



The Database of Now™

Dusty Everson - Account Executive

Mike Czabator - Director of Technical Sales

Mark Lochbihler - Global Director, Partner Engineering

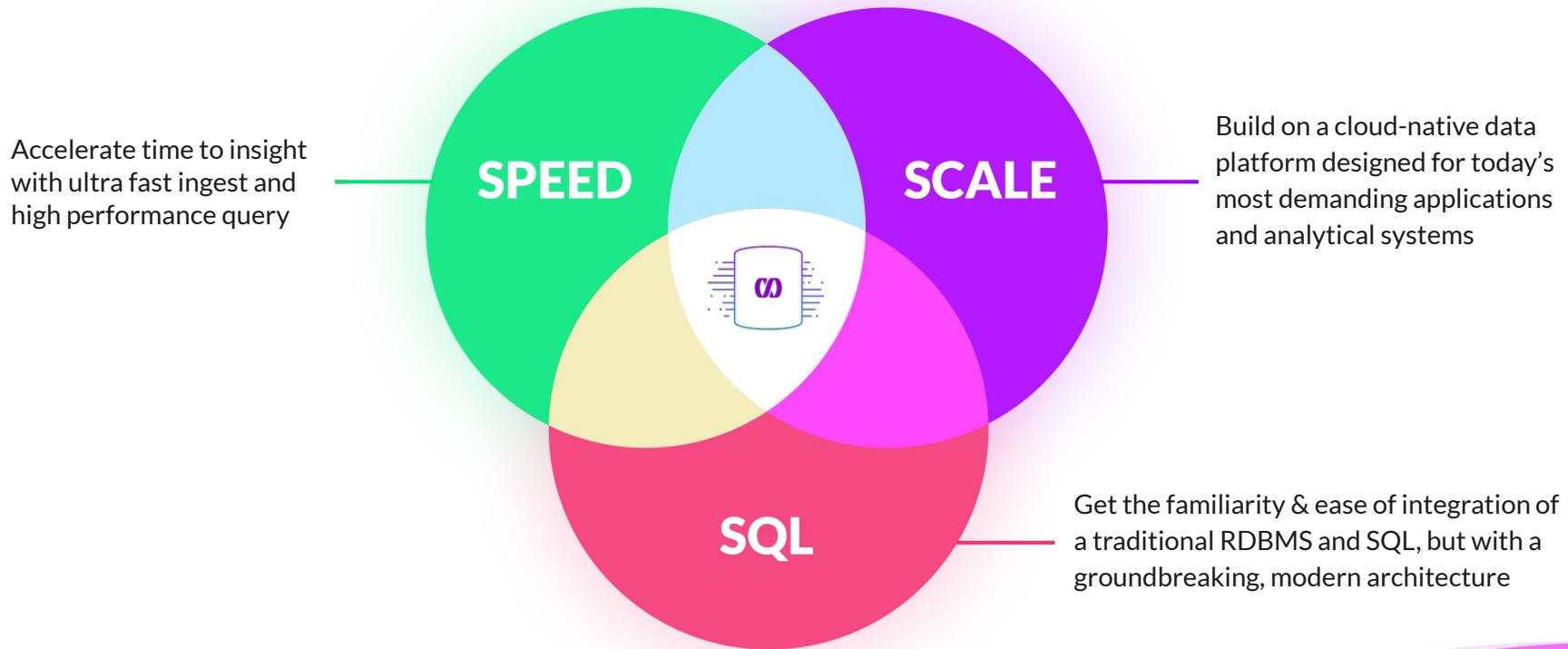
Sarung Tripathi - Solution Engineering

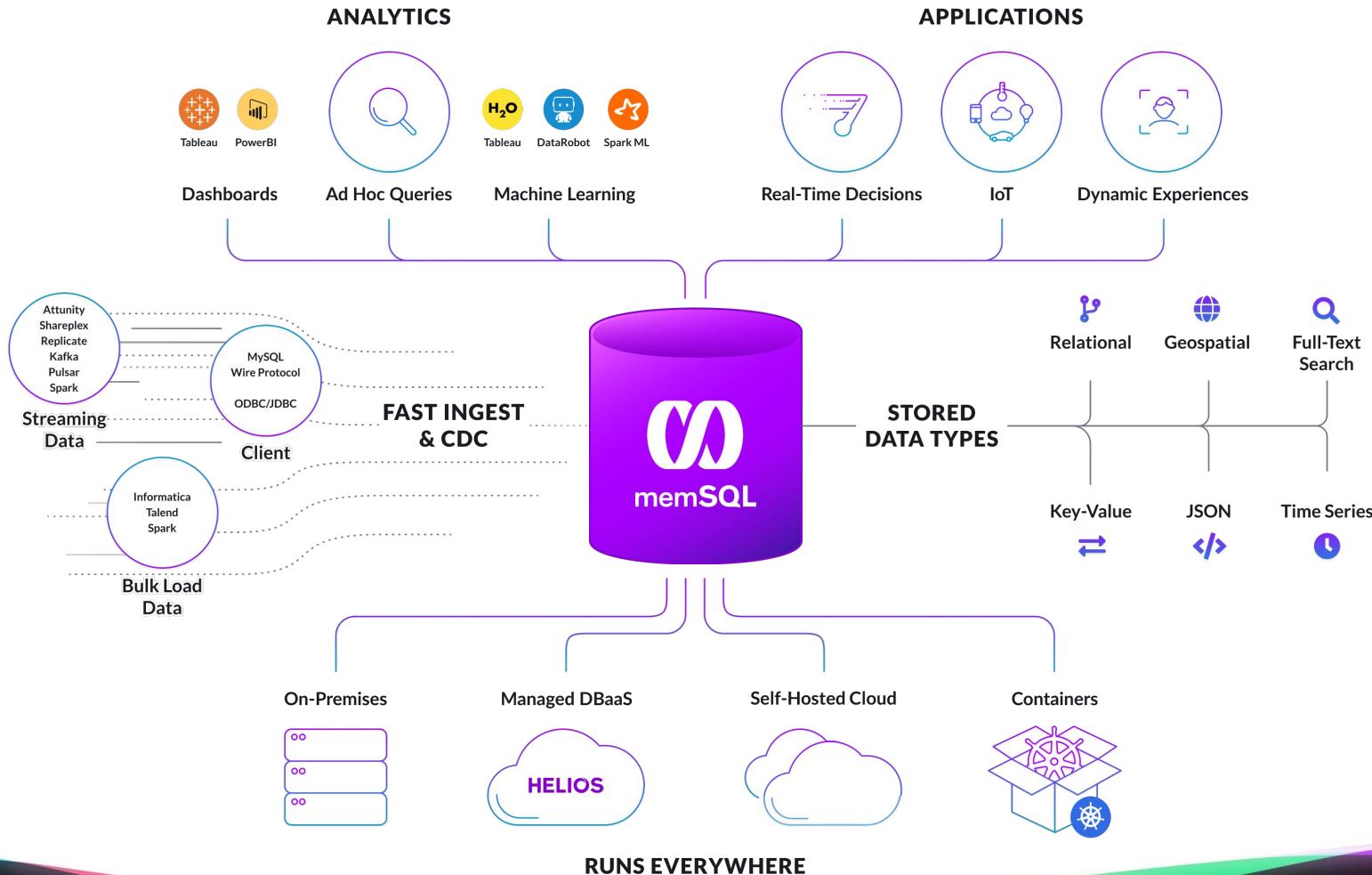
Roxanna Pourzand - Product Manager

MemSQL Introduction

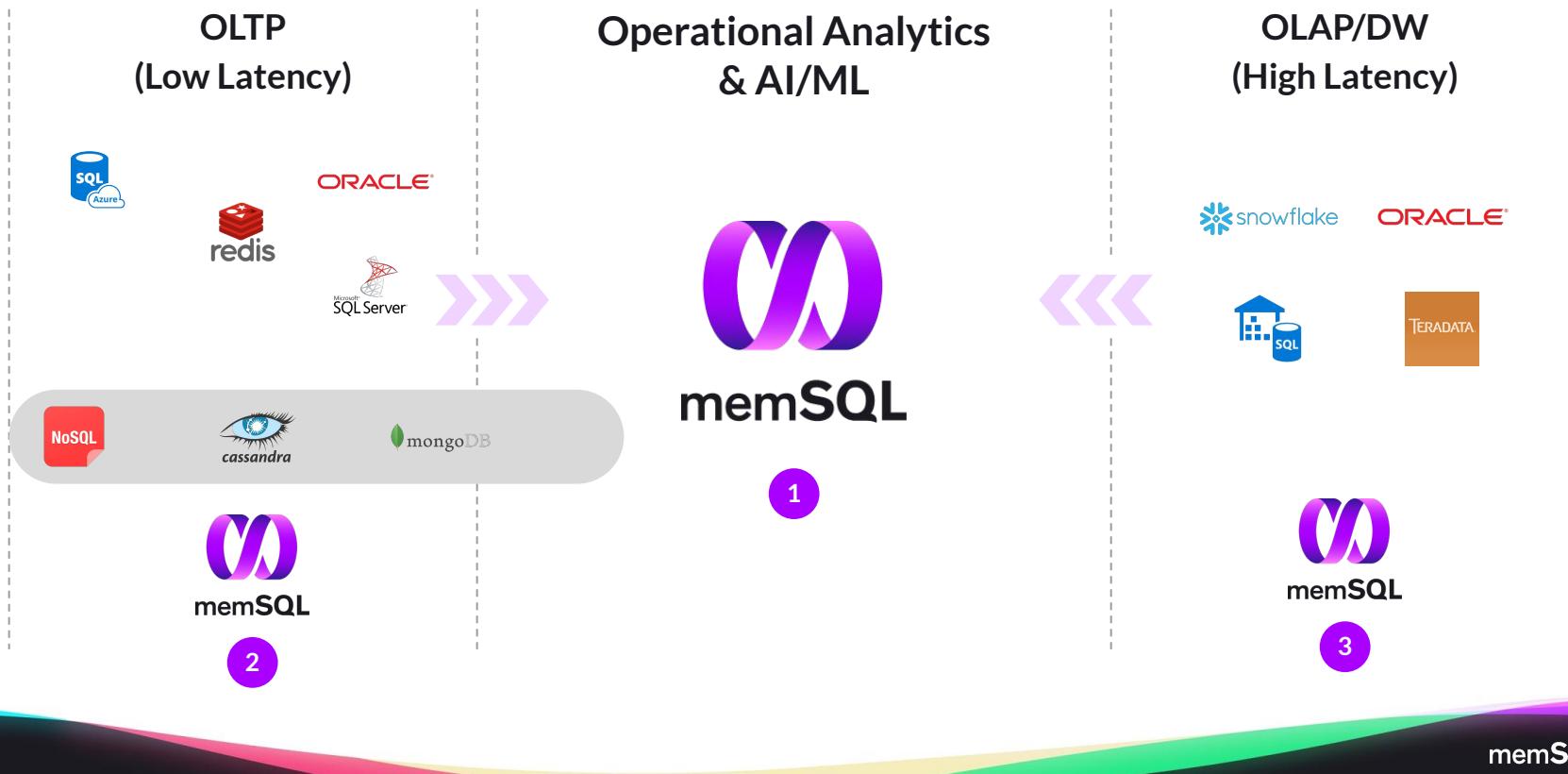
MemSQL: The Database of Now

The cloud-native operational database built for speed and scale





MemSQL - Data Landscape

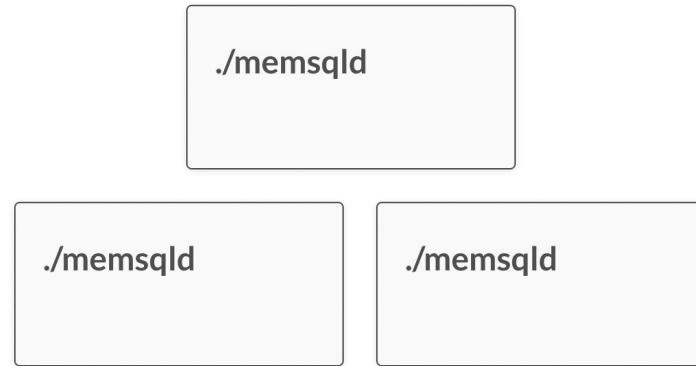


Architecture

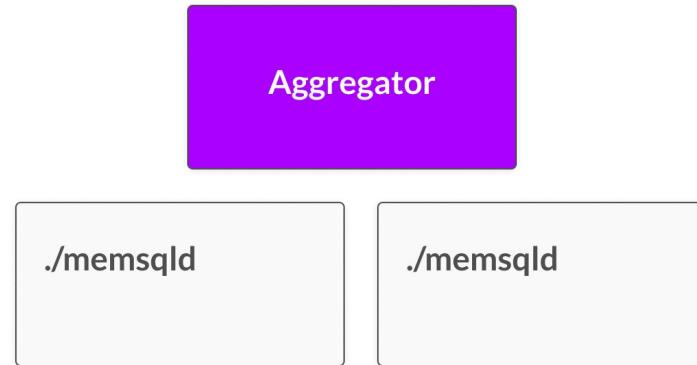
MemSQL is a Database, a Linux Daemon

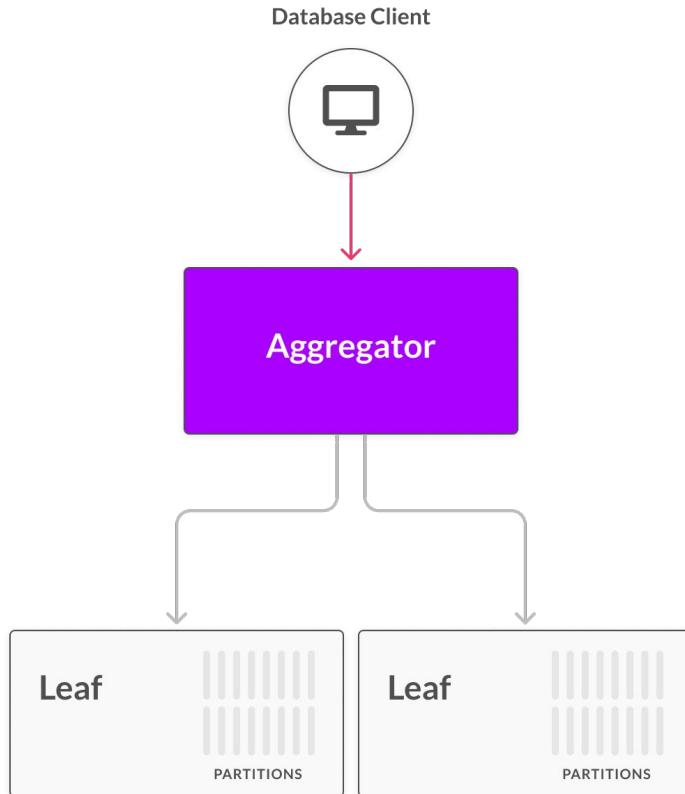
```
./memsqld
```

MemSQL is a Distributed System



