



# Deploying a modern Data Warehouse with Amazon Redshift

**Ninad Phatak**

Principal Data Architect  
Amazon Development Center India Pvt Ltd.

# Agenda

- Evolving Nature of Data & Analytics
- Amazon Redshift Architecture & Key Concepts
- Why is Amazon Redshift your modern Data Warehouse?
  - Performance at scale
  - Auto-everything
  - Advanced Analytics capabilities
  - Integration with Analytics Services
  - Variety of personas and use cases
- Customer case studies
- Migration to Amazon Redshift

# Evolving Nature of Data & Analytics

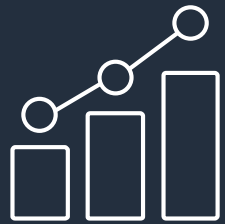
Agility is more  
important than ever

vyaire<sup>™</sup>  
MEDICAL

moderna



# Customers want more value from their data



---

Growing  
Exponentially



---

From new  
sources



---

Increasingly  
diverse



---

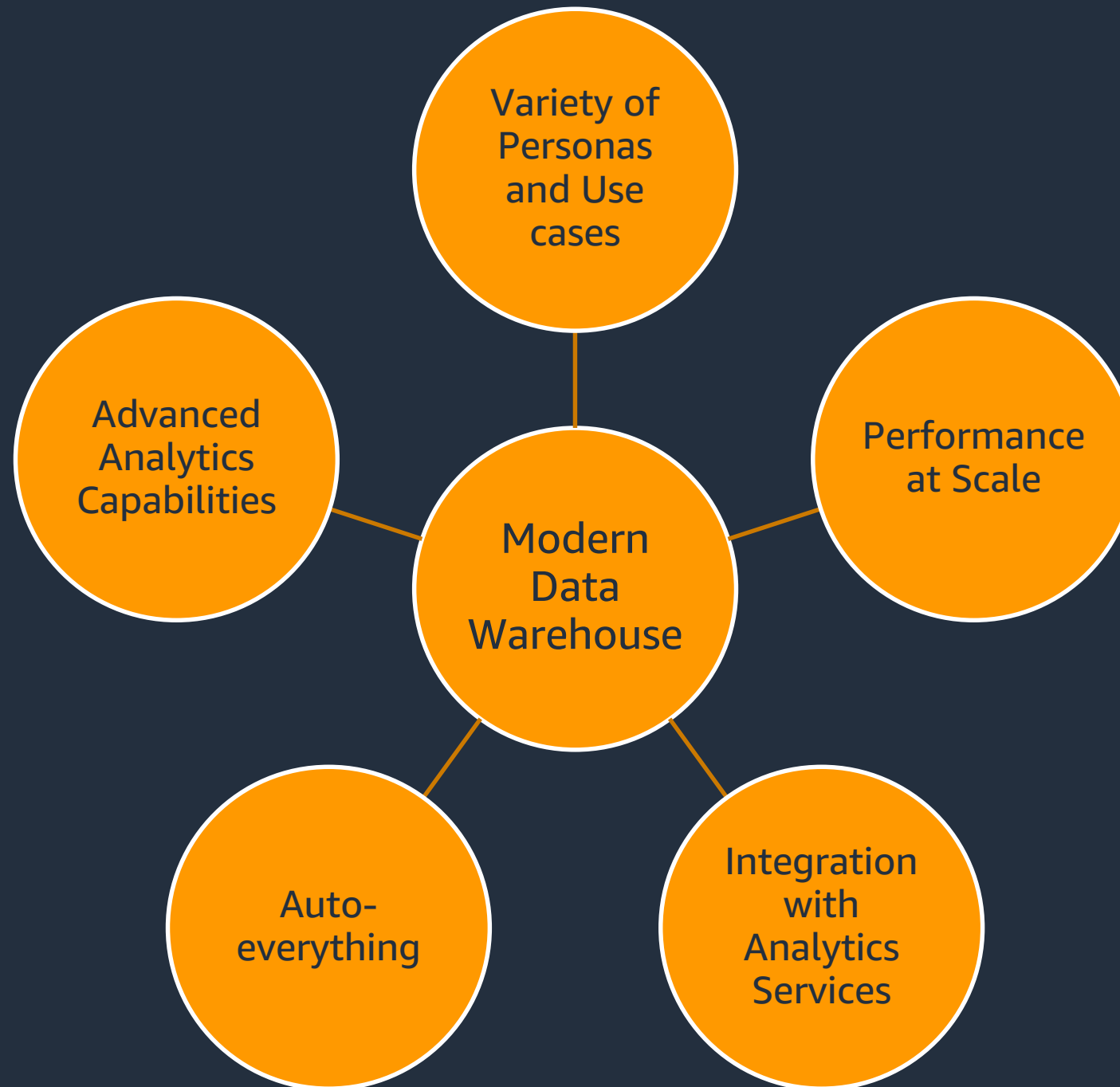
Used by  
