Operationalizing ML & AI with MemSQL
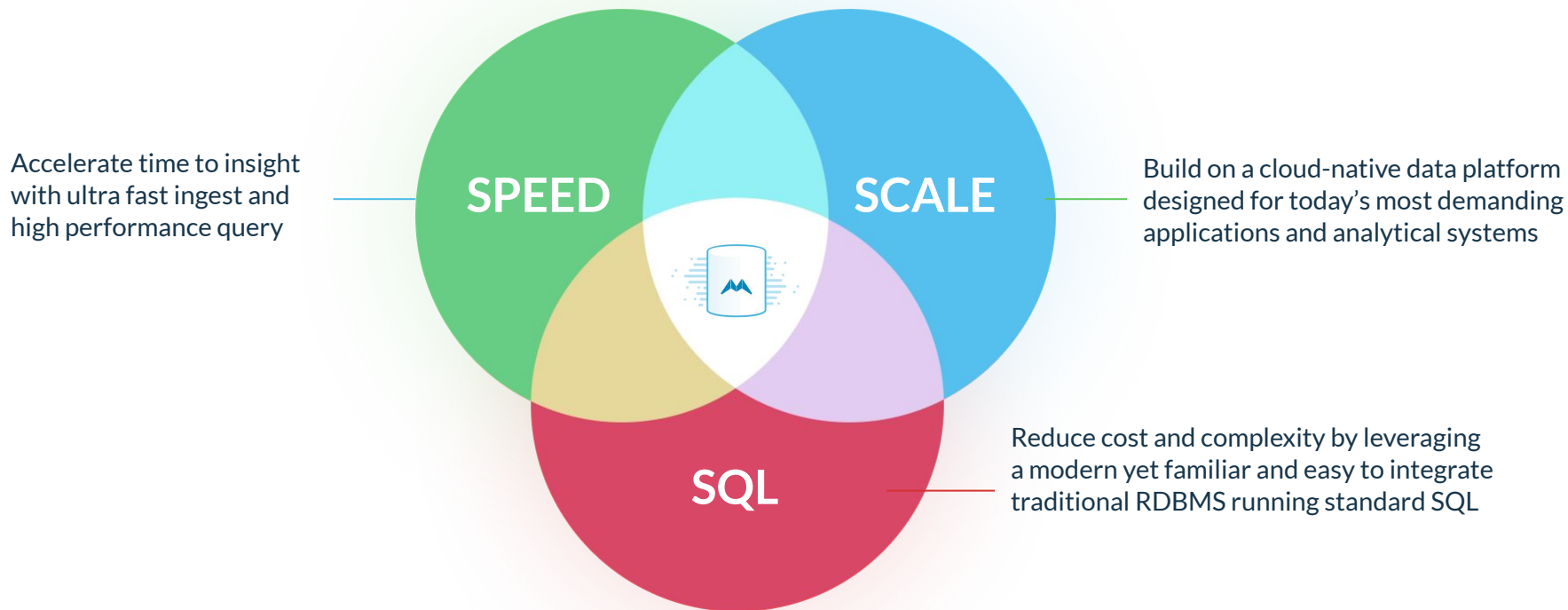
MEMSQL