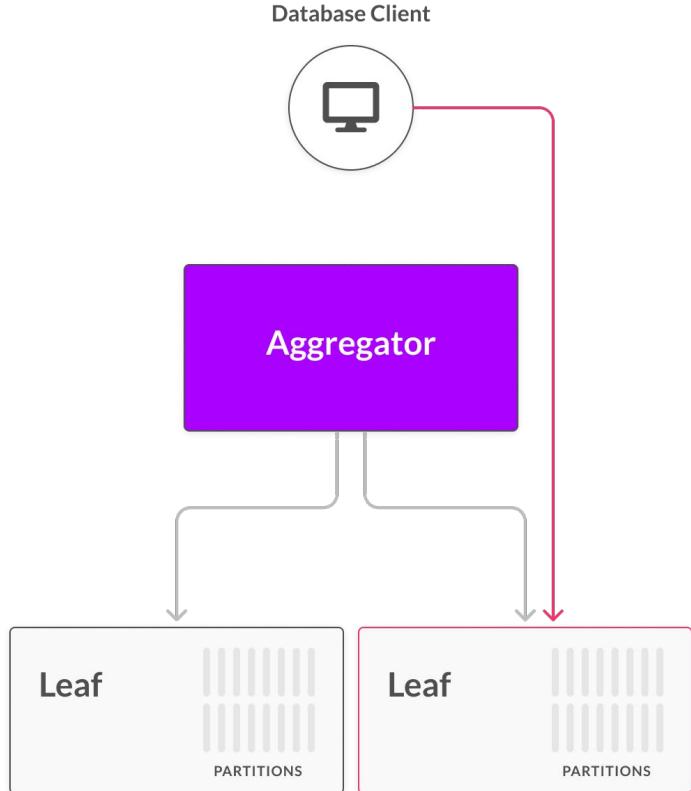
Aggregators Aggregate





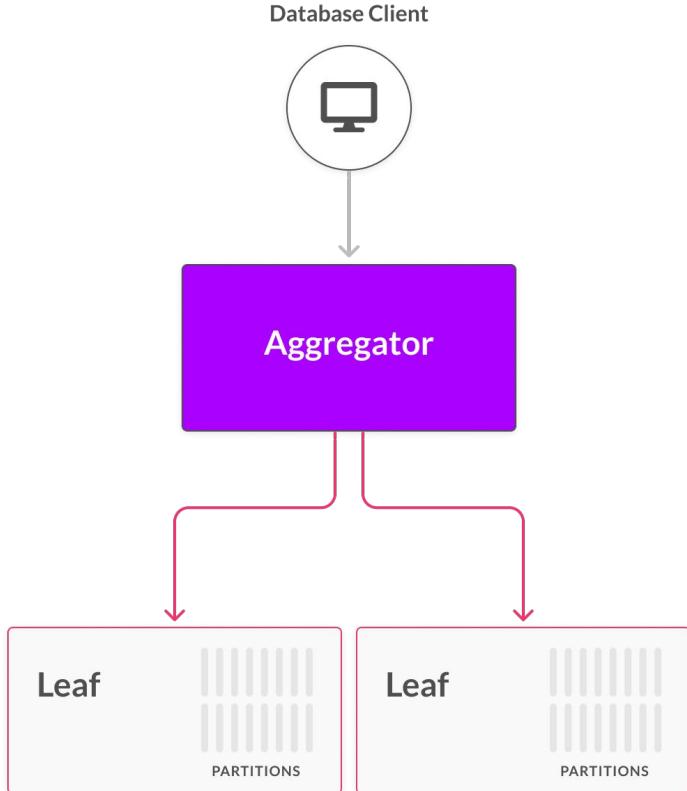
Aggregators interact with clients and leverage leaf nodes

```
1 aggregator-1> create database foo;
2 Query OK, 1 row affected (5.48 sec)
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
```



Leaves store a partition per core on the machine (by default)

```
1 leaf-2> show databases;
2 +-----+
3 | Database      |
4 +-----+
5 | cluster       |
6 | foo          |
7 | foo_1        |
8 | foo_3        |
9 | foo_5        |
10 | foo_7       |
11 | foo_9        |
12 | foo_11      |
13 | information_schema |
14 | memsql       |
15 +-----+
16 10 rows in set (0.01 sec)
17
18
19
20
21
22
```