many people



---

Analyzed by many  
applications

# Asks from a Modern Data Warehouse today



# Lake House architecture on AWS



**Scalable data lakes**

**Purpose-built data services**

**Seamless data movement**

**Unified governance**

**Performant and cost-effective**



# Amazon Redshift

Analyze all your data with the fastest and most widely used cloud data warehouse



## Analyze all your data

Deepest integration with your data lake



## Performance at any scale

Up to 3x better price performance than other cloud DW



## Lower your costs

At least 50% less expensive than other cloud DW



# Amazon Redshift Architecture & Key concepts

# Amazon Redshift Architecture

Massively parallel, shared nothing columnar architecture

## Leader node

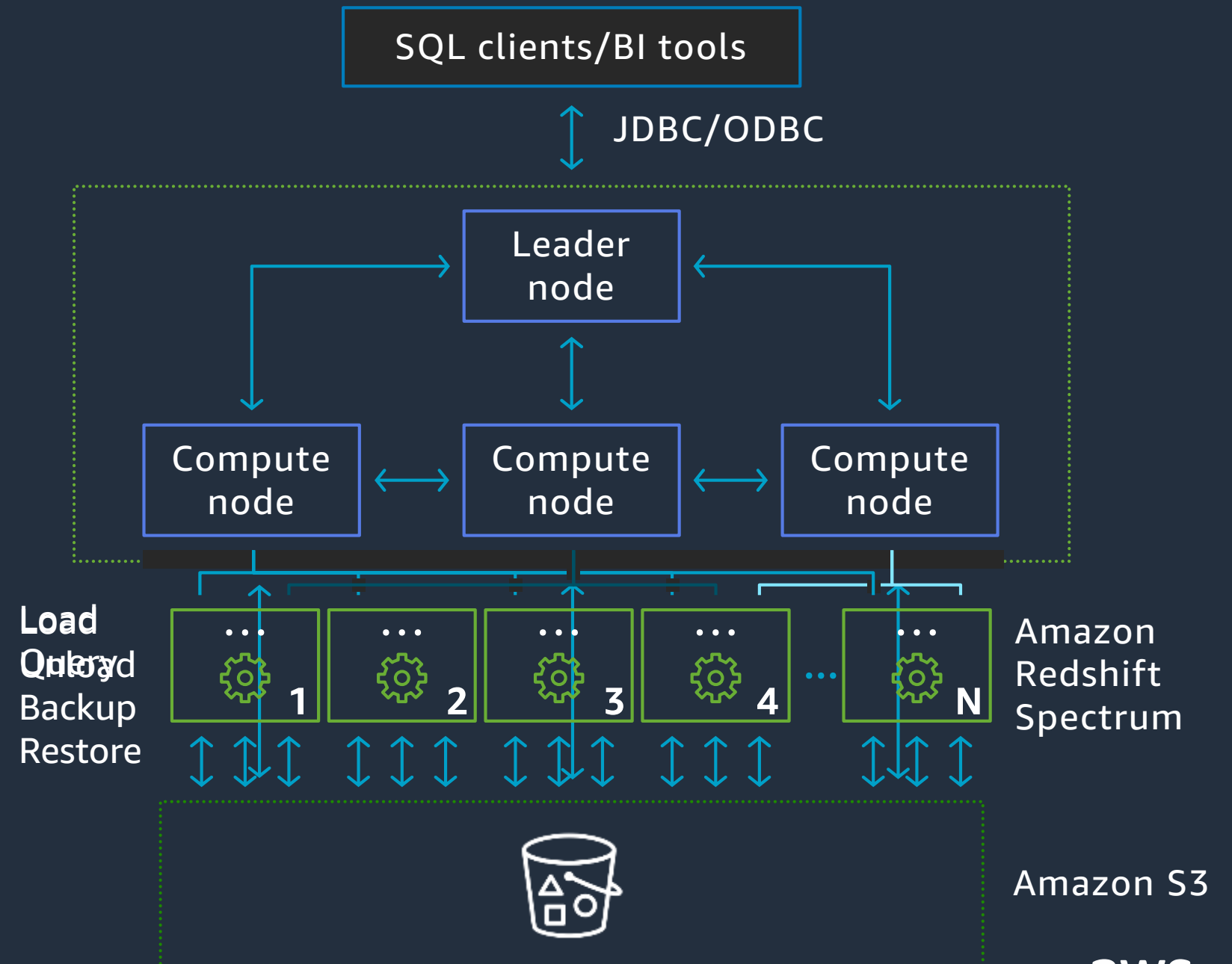
- SQL endpoint
- Stores metadata
- Coordinates parallel SQL processing