Today's Agenda

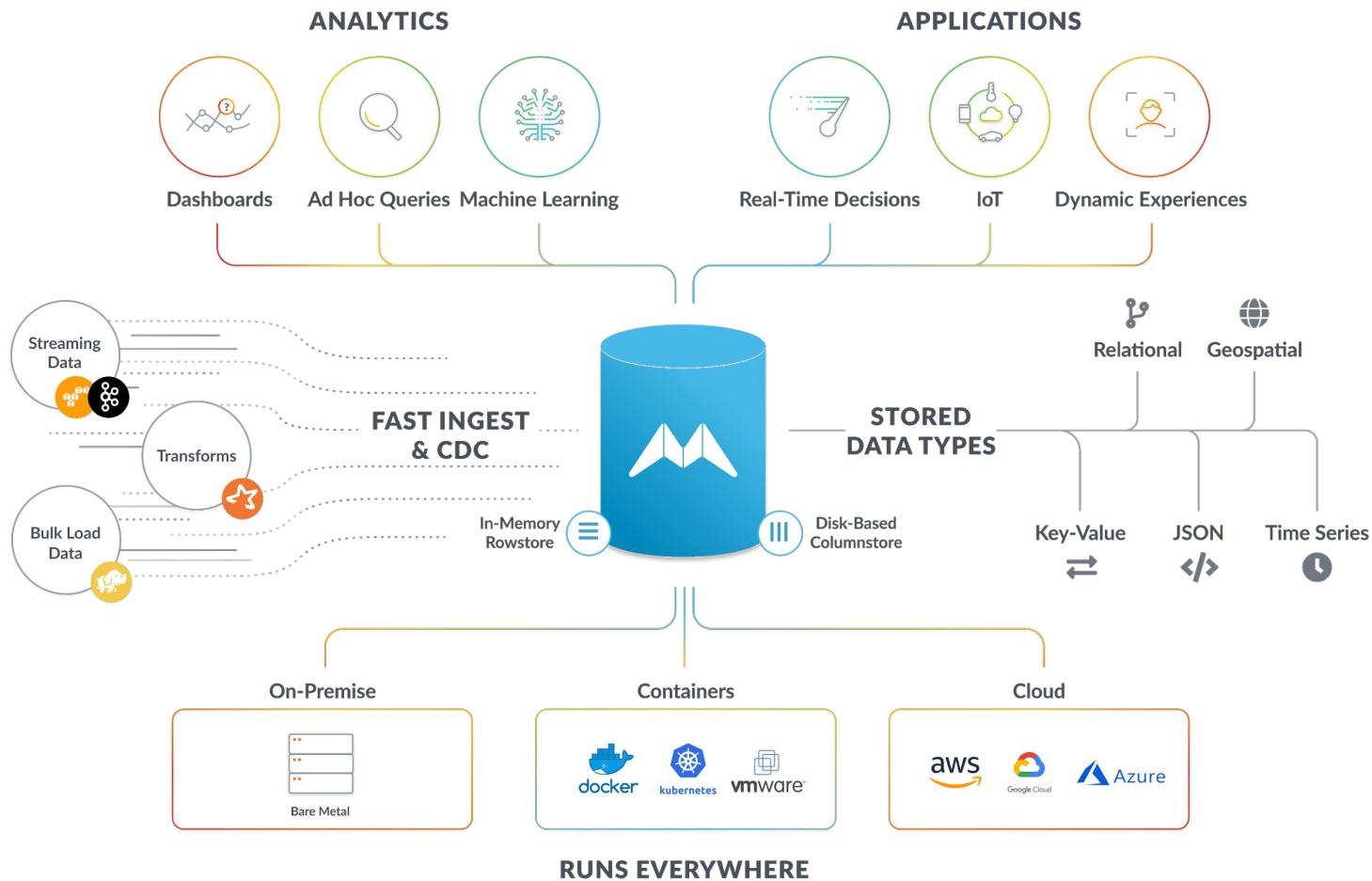**Eric Hanson**
Principal Product Manager
hanson@memsql.com

- MemSQL Overview

- The Promise and Challenges Delivering Real-Time ML/AI Applications

- How MemSQL Operationalizes ML Applications

- Q&A

MEMSQL

MemSQL Overview

MEMSQL

MemSQL: The No-Limits Database

The cloud-native operational database built for speed and scale

Accelerate time to insight with ultra fast ingest and high performance query

SPEED

SCALE

Build on a cloud-native data platform designed for today's most demanding applications and analytical systems

SQL

Reduce cost and complexity by leveraging a modern yet familiar and easy to integrate traditional RDBMS running standard SQL