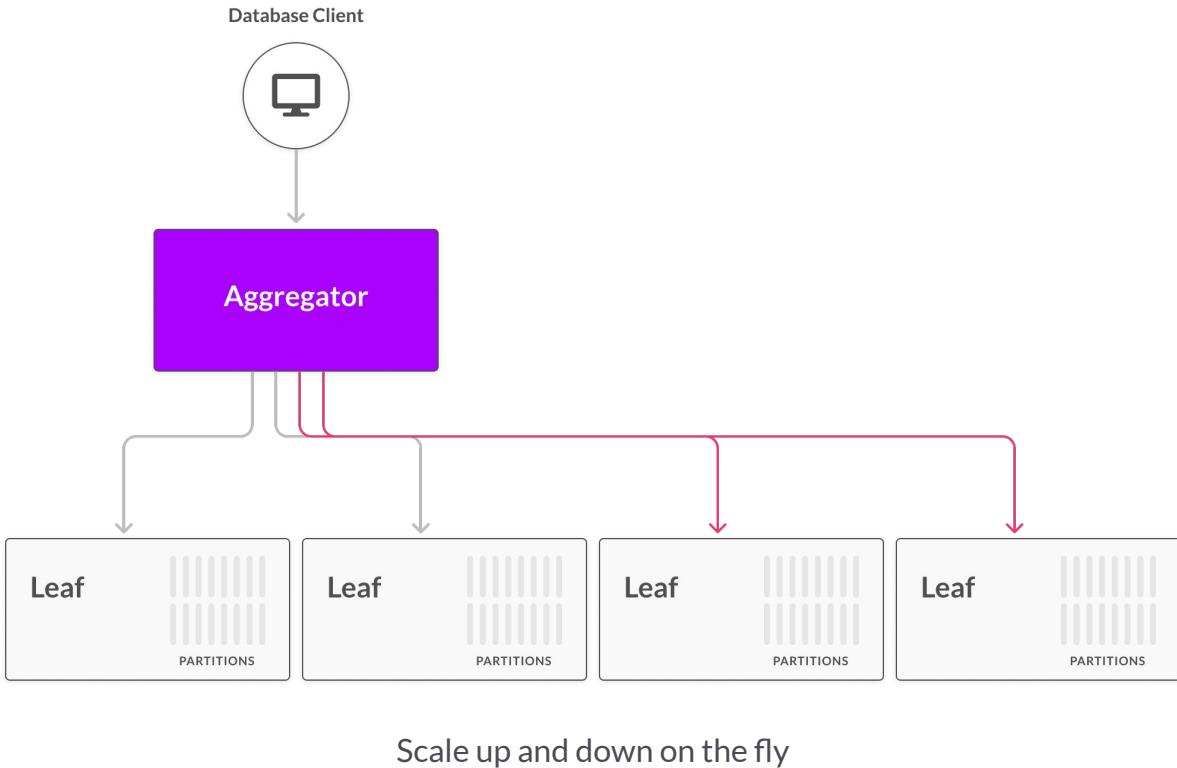


Massively parallel processing (MPP) across all the leaf nodes for query execution

```
1 aggregator-1> SELECT avg(price) FROM orders;  
2 ...  
4
```

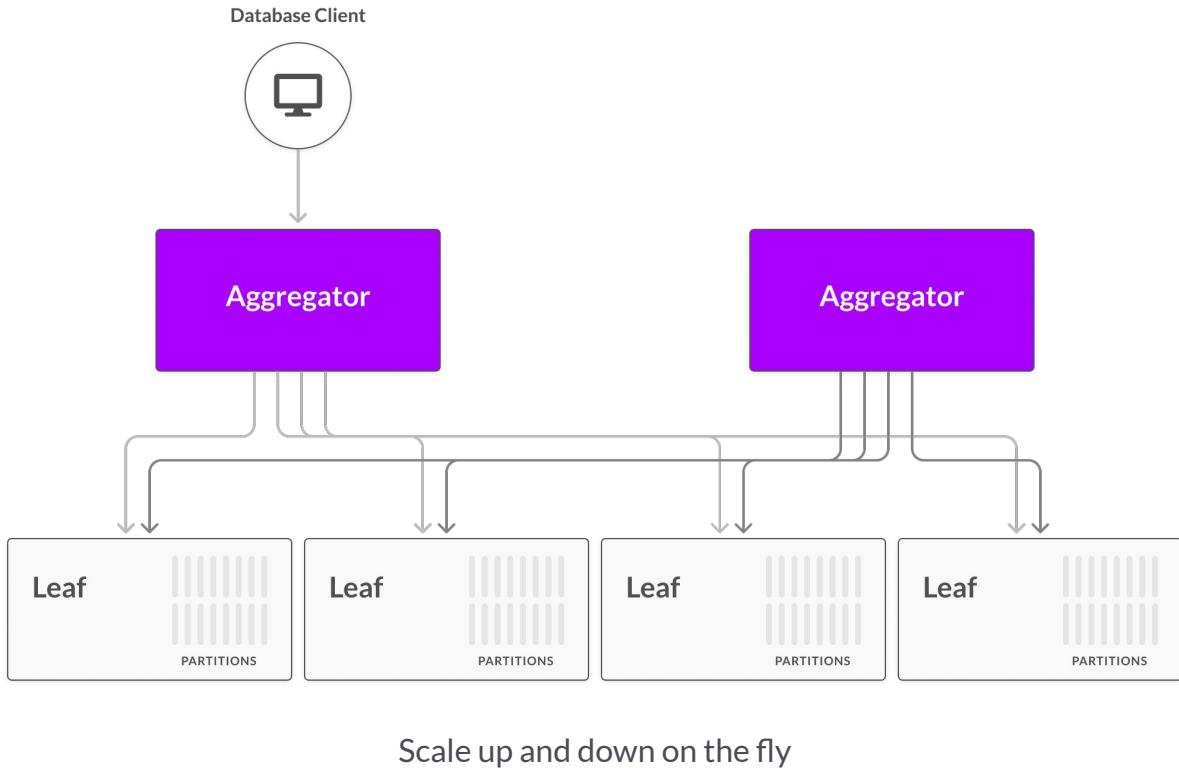
```
1 leaf-1> using memsql_demo_9 SELECT count(1), sum(price) FROM orders;  
2 ...  
4
```

```
1 leaf-2> using memsql_demo_17 SELECT count(1), sum(price) FROM orders;  
2 ...  
4
```



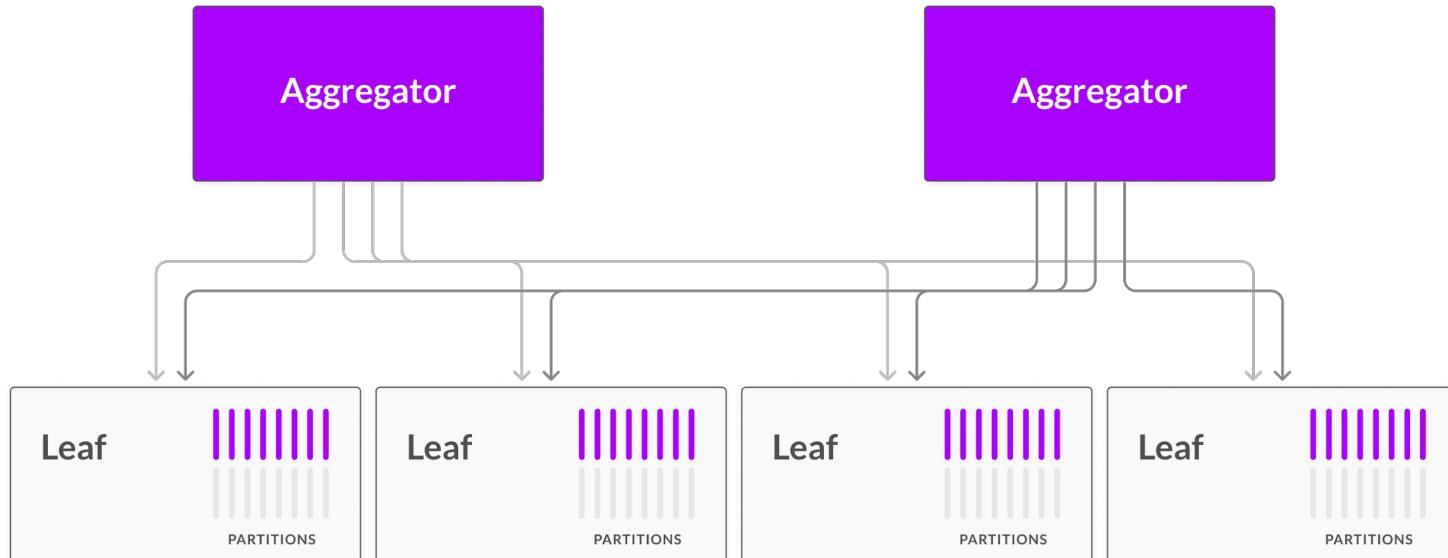
```
1 aggregator-1> ADD LEAF leaf-3...  
2 aggregator-1> REBALANCE PARTITIONS;  
3  
4
```

```
1 aggregator-1> ADD LEAF leaf-4...  
2 aggregator-1> REBALANCE PARTITIONS;  
3  
4
```



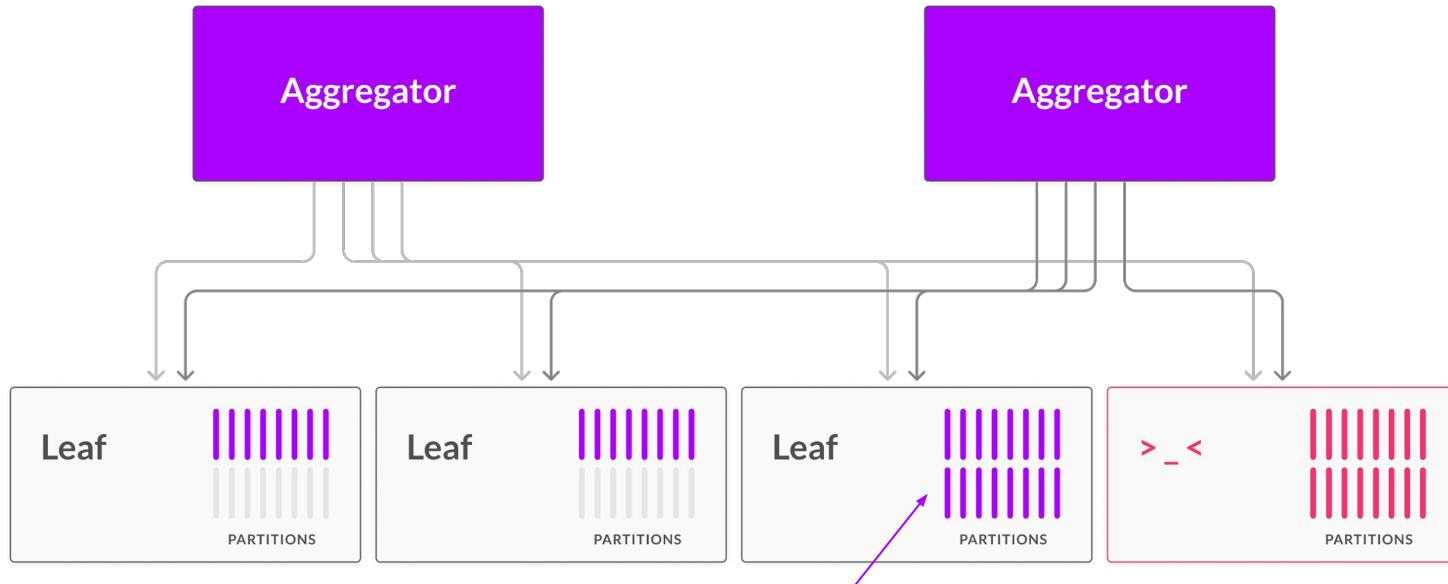
```
1 [memsql.cnf]  
2 master-agg=agg-1  
3  
4
```