## Compute nodes

- Local, columnar storage
- Executes queries in parallel
- Load, unload, backup, restore

## Amazon Redshift Spectrum nodes

- Execute queries directly against Amazon Simple Storage Service (Amazon S3)



# Redshift Node types(2<sup>nd</sup> Generation)

## Dense compute—DC2

- Solid-state disks

## Dense storage—DS2

- Magnetic disks

Instance type	Disk type	Size	Memory(GiB)	vCPUs	Slices
DC2 large	SSD	160 GB	15	2	2
DC2 8xlarge	SSD	2.56 TB	244	32	16
DS2 xlarge	Magnetic	2 TB	31	4	2
DS2 8xlarge	Magnetic	16 TB	244	36	16



# Columnar Architecture

Amazon Redshift uses a columnar architecture for storing data on disk

Physically store data on disk by column rather than row

Only read the column data that is required

---

**Goal:** Reduce I/O for analytics queries

# Columnar Architecture: Example

```
CREATE TABLE deep_dive (  
  aid INT --audience_id  
  ,loc CHAR(3) --location  
  ,dt DATE --date  
);
```

aid	loc	dt
1	SFO	2017-10-20
2	JFK	2017-10-20
3	SFO	2017-04-01
4	JFK	2017-05-14

aid	loc	dt

```
SELECT min(dt) FROM deep_dive;
```

## Row-based storage

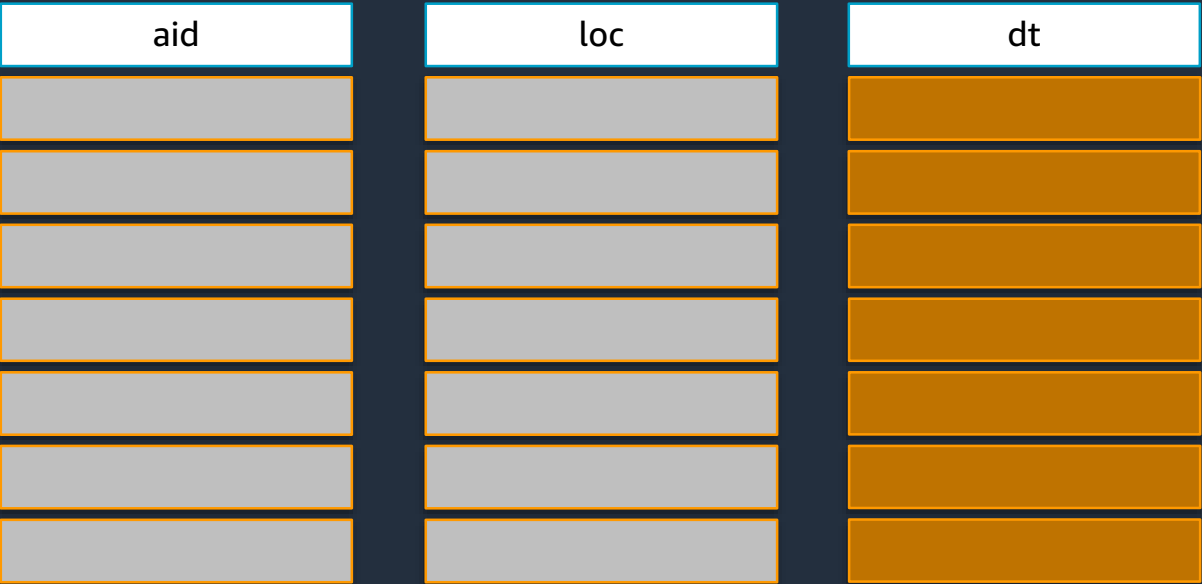
- Need to read everything
- Unnecessary I/O