ANALYTICS

Dashboards Ad Hoc Queries Machine Learning

APPLICATIONS

Real-Time Decisions IoT Dynamic Experiences

Streaming Data

Transforms

Bulk Load Data

FAST INGEST & CDC

In-Memory Rowstore

Disk-Based Columnstore

STORED DATA TYPES

Relational Geospatial

Key-Value JSON Time Series

On-Premise

Bare Metal

Containers

docker kubernetes vmware
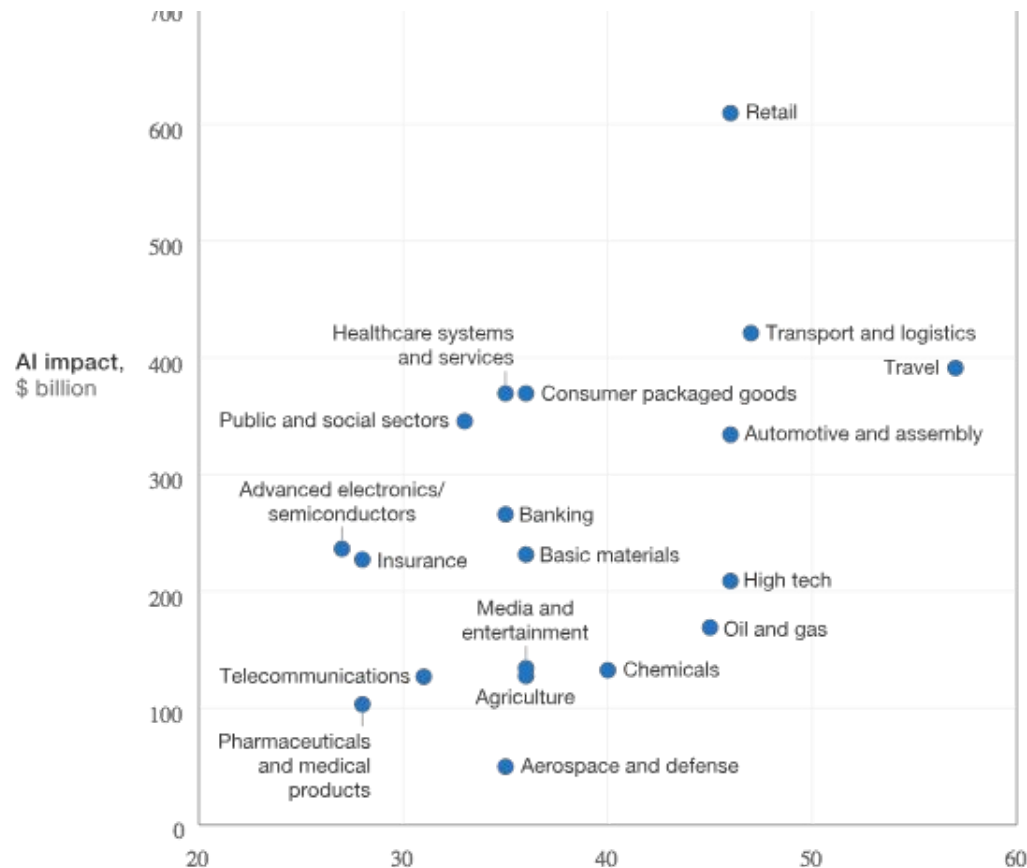
Cloud

aws Google Cloud Azure

RUNS EVERYWHERE

5

ML/AI & MemSQL

MEMSQL

The Promise of ML/AI

Artificial intelligence (AI) has the potential to create tremendous value across sectors