Data is Copied for High Availability

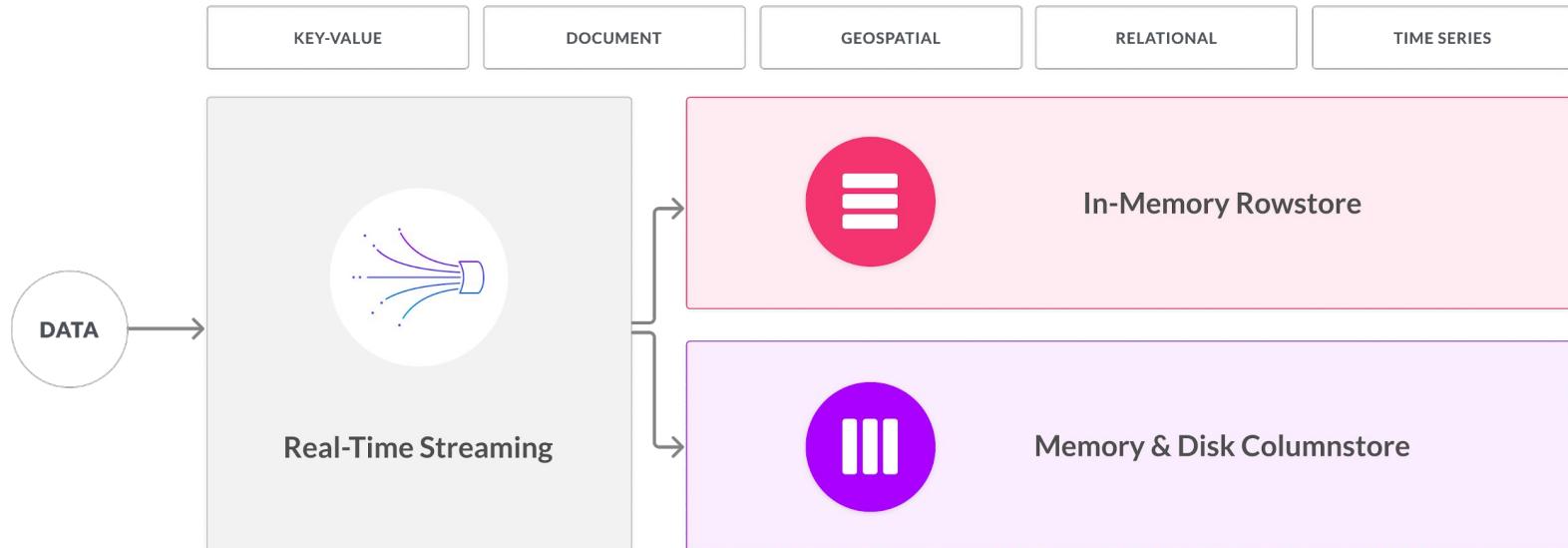


Each leaf has a pair which replicates its data

Stays Online in Case of Node Failure



Two Storage Engines

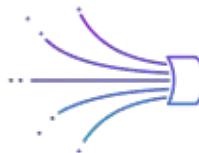


MemSQL capabilities

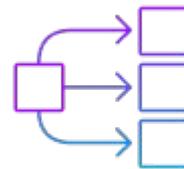
What Makes MemSQL Unique?



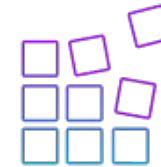
Distributed Cloud
Native Architecture



Streaming Ingest



Lock-free



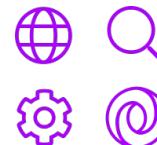
Query Compilation



ANSI SQL



Compatibility

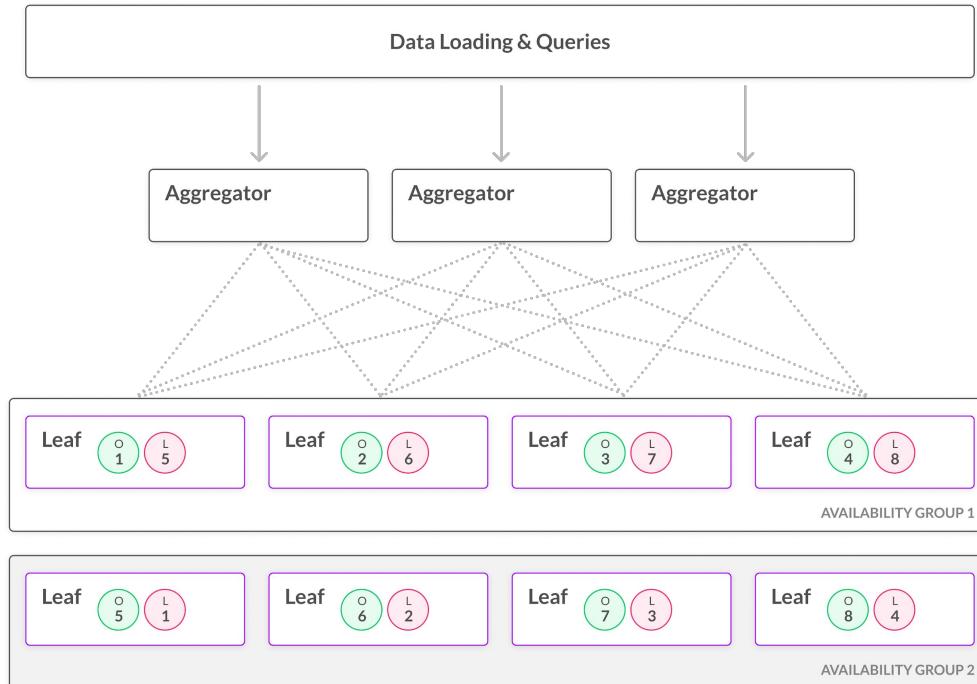


Data Flexibility

MemSQL Data Architecture

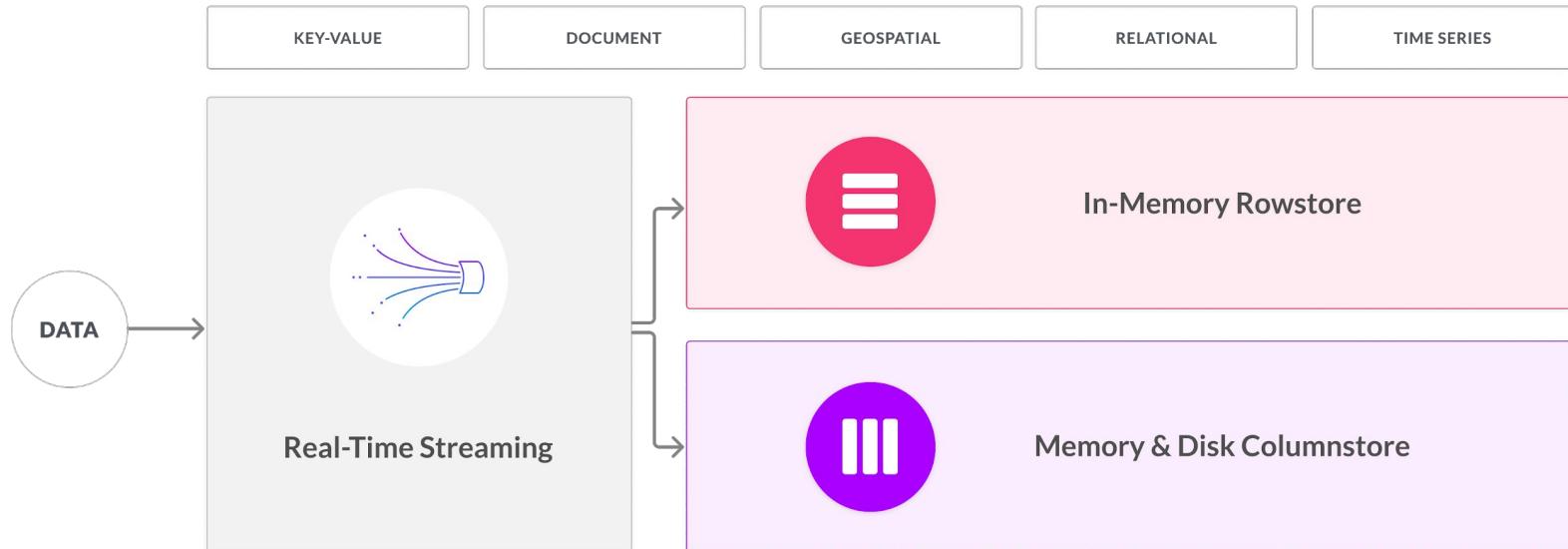
- **Aggregator Linear Scalability**
 - High concurrent OLTP and OLAP
 - High throughput data ingestion
- **Leaf Linear Scalability**
 - High data volume
 - High throughput ELT

