ORACLE DevLive | Level Up | MySQL Summit

Query Processing on Object Store with MySQL HeatWave (Lakehouse)

Nipun Agarwal
Senior Vice President, MySQL Database and HeatWave
Oracle

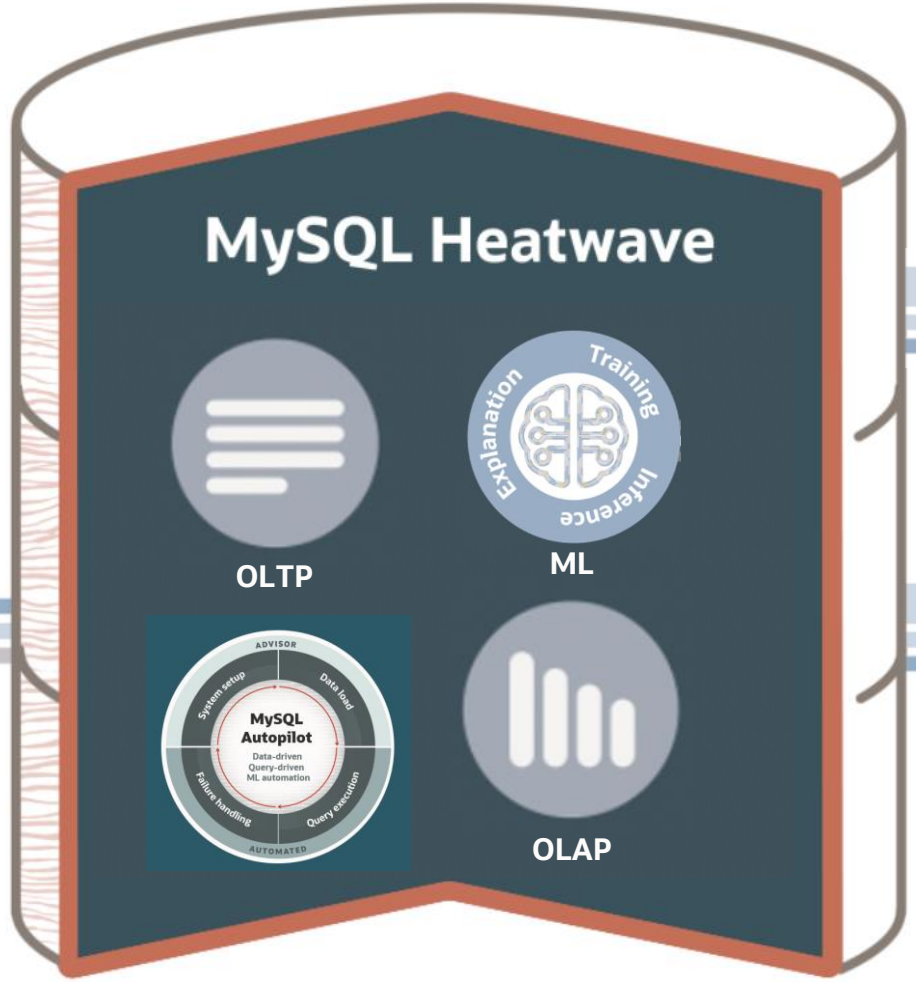
MySQL HeatWave – single database for OLTP, OLAP and ML