AI impact, $ billion

Share of AI impact in total impact derived from analytics, %

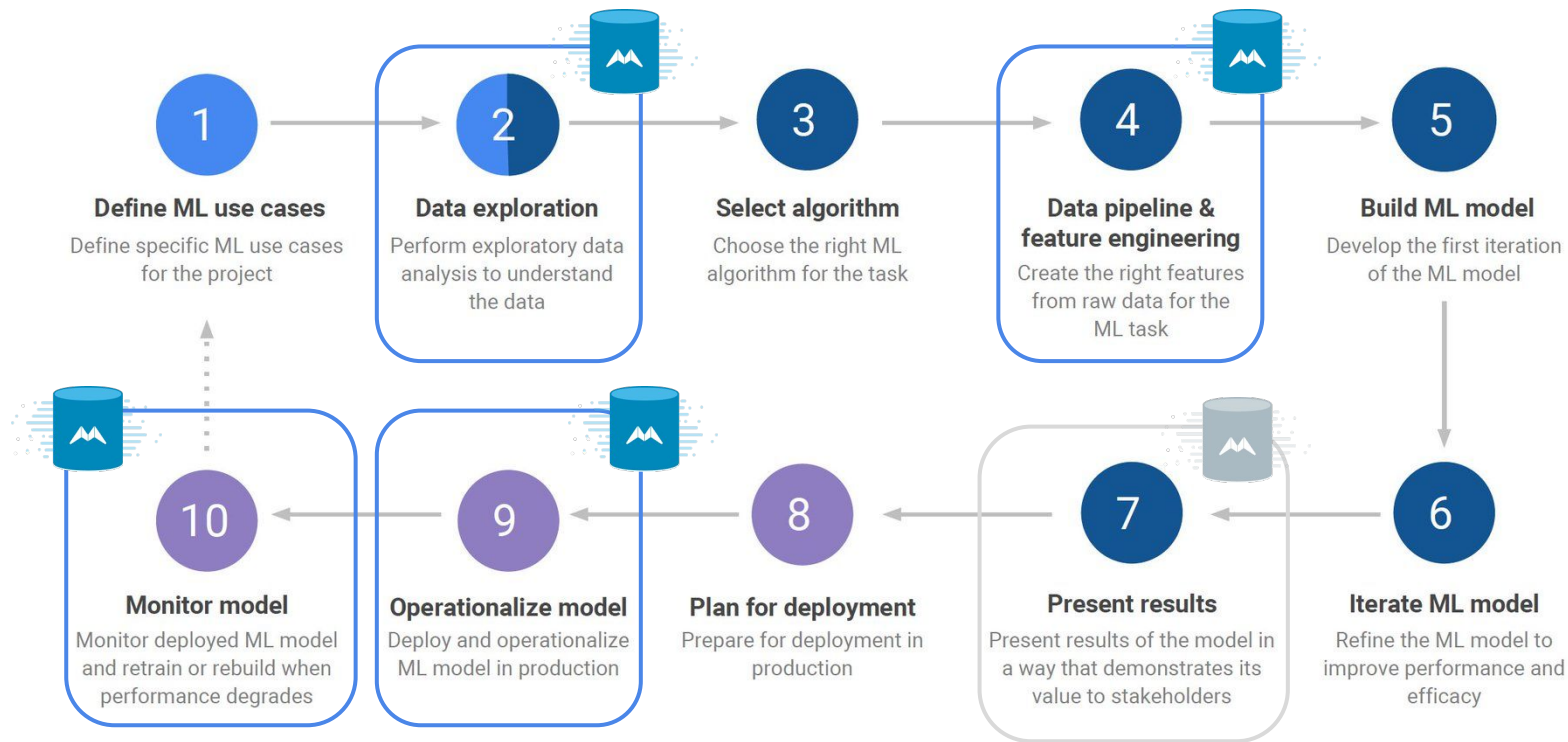Source: McKinsey Global Institute analysis

MEM**SQL**

Challenges Delivering a Real-Time ML Application

- Data exploration & preparation
- Algorithm selection, model training & refinement
- Assembling data for scoring
- Scoring
- Model monitoring, refinement, & retraining

MEMSQL

Challenges Delivering a Real-Time ML Application

- Data exploration & preparation
- Algorithm selection, model training & refinement
- Assembling data for scoring
- Scoring
- Model monitoring, refinement, & retraining