Data is replicated between
Leaf Nodes for HA



MemSQL table types

Two Storage Engines



Row and Column Orientations

In Memory Rowstore

- All data in memory, but also persisted to disk
- Scan ~40M rows / second / core
- Streaming or Batch Insert
- Select / Update Individual Rows Efficiently
- Row lock on update and delete
- Sparse Compression



Rowstore

PRODID	COLOR	PRICE
1	RED	10
2	RED	20
3	BLACK	20
4	WHITE	20

Memory Optimized Columnstore

- Data Stored on SSDs
- Scan >200M+ rows / second / core
- Streaming or Batch Insert
- ~80% Compression
- Row lock on update and delete



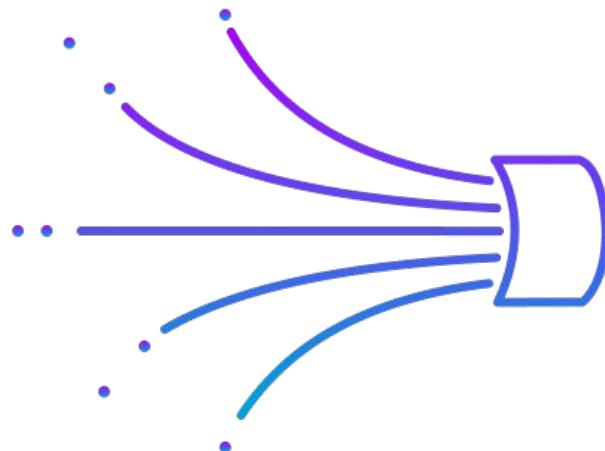
Columnstore

PRODID	COLOR	PRICE
1	RED x 2	10
2	BLACK	20 x 3
3	WHITE	
4		

MemSQL Pipelines

Pipelines

- Streaming ingestion
- Fast parallel bulk loading
- Transactional Consistency
- Exactly-Once Semantics
- Ingest Data into a Stored Procedure
- Native integrations with Kafka, AWS S3, Azure Blob, HDFS, NFS, etc.

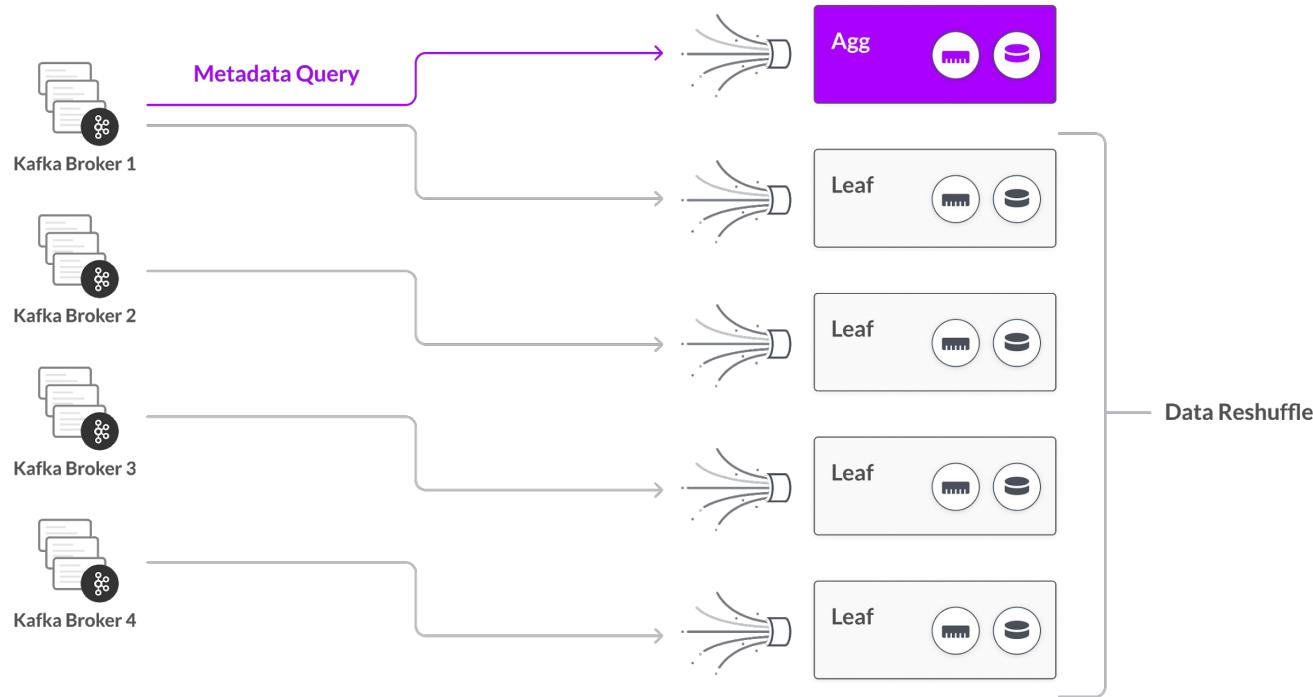


MemSQL Pipelines Sequence

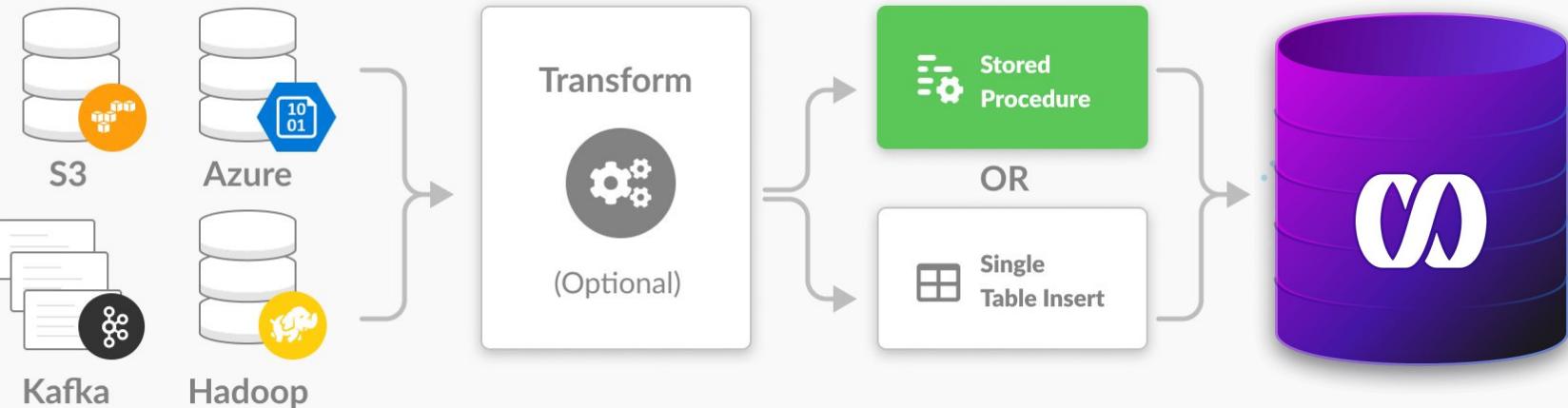


- 1 MemSQL polls for changes from a data source system.
- 2 MemSQL pulls the data into it's memory space (no commit) where a transform can be applied.
- 3 The data is committed in a transaction (and in parallel)

MemSQL Pipelines Sequence



Pipeline to Procedure



Geospatial Functionality

Accelerating the Value of Geospatial Data

**Store, query, and index geographic data types including polygons and points
to support area, distance, location analytics in real-time**

MemSQL makes geospatial data a first-class citizen: just as easy to use, at scale, at great speed and high throughput, as any other kind of relational data. MemSQL geospatial is a partial implementation of the OpenGIS standard for spatial processing.

- **Data types:** MemSQL supports Points, Linestrings, Polygons and Multipolygons natively using the industry-standard Well-Known Text (WKT) format;
- **Topological filter functions:** Contains, Intersects, Approximation, and Within Distance;
- **Measurement functions:** Length, Area, Distance
- **Indexing:** MemSQL leverages its fast, massively parallel, lock-free indexing for geospatial objects;
- **Spatial Joins:** MemSQL tables can be joined by their spatial relationships.

MPSQL

Procedural SQL Language Extensions (MPSQL)

- **Stored Procedures (SPs)**

Stored procedures can accept input parameters, query tables using SQL statements, call UDFs, define custom logic using control flow statements and variable assignment, and optionally return a value. Stored procedures can also be called across databases.

- **User-Defined Scalar-Valued Functions (UDFs)**

UDFs can accept input parameters, call other UDFs, define custom logic using control flow statements and variable assignment, and return a value.

- **User-Defined Table-Valued Functions (TVFs)**

TVFs can accept input parameters, execute a single SELECT statement, and return the result as a table-typed value.

- **User-Defined Aggregate Functions (UDAFs)**

UDAFs support creation of custom aggregation logic, beyond the built-in aggregate functions supplied by MemSQL.