No change to existing MySQL applications

OLTP Applications

- Social
- ECommerce
- FinTech
- SaaS

Real-time ML recommendations



ML training on latest data

Real-time analytics

ML Tools

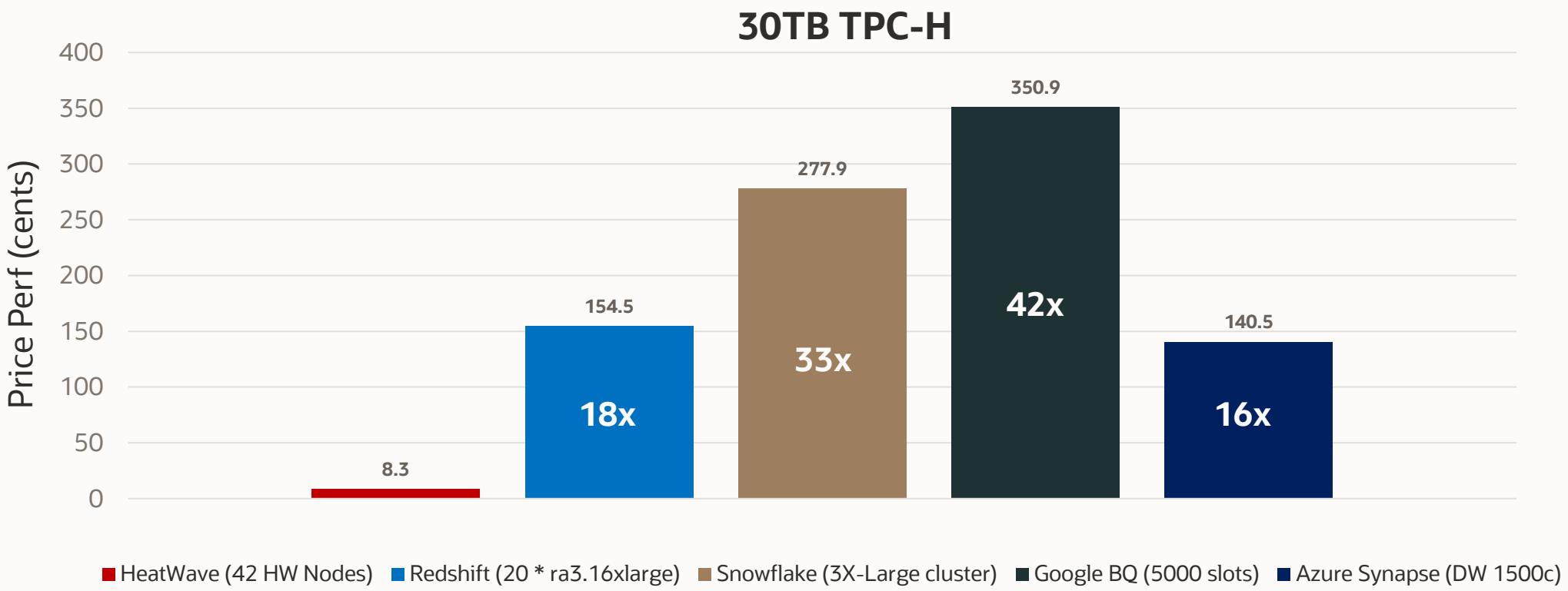
- jupyter
- Apache Zeppelin

Analytics Tools

- ORACLE Analytics Cloud
- tableau
- Qlik
- looker

Price-Performance comparison with analytic databases

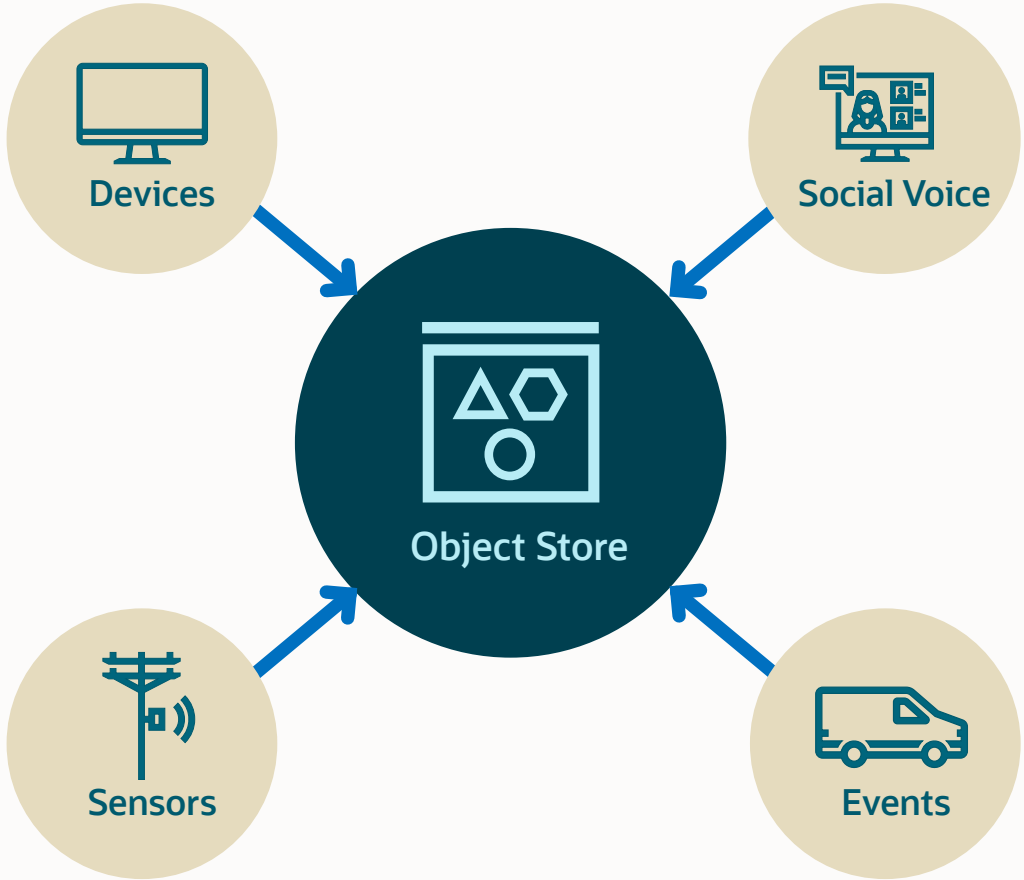
18x better than Redshift, 33x better than Snowflake, 42x better than BigQuery, 17x better than Synapse



- Pricing for Redshift is based on 1 year reserved instance, paid upfront. For Snowflake is based on **standard edition**
- Pricing for Google Big Query is based on monthly flat rate commitment. For Azure Synapse is based on 1 year reserved pricing