MEMSQL

MemSQL Drives Improvement Across the ML Lifecycle

1

Define ML use cases

Define specific ML use cases for the project

2

Data exploration

Perform exploratory data analysis to understand the data

3

Select algorithm

Choose the right ML algorithm for the task

4

Data pipeline & feature engineering

Create the right features from raw data for the ML task

5

Build ML model

Develop the first iteration of the ML model

10

Monitor model

Monitor deployed ML model and retrain or rebuild when performance degrades

9

Operationalize model

Deploy and operationalize ML model in production

8

Plan for deployment

Prepare for deployment in production

7

Present results

Present results of the model in a way that demonstrates its value to stakeholders

6

Iterate ML model

Refine the ML model to improve performance and efficacy

MEMSQL

Operationalize ML/AI with Speed, Scalability, SQL, & Programmability

- Score data during streaming load with pipelines transforms
- Score fast in-database with
  - vector functions
  - compiled expressions and functions
  - scale-out & parallelism
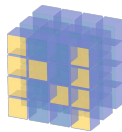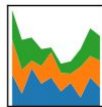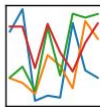- Assemble up-to-date feature records to score with an external app

MEMSQL

How MemSQL Operationalizes ML Applications

MEMSQL

AI/ML Tool Integration with MemSQL

- MemSQL as a source for training data

- Support for virtually all tools that can connect to MySQL, e.g.

- named connector

Use Loading Tools for Bulk or Real-Time Ingest

Real-Time Data

Kafka

CDC
ATTUNITY
REPLICATE

Transforms

Spark

ETL/ELT
Informatica, Talend

Data Lakes

HDFS

AWS S3

Azure Blob

MEMSQL

Use MemSQL Pipelines for Scoring on Load

- MemSQL Pipelines
  - Automatic streaming loader

- Transform types
  - Python
  - Executable

- Use
  - Compute new "score" column from other columns during load

- Benefits
  - Use with any external scoring code or libraries

S3

Kafka

HDFS

Filesystem
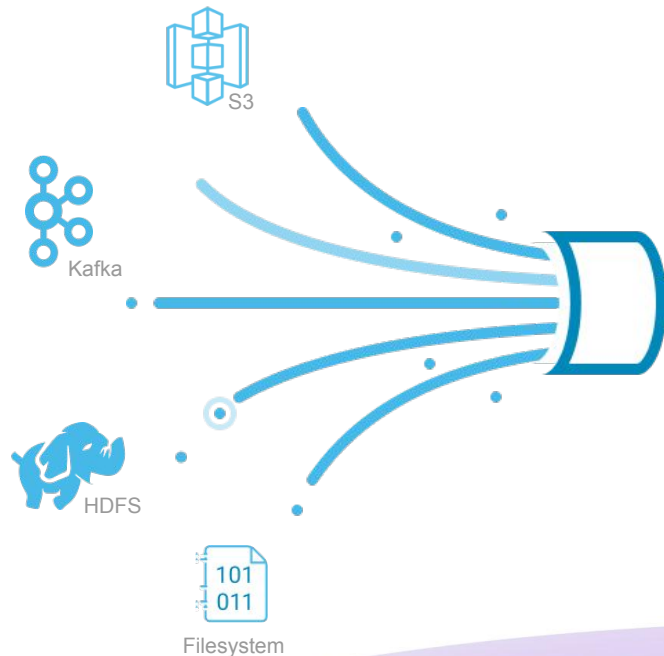
MEMSQL

Pipeline Transform Example

```
CREATE PIPELINE mypipeline AS
LOAD DATA KAFKA '192.168.1.100:9092/my-topic'
WITH TRANSFORM
('http://www.memsql.com/my-transform.tar.gz',
'my-executable.py', '')
INTO TABLE t
```

More at https://docs.memsql.com/memsql-pipelines/v6.8/transforms/

MEM**SQL**

Implementing Scoring With Extensibility

- Support for:
  - User-defined functions (UDFs)
  - Stored procedures (SPs)
  - User-defined aggregates (UDAFs)
  - Arrays
  - Records
  - Control structures