Client Connectivity

Easy to Manage and Tune with MemSQL Studio

- Execute DML and DDL commands and observe results
- Monitor Performance and Capacity
- API-based Deployment and Configuration
- Full Administration and Scaling
- Query Profiler
- Visual Query Plans



Compatibility with what is familiar



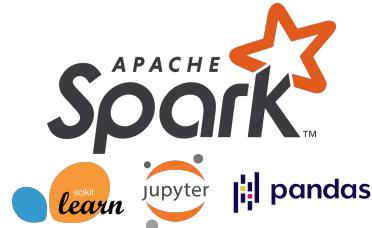
- Connectors for all major BI and analytics platforms
- Run queries using [MemSQL Studio](#) or [MySQL Workbench](#)
- [memSQL Python Connector](#)
- [memSQL Spark Connector](#)



MemSQL: Spark Connector 3.0

—
February 2020

Spark and MemSQL: Better Together



MemSQL Spark Connector:
Full SQL Pushdown



- Leverage existing models in-place with the MemSQL Spark Connector (true Spark DataSource)
- Push down models built in Python, Scala or Java and leverage the speed and scale of memSQL (20X performance improvement on gradient boosting)
- Applicable Spark dataframe operations get translated to true SQL and executed, with high performance and excellent concurrency

Together: The fastest data platform for integration and analytics

Features

- Compatible with Spark 2.3 and 2.4
- True Spark data source compatible with JDBC Reader/Writer API
- Implemented as a native SparkSQL plugin
- Robust SQL Pushdown support (including partial pushdown)
- Leverages MemSQL LOAD DATA to accelerate ingest from Spark via compression
- Takes advantage of all the latest and greatest features in MemSQL 7.0+

Requirements

You need the following to get started:

- MemSQL environment running version 6.7 or greater (6.8+ recommended)
- Apache Spark Environment 2.3+
- MariaDB JDBC driver 2.0 and above (this will get pulled in automatically as a dependency)

Downloading

- You can find the connector and code at: <https://github.com/memsql/memsql-spark-connector>
- You can also add this to your library via [spark-shell](#) , using [Maven](#), or SBT
- Instructions for each can be found on our [documentation](#)

Examples

—
February 2020

Configuring the Connector: Example

See all configuration options [here](#)

```
spark.conf.set("spark.datasource.memsql.ddlEndpoint",
"memsql-master.cluster.internal")
spark.conf.set("spark.datasource.memsql.dmlEndpoints",
"memsql-master.cluster.internal,memsql-child-1.cluster.internal:3307")
spark.conf.set("spark.datasource.memsql.user", "admin")
spark.conf.set("spark.datasource.memsql.password", "s3cur3-pa$$word")
```

Example: Load + Read Data

Scala - reads data from table 'foo' in MemSQL

```
val df = spark.read  
  .format("memsql")  
  .option("ddlEndpoint", "memsql-master.cluster.internal")  
  .option("user", "admin")  
  .load("foo")
```

Spark SQL - registers table 'bar' from MemSQL

```
CREATE TABLE bar USING memsql OPTIONS ('ddlEndpoint'='memsql-master.cluster.internal','dbtable'='foo.bar')
```

Example: Write Data

Scala - Write data from dataframe 'df' in Spark to table 'foo.bar'

```
df.write  
  .format("memsql")  
  .option(" loadDataCompression", "LZ4")  
  .mode(SaveMode.Overwrite)  
  .save("foo.bar") // in format: database.table
```

Datasource API

All operations supported by the `dataSource API` are supported by the connector

[Dataframe Operations - Spark Documentation](#)

Documentation Page

[MemSQL Spark Connector Reference: MTA Hackathon](#)

MemSQL Deployment Options

MemSQL Deployment Options on MS Azure

The screenshot shows a web browser displaying the MemSQL Documentation website at docs.memsql.com/v7.1/guides/deploy-memsql/self-managed/. The page title is "Select a deployment type - MemSQL". The main content area is titled "Select a deployment type" and lists three options: "Bare Metal or VMs", "Kubernetes", and "Cluster in a Box". Each option has a corresponding icon and a "Start Guide" button.

memSQL Docs

Product Solutions Docs Blog Resources Contact Us

Sign In Try Free →

Search for features, version numbers, keywords...

Introduction v7.1

MemSQL Helios

Guides

- Overview
- Deploy MemSQL
- Data Ingest
- Development
- Cluster Management
- Client and Application
- Security

Concepts

Tools

Reference

Third-Party Integrations

Release Notes

Deploy MemSQL / Deployment: Self-Managed / Type: Select one

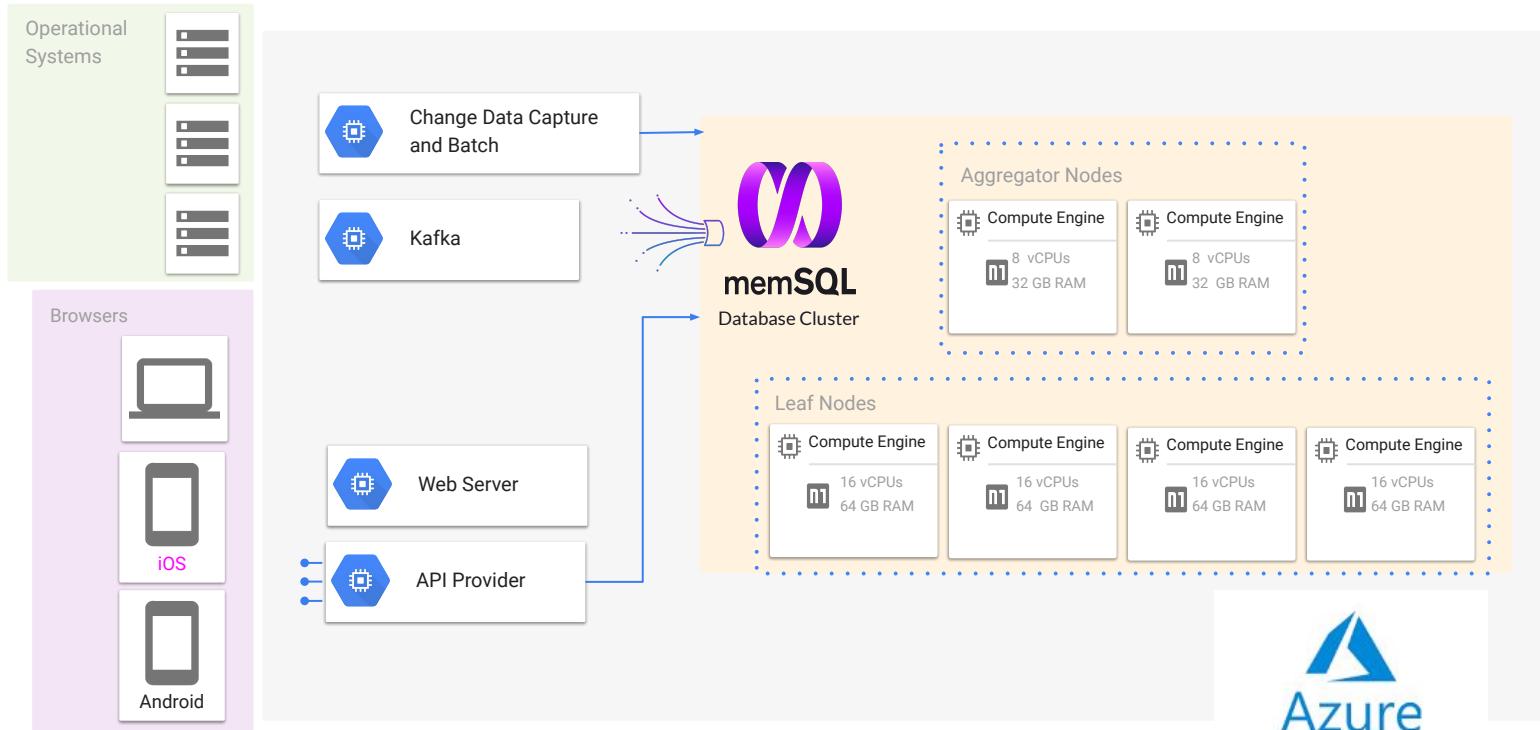
Select a deployment type

 Bare Metal or VMs 10-30 min read Deploy a MemSQL cluster on Linux using MemSQL tools. Start Guide	 Kubernetes 10-30 min read Deploy a MemSQL cluster using the MemSQL Kubernetes operator. Start Guide	 Cluster in a Box 10-30 min read Use Docker to deploy a MemSQL cluster-in-a-box for development. Start Guide
--	--	--

Was this Article Helpful?

Need help or have suggestions? Go to [MemSQL Forums](#)

MemSQL VM Deployment on MS Azure



MemSQL VM Deployment on Azure

- Determine appropriate number of VMs to spin up to support your MemSQL Cluster
 - Determine the number of Aggregators (scales concurrency)
 - Determine the number of Leaves you would like to Spin Up (scales workload and data capacity)
- Follow Instructions from this MemSQL VM Deployment Guide starting here:
 - <https://docs.memsql.com/v7.1/guides/deploy-memsql/self-managed/linux/step-1/>

MemSQL UI Installation Videos:

[Getting Started with MemSQL Setup UI](#)

[MemSQL Setup UI](#)

MemSQL VM Deployment - UI Installation

The screenshot shows a web browser window titled "Install MemSQL Tools - MemSQL" with tabs for "Instances | EC2 Management" and "MemSQL Studio". The URL is "ec2-18-235-248-49.compute-1.amazonaws.com:8081/#3c78f58e-cc62-4cf1-8f32-71b8e511a9c6". The main content area displays a step-by-step wizard for creating a new MemSQL cluster:

- Create New MemSQL Cluster**
- Configure Cluster Settings**
- Add & Provision Hosts**
- Configure Nodes**
- Review Cluster Configuration**
- Connect to Cluster**
- Connect to Studio**

Create New MemSQL Cluster

Before we start, please make sure your physical or virtual machines meet the following prerequisites:

System & Hardware Requirements:

- At least four CPU cores and eight GB of RAM per machine (8 vCPU and 32 GB of RAM is recommended for leaf nodes to align with license unit calculations).
- Running 64-bit version of RHEL/CentOS (6 or higher) or Debian (8 or higher).
- It is recommended you run MemSQL on NUMA-aware machines if you plan to co-locate multiple MemSQL nodes on a single host. [Learn more](#).

Port Requirements:

- Port 3306 open on all host machines for intra-cluster communication.