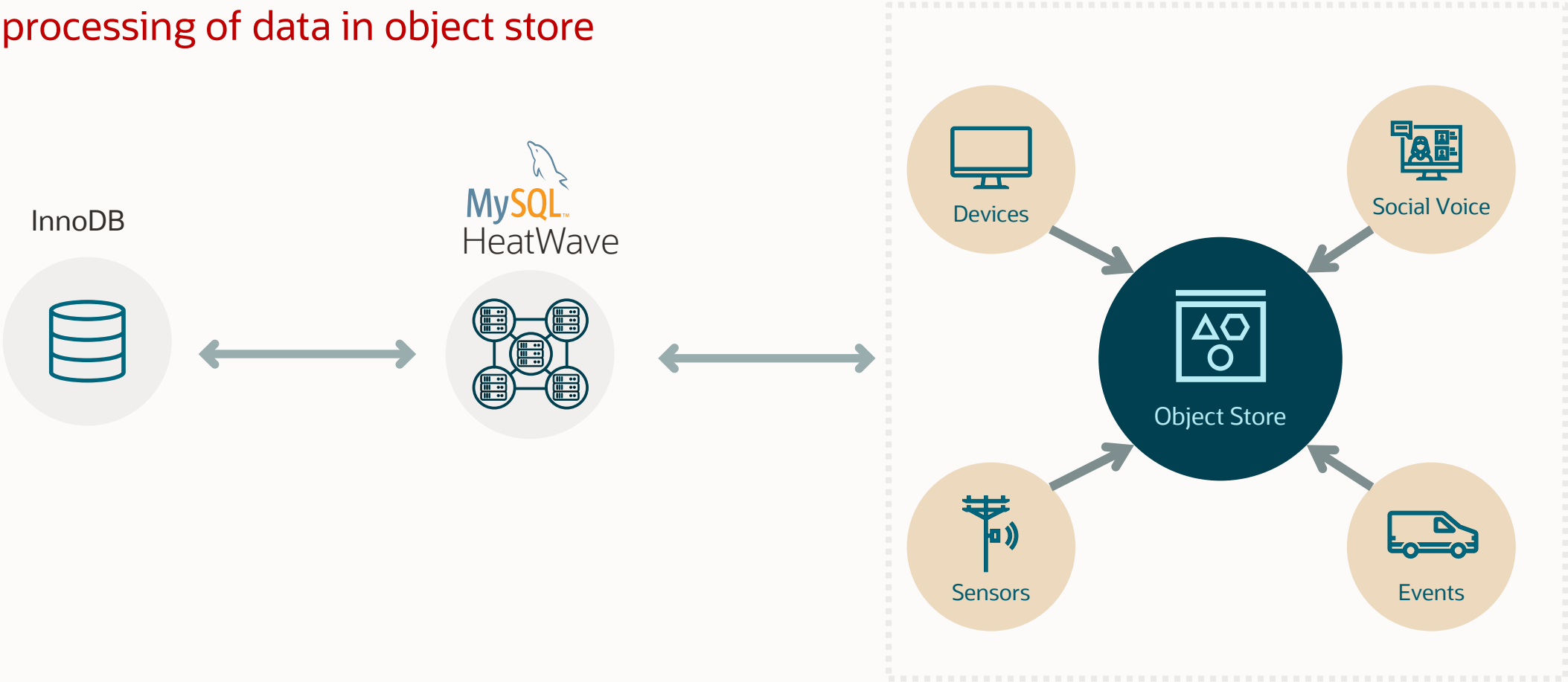
Significant growth of data outside databases

- Databases are systems of record
- Data lake is the repository for other types of data
- 79 Zetabytes of data generated in 2021, 180 ZB expected in 2025 (IDC)
- In 2019, IOTs alone generated 13.6 ZB of data
- 99.5% of collected data remains unused
- More than 80% of data is unstructured



MySQL HeatWave Lakehouse – now in Beta

Query processing of data in object store



Fully compliant with MySQL syntax

Example: querying across relational and external data source

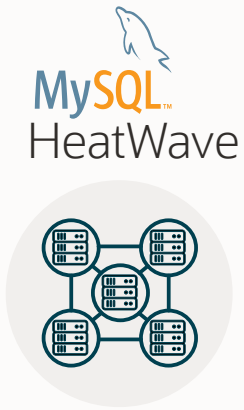
- A csv file in object store contains telemetry data collected from temperature sensors (temp_sensor_1.csv)
- Operational sales data is in MySQL database (InnoDB relation SALES)
- User wants to query number of bottles of beverage sold on days when temperature > 30℃

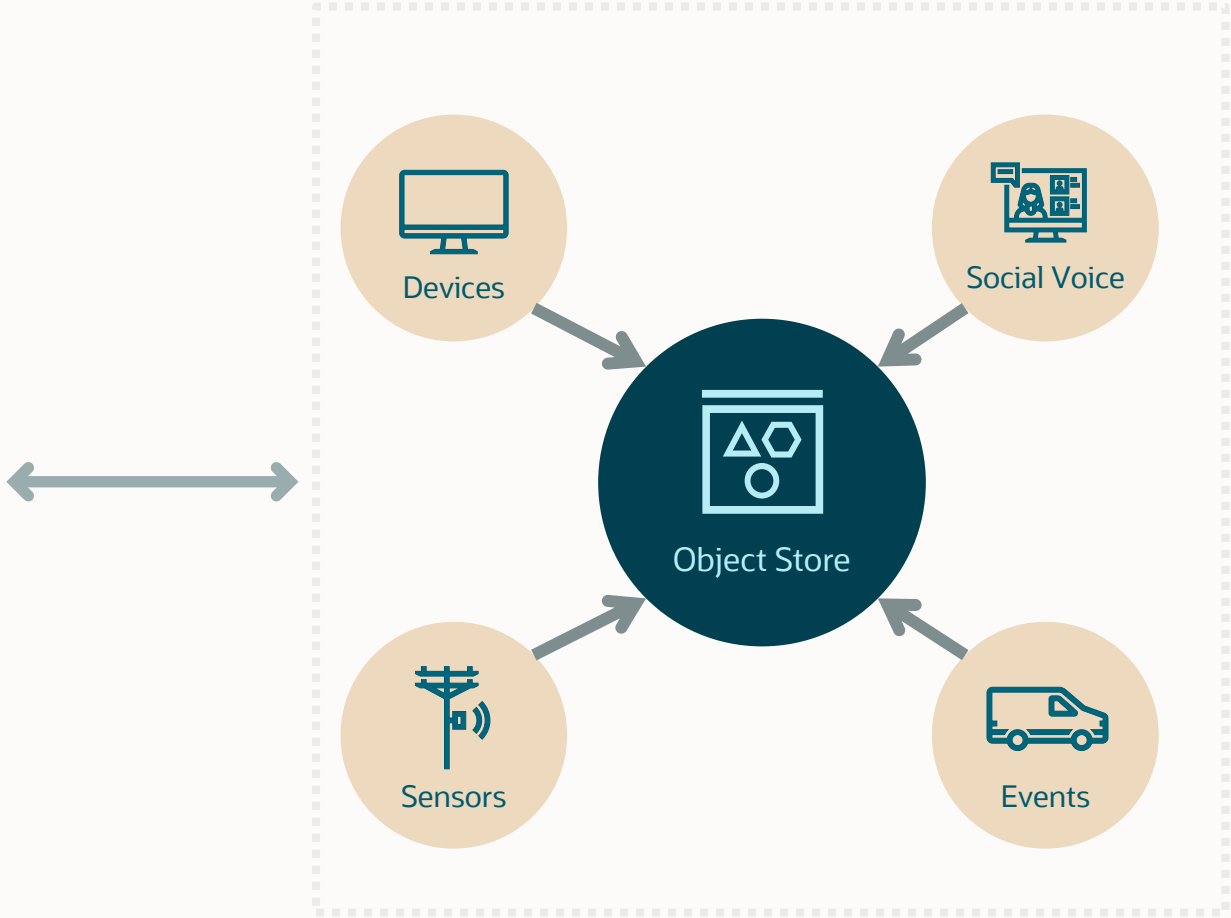
```
mysql> CREATE TABLE Sensor {date DATE, degrees INT} SECONDARY_ENGINE RAPID
SECONDARY_EXTERNAL_SOURCE ”temp_sensor_1.csv”;

mysql> ALTER TABLE Sensor SECONDARY_LOAD;

mysql> ALTER TABLE SALES SECONDARY_LOAD;

mysql> SELECT count(*) FROM Sensor, SALES WHERE Sensor.degrees > 30 and
Sensor.date = SALES.date;
```