- Implement PMML or other models for in-DB scoring

- Benefits
  - Combine with other SQL (joins, filters, etc.)
  - Use with SQL-compatible tools
  - Compiled code and scale-out performance
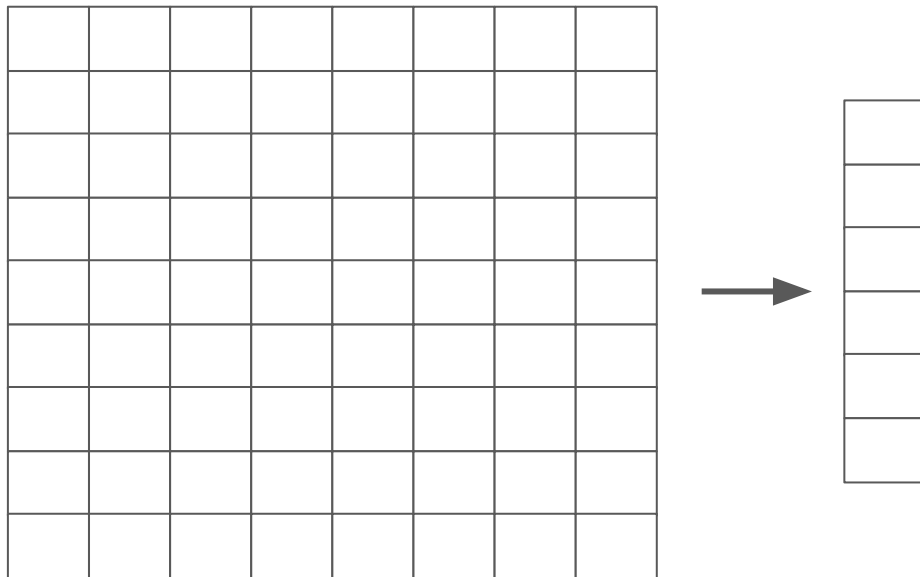
MEMSQL

Example: Regression Line

- $f(x) = 0.73\,x + 2.95$

- ```
  create or replace function f(x double)
  returns double as

  begin
     return 0.73 * x + 2.95;
  end;
  ```

- ```
  select a, b, f(x) from t;
  ```

MEMSQL

Scoring Using Vector Embeddings

- Produced by deep neural net hidden layer
- Embedding maps high-dimensional space to low e.g.
  - High: image (256 X 256 matrix)
  - Low: 1000-element vector
- Applications
  - face matching
  - product photo matching
  - document similarity
  - and on and on!

MEMSQL

Real-Time Image Recognition Workflow

- Train the model with Spark, TensorFlow, and Gluon

- Use the Model to extract feature vectors (embeddings) from images
  - Model + Image => FV

- You can store every feature vector in a MemSQL table

```
CREATE TABLE features (
   id bigint(11) NOT NULL,
   feature binary(4096),
   KEY id (id)USING CLUSTERED
COLUMNSTORE
)
```

MEMSQL

MemSQL Functions for Vector Similarity Matching

- DOT_PRODUCT(vector, vector)

- EUCLIDEAN_DISTANCE(vector, vector)

(fast SIMD implementations)

Helper functions:

- JSON_ARRAY_PACK('[float [, ...]]')
- VECTOR_SUB(vector, vector)

MEMSQL

Working with Feature Vectors

For every image, we store an `ID` and a normalized `FEATURE` vector in a MemSQL table called features.

```
ID   | Feature Vector
-----+-----------------
int  | 4KB binary
```

