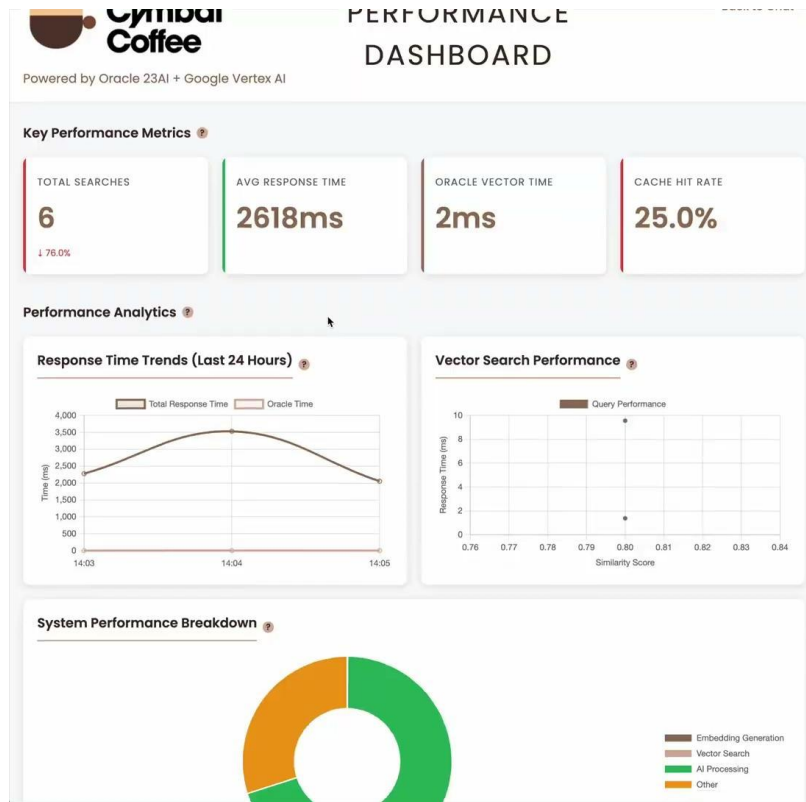
# Data Flow





Let's  
Demo!

# Vector Search Demonstration



# Coffee Chat Demonstration

The screenshot shows a web interface for 'Cymbal Coffee CONNOISSEUR'. At the top left is the Cymbal Coffee logo, and to its right is the word 'CONNOISSEUR'. Below the logo, it says 'Powered by Oracle 23AI + Google Vertex AI'. A navigation bar contains four buttons: 'Coffee Novice', 'Coffee Enthusiast' (which is selected), 'Coffee Expert', and 'Professional Barista'. The main chat area shows a message from the 'AI Coffee Expert' saying 'Hello! Welcome to Cymbal Coffee Connoisseur. How can I help you today?'. Below the chat area is a text input field with the placeholder text 'Ask me about coffee varieties, brewing methods...' and a 'Send' button. At the bottom, a 'Performance Metrics' section displays four statistics: '19 Total Searches', '1593ms Avg Response Time', '9ms Oracle Vector Time', and '0.54 Avg Similarity'. A 'View Full Dashboard' link is also present. A green checkmark icon is located in the bottom right corner of the interface.

**Cymbal Coffee** CONNOISSEUR

Powered by Oracle 23AI + Google Vertex AI

Chat with AI Coffee Expert

Coffee Novice **Coffee Enthusiast** Coffee Expert Professional Barista

AI Coffee Expert: Hello! Welcome to Cymbal Coffee Connoisseur. How can I help you today?

Ask me about coffee varieties, brewing methods... Send

**Performance Metrics** [View Full Dashboard](#)

19 Total Searches	1593ms Avg Response Time	9ms Oracle Vector Time	0.54 Avg Similarity
----------------------	-----------------------------	---------------------------	------------------------

# Start your transformation journey today

01

Discuss business priorities related to current Oracle investments.

02

Assess pilot strategy for Oracle based solutions on Google Cloud, Oracle Database@Google Cloud, and Oracle Interconnect for Google Cloud.

03

Start a pilot project.

# Thank you



[cloud.google.com/solutions/oracle](https://cloud.google.com/solutions/oracle)



[github.com/cofin/oracledb-vertexai-demo](https://github.com/cofin/oracledb-vertexai-demo)

**Google** Cloud