400TB TPCH, 512 nodes

42 seconds

Average execution time





Comparison with Snowflake

17x faster **Query**

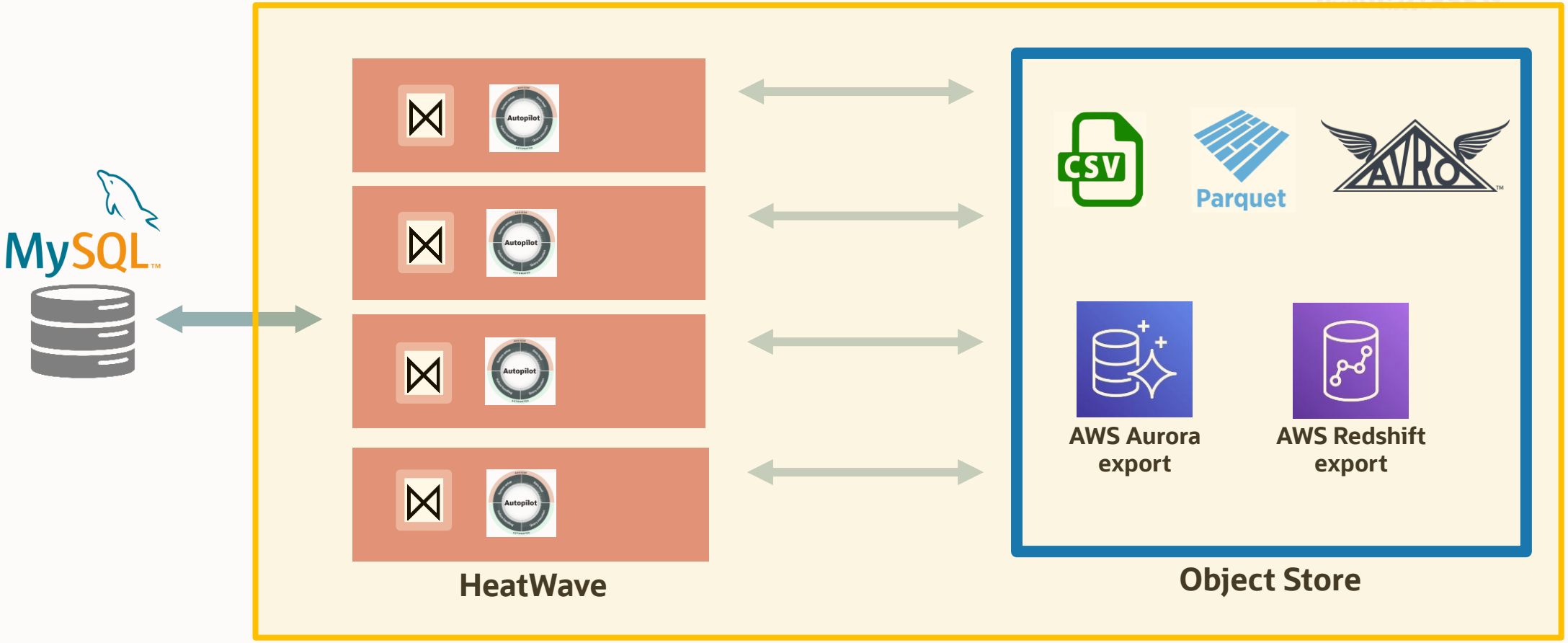
2.7x faster **Load**

30% lower **Cost**

+ OLTP + ML

Scale out architecture & Autopilot

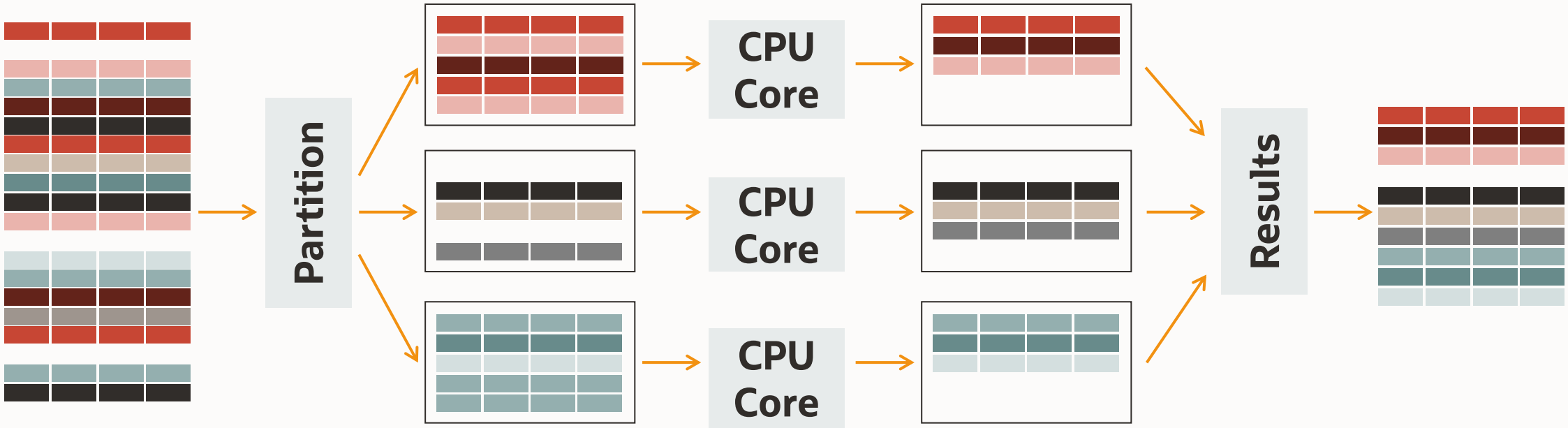
Processing data in object store, multiple file formats, larger data sizes