To find similar images, we use this SQL query

```
SELECT
    id, DOT_PRODUCT(feature, <input>) as score
FROM
    features
WHERE
    DOT_PRODUCT(feature, <input>) > 0.9
ORDER BY score DESC
```
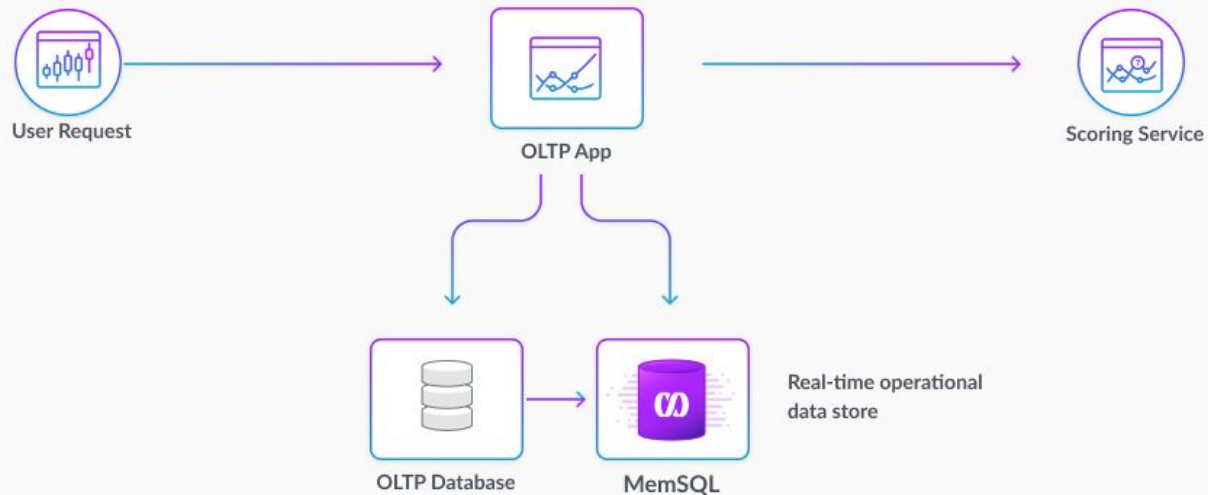
MEMSQL

Face Matching with MemSQL

- Face matching, pre-trained networks

  - VGG-Face

  - Facenet

- Match millions of images to a query image in a fraction of a second

- Apps can use this combined with SQL join/filter

- New images can be continuously added and tagged with embedding

MEMSQL

Real-Time Feature Retrieval For Scoring in OLTP Apps

- Example: credit card fraud detection
- 1 second budget from card swipe to approval
- Do fraud detection within this 1 second
- 50 msec budget
- 70-value feature record to score

MEMSQL

Credit OLTP + Scoring App Architecture

User Request → OLTP App → Scoring Service

OLTP App → OLTP Database → MemSQL

Real-time operational data store

MEMSQL

Creating the feature record in real time

Example features:

has transacted with merchant
days since last transaction with merchant
max amount transacted with merchant in last 180 days
min   "   "
180 days total merchant transaction amount
card present transaction count last 1, 7, 30 days
online transaction count last 1, 7, 30 days

...
(70 or more features!)

Options

1. **Old way:** nightly batch job to accumulate feature record for each customer in traditional "operational data store"; look it up for scoring

2. **New way:** run up to 70 queries concurrently (one per feature or few features) on latest data to get features in < 50 msec *total* in real time operational data store

MemSQL enables #2!

Benefits:
- enables new features like "transactions in last 2 hours"
- fresh features & new kinds of features enable catching more fraud cases

MEMSQL

Additional Resources

MemSQL Vector Functions
https://msql.co/vector-functions

Google Crash Course on ML: Embeddings
https://msql.co/google-embeddings

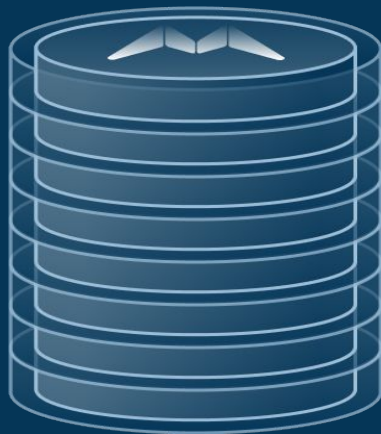**MemSQL Face Matching Blog**
https://msql.co/Face-matching

MEMSQL

Thank You! Questions?

Try at memsql.com/download

- No time limit
- Deploy to production
- Full featured
- Up to 4 Nodes, unlimited disk
- Get support at forum.memsql.com

Learn more at memsql.com/product

Questions? Email us at **team@memsql.com**

MEMSQL