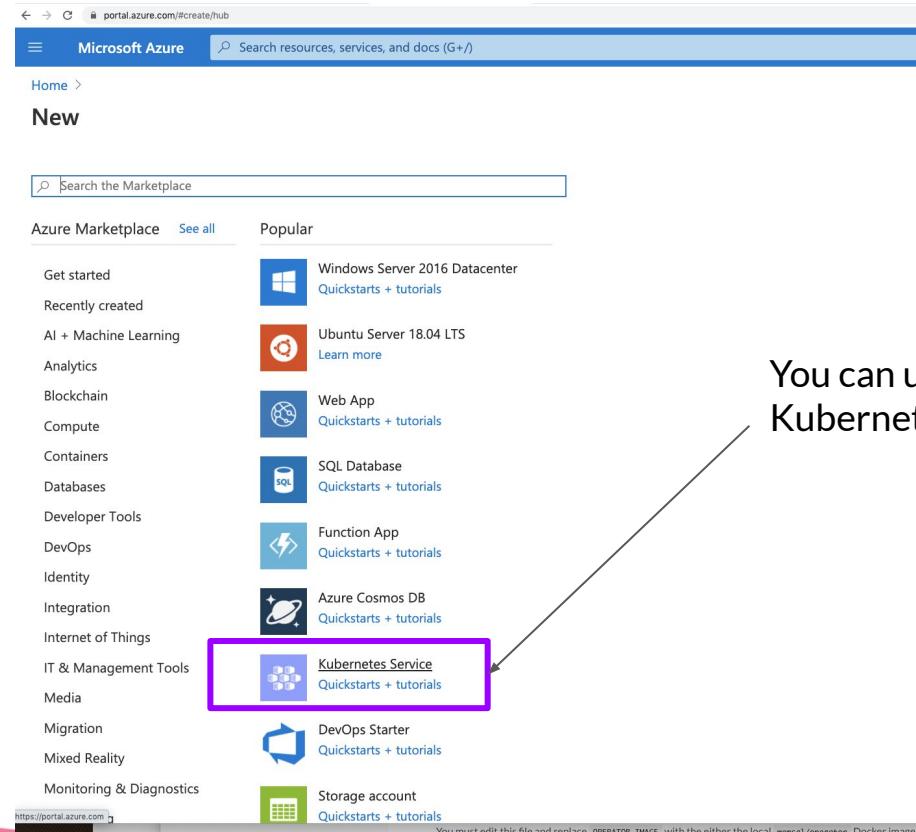
User Privilege:

- A non-root user with sudo privileges available on all host machines in the cluster that will be used to run MemSQL services and own the corresponding runtime state.

Learn more about the prerequisites for creating a new MemSQL cluster.

Next

MemSQL Kubernetes Deployment on Azure



Azure Marketplace [See all](#)

Popular

- Get started
- Recently created
- AI + Machine Learning
- Analytics
- Blockchain
- Compute
- Containers
- Databases
- Developer Tools
- DevOps
- Identity
- Integration
- Internet of Things
- IT & Management Tools
- Media
- Migration
- Mixed Reality
- Monitoring & Diagnostics

Search the Marketplace

Kubernetes Service

Windows Server 2016 Datacenter

Ubuntu Server 18.04 LTS

Web App

SQL Database

Function App

Azure Cosmos DB

DevOps Starter

Storage account

You can use the
Kubernetes Service

MemSQL Kubernetes Deployment on Azure

The screenshot shows the Microsoft Azure portal interface. At the top, there's a navigation bar with icons for back, forward, search, and user profile. Below it, the main dashboard features a section titled "Azure services" with various service icons: Create a resource (highlighted with a purple box), Kubernetes services, Virtual machines, App Services, Storage accounts, SQL databases, and Azure Database for PostgreSQL. Below this are links for Azure Cosmos DB, Function App, and More services. A large callout text "My Kubernetes Cluster is Available" points to the "Kubernetes services" icon. In the "Recent resources" section, a table lists a single item: Name (mta_hackathon), Type (Kubernetes service), and Last Viewed (29 minutes ago). The "mta_hackathon" row is also highlighted with a purple box. At the bottom, there are links for "Create", "Resource groups", and "All resources". The footer includes the URL <https://portal.azure.com/#create/hub> and the memSQL logo.

← → 🔍 portal.azure.com/#home

Microsoft Azure Search resources, services, and docs (G+/)

mlochbihler@Memsql.o... MEMSQL

Azure services

Create a resource

Kubernetes services

Virtual machines

App Services

Storage accounts

SQL databases

Azure Database for PostgreSQL...

Azure Cosmos DB

Function App

More services

My Kubernetes Cluster is Available

Recent resources

Name	Type	Last Viewed
mta_hackathon	Kubernetes service	29 minutes ago

Navigate

[Create](https://portal.azure.com/#create/hub) [Resource groups](#) [All resources](#)

memSQL

MemSQL Kubernetes Deployment on Azure

The screenshot shows the Microsoft Azure portal interface. At the top, there's a navigation bar with icons for back, forward, search, and user profile. Below it, the main dashboard features a section titled "Azure services" with various service icons: Create a resource (highlighted with a purple box), Kubernetes services, Virtual machines, App Services, Storage accounts, SQL databases, and Azure Database for PostgreSQL. Below this are links for Azure Cosmos DB, Function App, and More services. A large callout text "My Kubernetes Cluster is Available" points to the "Kubernetes services" icon. In the "Recent resources" section, a table lists a single item: Name (mta_hackathon), Type (Kubernetes service), and Last Viewed (29 minutes ago). The "mta_hackathon" row is also highlighted with a purple box. At the bottom, there are links for "Create", "Resource groups", and "All resources". The footer includes the URL <https://portal.azure.com/#create/hub> and the memSQL logo.

← → 🔍 portal.azure.com/#home

Microsoft Azure Search resources, services, and docs (G+/)

mlochbihler@Memsql.o... MEMSQL

Azure services

Create a resource

Kubernetes services

Virtual machines

App Services

Storage accounts

SQL databases

Azure Database for PostgreSQL...

Azure Cosmos DB

Function App

More services

My Kubernetes Cluster is Available

Recent resources

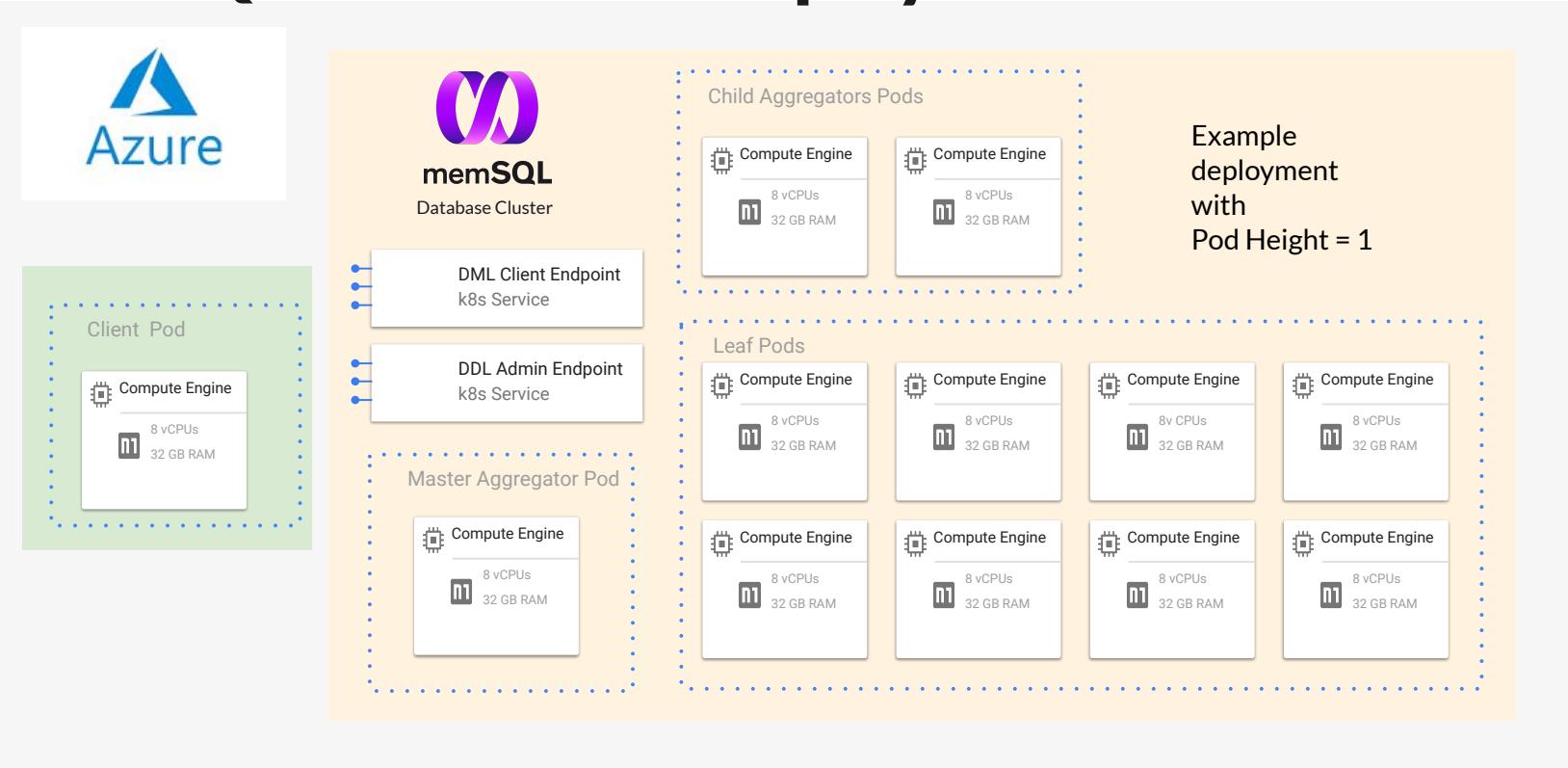
Name	Type	Last Viewed
mta_hackathon	Kubernetes service	29 minutes ago

Navigate

[Create](https://portal.azure.com/#create/hub) [Resource groups](#) [All resources](#)

memSQL

MemSQL Kubernetes Deployment on Azure



MemSQL Kubernetes Deployment on Azure

Microsoft Azure Search resources, services, and docs (G+)

Home > mta_hackathon Kubernetes service

Search (Cmd+/) Delete Refresh

Overview

Resource group (change) : mta-hackathon

Status : Succeeded

Location : East US

Subscription (change) : memsql-master

Subscription ID : 181c043a-8104-4baa-95cf-bcdc211723d8

Kubernetes version : 1.15.11

API server address : mta-hackathon-dns-693a9e99.hcp.eastus.azmk8s.io

Network type (plugin) : Basic (Kubenet)