Data in object store remains in object store. Only metadata created in MySQL

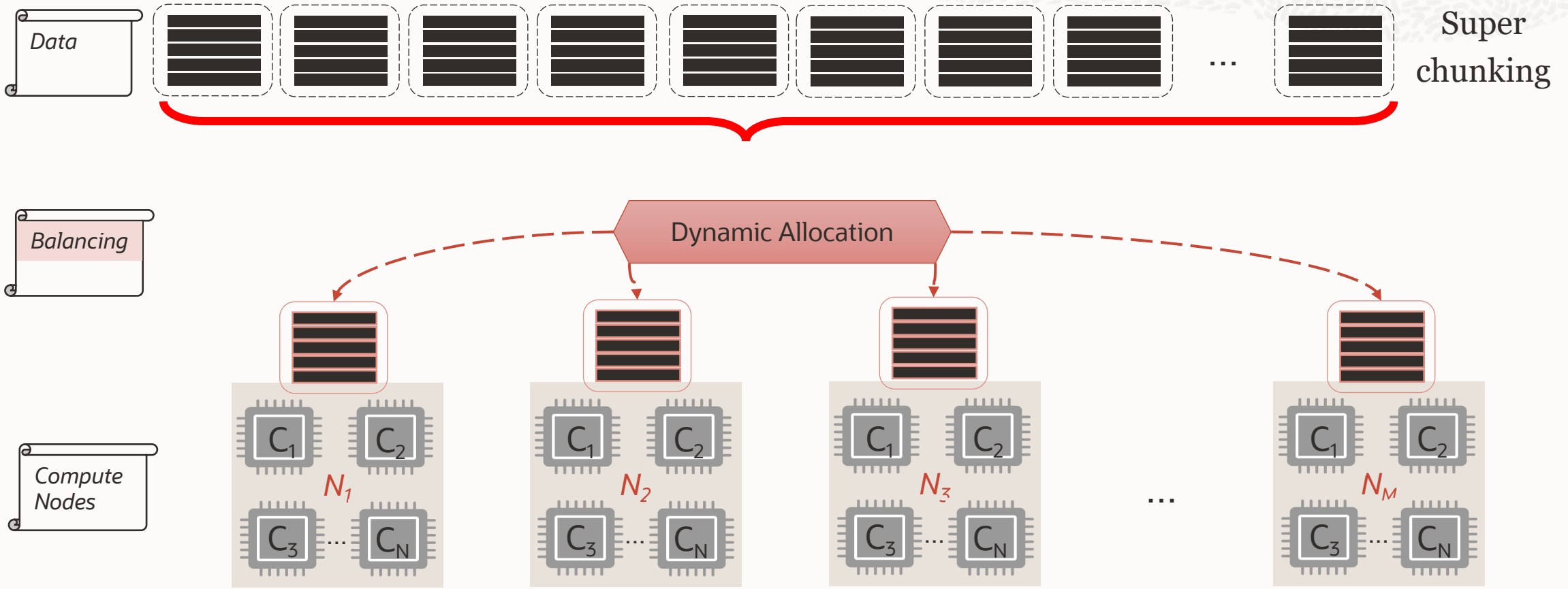
Scaling out query processing

Enabled by a massively partitioned architecture



Scaling out load performance

Super chunking to mitigate impact of stragglers

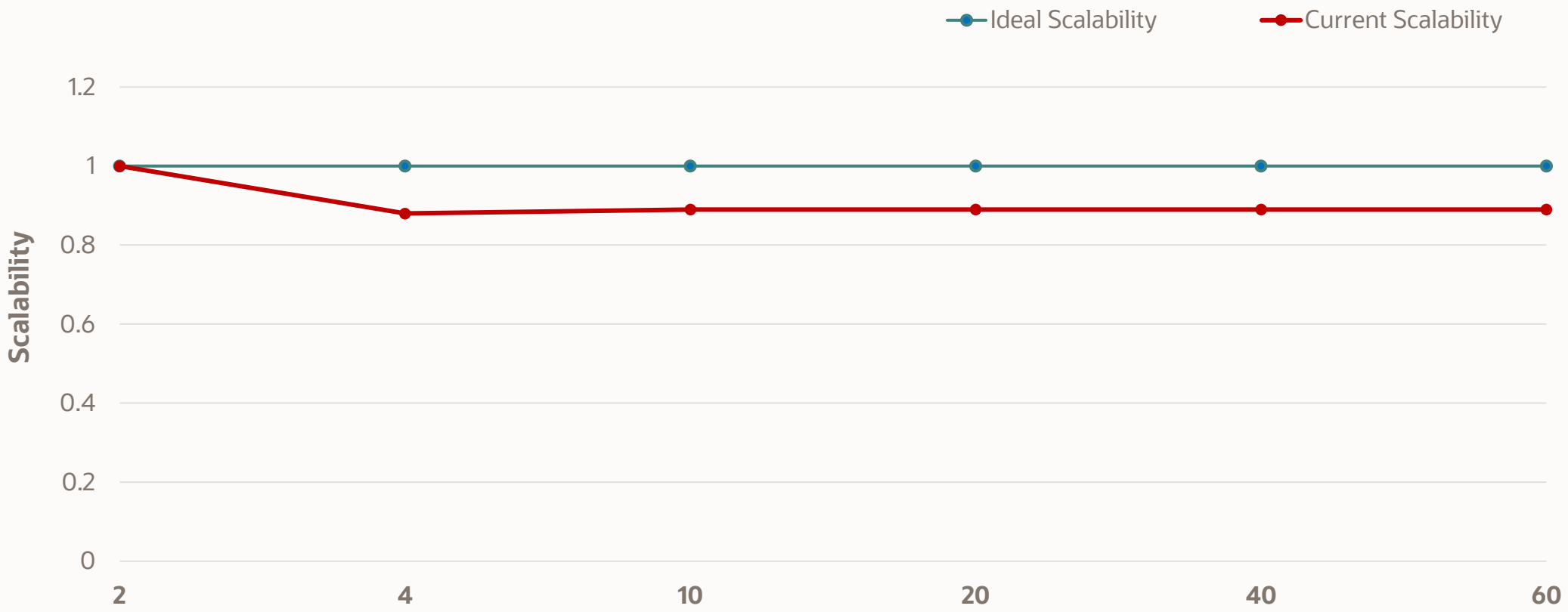


- Stragglers due to varying latency from object store
- Difference in computation time for each chunk

- Super chunking creates more tasks than #cores
- Dynamic allocation of chunks to nodes

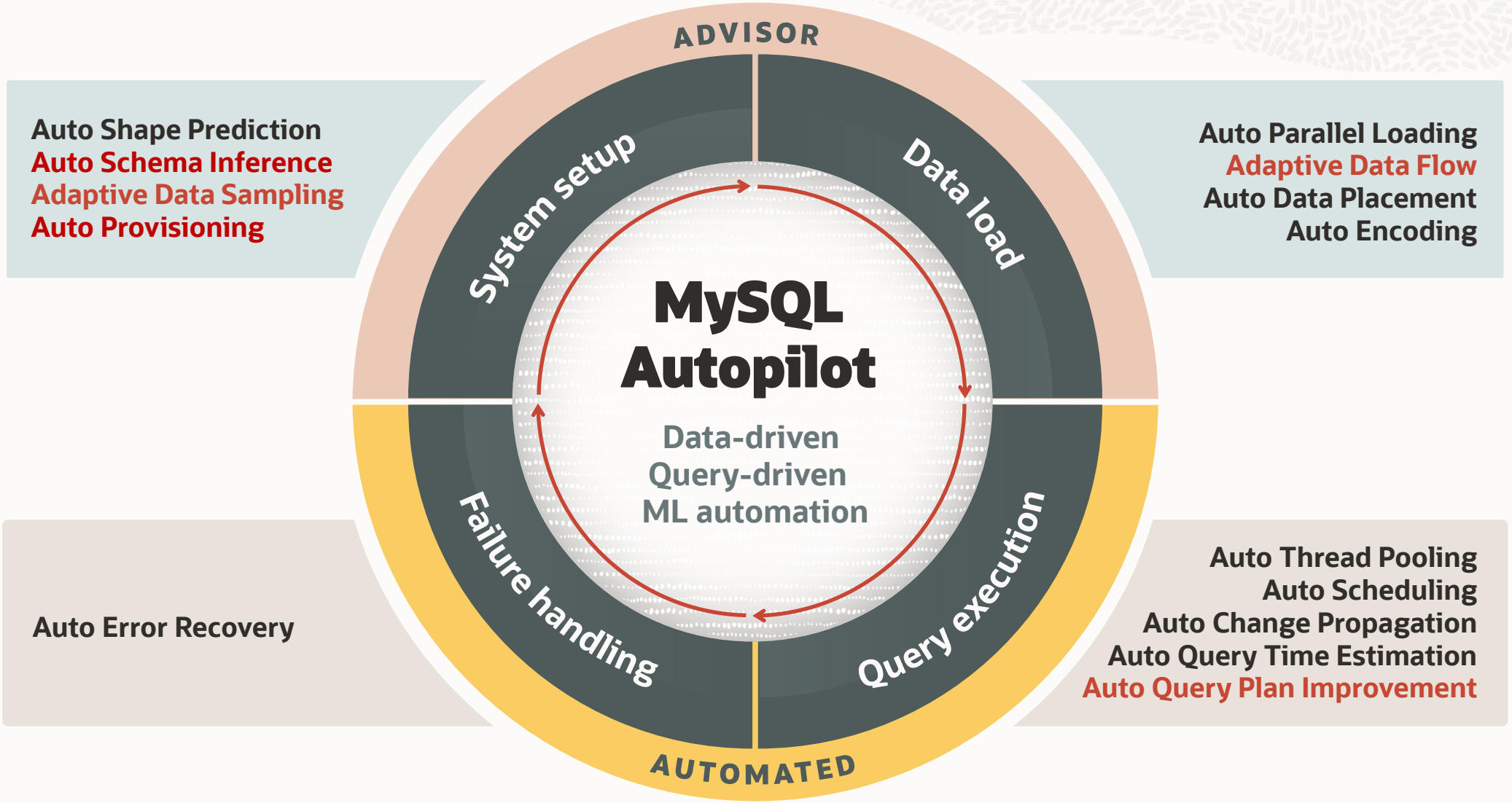
HeatWave scales near linearly with cluster size