Node pools : 1 node pool

Tags (change) : CostCenter : SE Name : k8s-cluster Owner : mlochbihler Project : mta-hackathon

Activity log Access control (IAM) Tags Diagnose and solve problems Security

Monitor containers Get health and performance insights Go to Azure Monitor insights

View logs Search and analyze logs using ad-hoc queries Go to Azure Monitor logs

Bash Requesting a Cloud Shell. Succeeded. Connecting terminal...

```
mark@Azure:~/memsql$ kubectl get pods
```

NAME	READY	STATUS	RESTARTS	AGE
memsql-operator-cb6465b56-2f6q8	1/1	Running	0	58m
node-memsql-cluster-aggregator-0	2/2	Running	0	58m
node-memsql-cluster-leaf-ag1-0	2/2	Running	0	58m
node-memsql-cluster-leaf-ag1-1	2/2	Running	0	58m
node-memsql-cluster-master-0	2/2	Running	0	58m

mark@Azure:~/memsql\$

https://portal.azure.com/

MemSQL Kubernetes Deployment on Azure

Microsoft Azure Search resources, services, and docs (G+) mlochbihler@Memsql.o... MEMSQL

Home > mta_hackathon Kubernetes service

Search (Cmd+ /) Delete Refresh

Overview Resource group (change) : mta-hackathon Status : Succeeded Location : East US Subscription (change) : memsql-master Subscription ID : 181c043a-8104-4baa-95cf-bcdc211723d8 Kubernetes version : 1.15.11 API server address : mta-hackathon-dns-693a9e99.hcp.eastus.azmk8s.io Network type (plugin) : Basic (Kubenet) Node pools : 1 node pool Tags CostCenter : SE Name : k8s-cluster Owner : mlochbihler Project : mta-hackathon

Bash Requesting a Cloud Shell. Succeeded. Connecting terminal...

```
mark@Azure:~$ cd memsql
mark@Azure:~/memsql$ kubectl get services
NAME         TYPE      CLUSTER-IP   EXTERNAL-IP   PORT(S)        AGE
kubernetes   ClusterIP  10.0.0.1    <none>        443/TCP       33h
svc-memsql-cluster   ClusterIP  None        <none>        3306/TCP      60m
svc-memsql-cluster-ddl LoadBalancer 10.0.226.16  52.142.27.165 3306:32237/TCP 60m
svc-memsql-cluster-dml LoadBalancer 10.0.175.125 52.147.218.140 3306:31538/TCP 60m
mark@Azure:~/memsql$
```

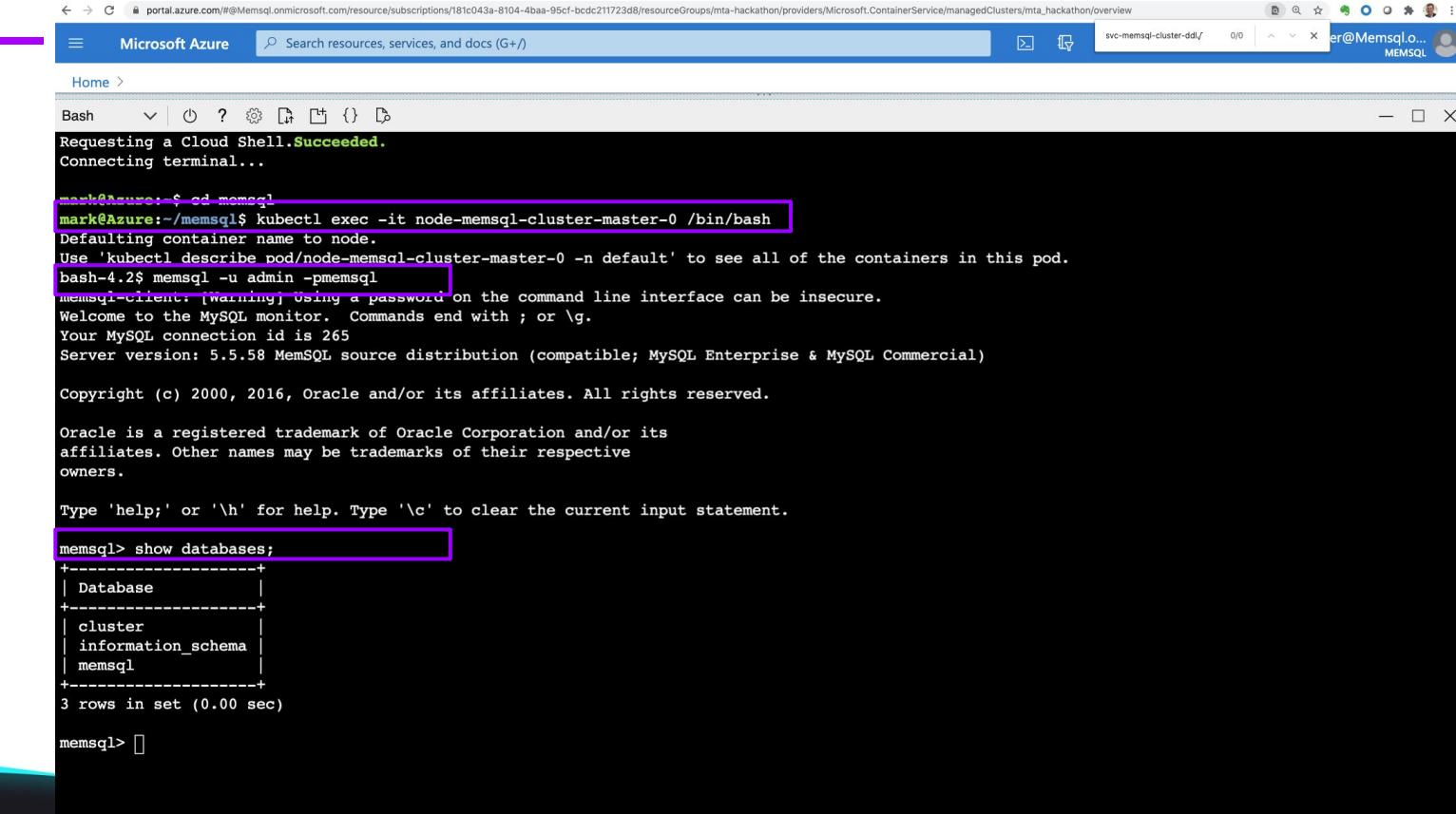
MemSQL Kubernetes Deployment on Azure

The screenshot shows the Microsoft Azure portal interface. At the top, there's a navigation bar with the URL https://portal.azure.com/#@MemSQL.onmicrosoft.com/resource/subscriptions/181c043a-8104-4baa-95cf-bcdc211723d8/resourceGroups/mta-hackathon/providers/Microsoft.ContainerService/managedClusters/mta_hackathon/overview. Below the navigation bar, the main content area shows a Kubernetes service named "mta_hackathon". The service is listed under the "Kubernetes service" category. The "Overview" tab is selected, showing details like the Resource group (change) to "mta-hackathon", Status "Succeeded", Kubernetes version "1.15.11", and API server address "mta-hackathon-dns-693a9e99.hcp.eastus.azmk8s.io". Below the overview, there's an "Activity log" section and a terminal window titled "Bash". In the terminal, the user has run the command `mark@Azure:~/memsql$ kubectl describe service svc-memsql-cluster-dml`. The output of this command is highlighted with a purple rectangle and shows the following details:

```
Name: svc-memsql-cluster-dml
Namespace: default
Labels: app.kubernetes.io/component=aggregator
         app.kubernetes.io/instance=memsql-cluster
         app.kubernetes.io/name=memsql-cluster
         custom=label
Annotations: custom: annotations
             service.beta.kubernetes.io/aws-load-balancer-connection-idle-timeout: 4000
Selector: app.kubernetes.io/component=aggregator,app.kubernetes.io/instance=memsql-cluster,app.kubernetes.io/name=memsql-cluster
Type: LoadBalancer
IP: 10.0.175.125
LoadBalancer Ingress: 52.147.218.140
Port: <unset> 3306/TCP
TargetPort: 3306/TCP
NodePort: <unset> 31538/TCP
Endpoints: 10.244.0.9:3306
Session Affinity: None
External Traffic Policy: Cluster
Events: <none>
```

At the bottom of the terminal window, the prompt `mark@Azure:~/memsql$` is visible.

MemSQL Kubernetes Deployment on Azure



The screenshot shows a Microsoft Azure Cloud Shell interface. The title bar includes the URL portal.azure.com/#@Memsql.onmicrosoft.com/resource/subscriptions/181c043a-8104-4baa-95cf-bcdc211723d8/resourceGroups/mta-hackathon/providers/Microsoft.ContainerService/managedClusters/mta_hackathon/overview, the Microsoft Azure logo, a search bar, and a user profile for 'mark@Memsql... MEMSQL'.

The main area is a terminal window titled 'Bash'. It displays the following session:

```
Requesting a Cloud Shell.Succeeded.
Connecting terminal...

mark@Azure:~$ cd memsql
mark@Azure:~/memsql$ kubectl exec -it node-memsql-cluster-master-0 /bin/bash
Defaulting container name to node.
Use 'kubectl describe pod/node-memsql-cluster-master-0 -n default' to see all of the containers in this pod.
bash-4.2$ memsql -u admin -pmemsql
memsql-client: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 265
Server version: 5.5.58 MemSQL source distribution (compatible; MySQL Enterprise & MySQL Commercial)

Copyright (c) 2000, 2016, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

memsql> show databases;
+-----+
| Database |
+-----+
| cluster  |
| information_schema |
| memsql   |
+-----+
3 rows in set (0.00 sec)

memsql> 
```

Several lines of the command history are highlighted with a purple rectangular selection, specifically the command `kubectl exec -it node-memsql-cluster-master-0 /bin/bash` and the subsequent MySQL session commands.