TPCH, TPCDS



MySQL Autopilot: machine learning powered automation

Helps achieve good query performance & ease of management



Same query performance for data inside MySQL or in object store

	Inside MySQL		Object Store	
TPC-H	# of nodes	Query time	# of nodes	Query time
10 TB	12	16 sec	12	16 sec
30 TB	36	18 sec	36	18 sec

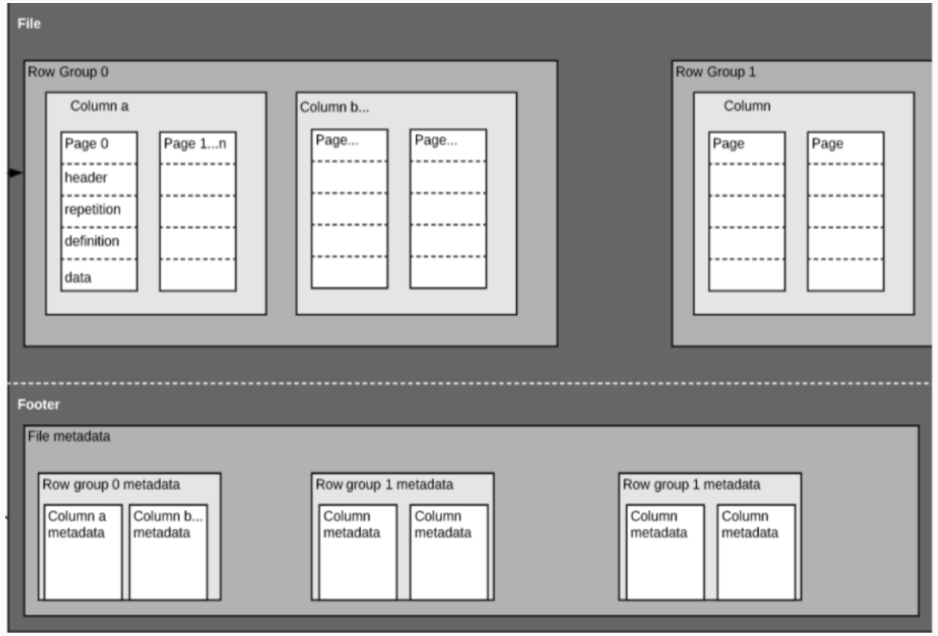
Performance of MySQL HeatWave on object store

Faster to load, faster to query and less expensive than other databases

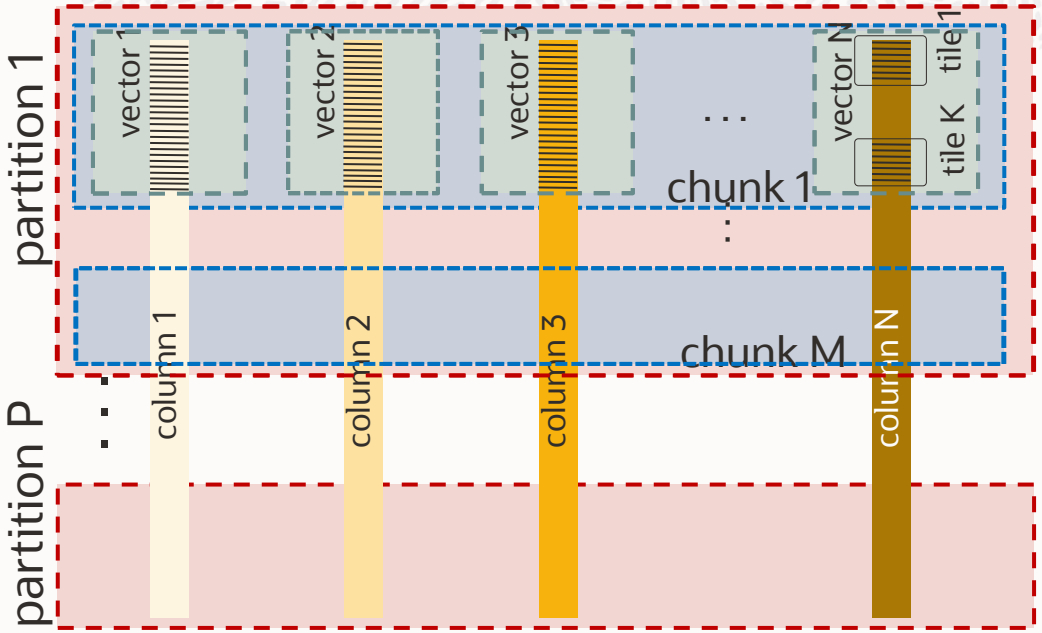
400TB TPC-H (CSV format)	MySQL HeatWave	Redshift	Snowflake
Annual cost	$1,589,036	$2,261,760	$2,242,560
Load Time	4.1 hrs	32.3 hrs	11.2 hrs
Query Time	42 sec	268 sec	712 sec

Parquet files, Aurora, Redshift export files can be directly loaded to HeatWave

Efficient transformation into HeatWave hybrid columnar format



Parquet



HeatWave Hybrid Columnar

10TB TPCH (12 nodes)	HeatWave Load time	Query performance
CSV format	2 hrs	19 sec
Parquet format (optimized layout)	2 hrs	19 sec
Aurora backup	7.2 hrs (Redshift takes 11.3 hrs)	19 sec

Autopilot: Auto schema inference

Easy to process files in object store

```
CREATE DATABASE `tpch_100T`;

CREATE TABLE `tpch_100T`.`customer`(
     `col_0` bigint NOT NULL, `col_1` varchar(20) NOT NULL,
     `col_2` varchar(40) NOT NULL, `col_3` tinyint NOT NULL,
     `col_4` varchar(15) NOT NULL, `col_5` decimal(6,2) NOT NULL,
     `col_6` varchar(10) NOT NULL, `col_7` varchar(116) NOT NULL)
    ENGINE=datalake SECONDARY_ENGINE=RAPID
    ENGINE_ATTRIBUTE='{"file": [{"prefix": "src_data/tpch_100T/customer/"}],
                       "dialect": {"format": "csv", "field_delimiter": "|",
                                   "record_delimiter": "|\\n"}}';

ALTER TABLE `tpch_100T`.`customer` SECONDARY_LOAD;
```

1

2

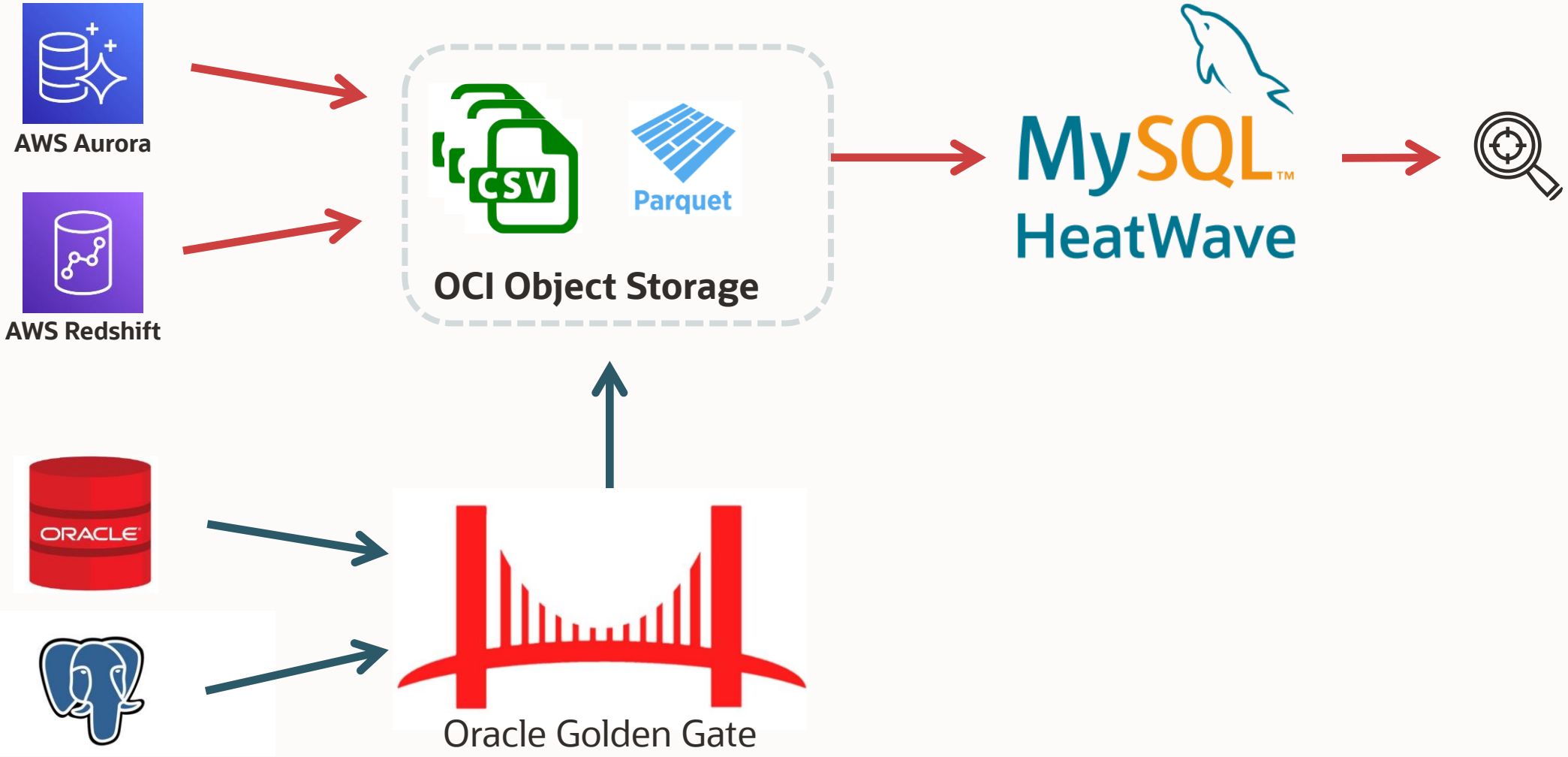
3

Auto Provisioning

```
CAPACITY ESTIMATION

Default load pool for tables: SNAPSHOT
Default encoding for string columns: VARLEN (unless specified in the schema)
Estimating memory footprint for 1 schema(s)

                          TOTAL      ESTIMATED      ESTIMATED      TOTAL     DICTIONARY      VARLEN      ESTIMATED
SCHEMA                OFFLOADABLE  HEATWAVE NODE    MYSQL NODE    STRING      ENCODED      ENCODED           LOAD
NAME                       TABLES      FOOTPRINT     FOOTPRINT   COLUMNS      COLUMNS      COLUMNS           TIME
`tpch_100T`                     8       61.78 TiB      4.31 MiB        29            0           29        1.98 h

Sufficient MySQL host memory available to load all tables.
HeatWave cluster memory might be insufficient to load all the tables.
The estimated load time assumes a cluster with sufficient size.
Please refer to the user manual for more details.
```


MySQL HeatWave can process data from multiple data sources

Data can be in a file or other databases



Summary

1. MySQL HeatWave can now process data inside MySQL and in object store in multiple file formats
2. Processing data in object store has comparable performance to querying data inside MySQL
3. Fully compliant with standard MySQL syntax
4. Offers best performance and price performance for query processing and loading data
5. MySQL Autopilot extended to improve manageability of processing from object store
6. Now available in OCI

Thank You.