# Columnar Architecture: Example

```
CREATE TABLE deep_dive (  
  aid INT --audience_id  
  ,loc CHAR(3) --location  
  ,dt DATE --date  
);
```

aid	loc	dt
1	SFO	2017-10-20
2	JFK	2017-10-20
3	SFO	2017-04-01
4	JFK	2017-05-14



```
SELECT min(dt) FROM deep_dive;
```

## Column-based storage

- Only scans blocks for relevant column



# Compression

## Goals

Allow more data to be stored within an Amazon Redshift cluster

Improve query performance by decreasing I/O

---

## Impact

Allows two to four times more data to be stored within the cluster

`ANALYZE COMPRESSION` is a built-in command that will find the optimal compression for each column on an existing table



# Compression: Example

```
CREATE TABLE deep_dive (  
  aid INT      --audience_id  
  ,loc CHAR(3) --location  
  ,dt DATE     --date  
);
```

aid	loc	dt
1	SFO	2017-10-20
2	JFK	2017-10-20
3	SFO	2017-04-01
4	JFK	2017-05-14

aid	loc	dt

Add 1 of 13 different encodings  
to each column

# Compression: Example

```
CREATE TABLE deep_dive (  
  aid INT ENCODE AZ64  
  ,loc CHAR(3) ENCODE BYTEDICT  
  ,dt DATE ENCODE RUNLENGTH  
);
```

aid	loc	dt

aid	loc	dt
1	SFO	2017-10-20
2	JFK	2017-10-20
3	SFO	2017-04-01
4	JFK	2017-05-14

More efficient compression is due to storing the same data type in the columnar architecture

Columns grow and shrink independently

Reduces storage requirements

Reduces I/O

# Data distribution

Distribution style is a table property which dictates how that table's data is distributed throughout the cluster

**KEY:** Value is hashed, same value goes to same location (slice)

**ALL:** Full table data goes to the first slice of every node

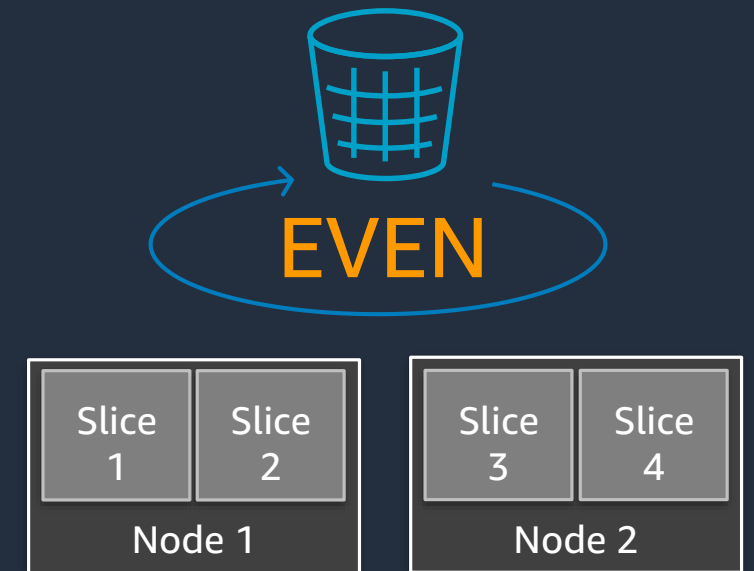
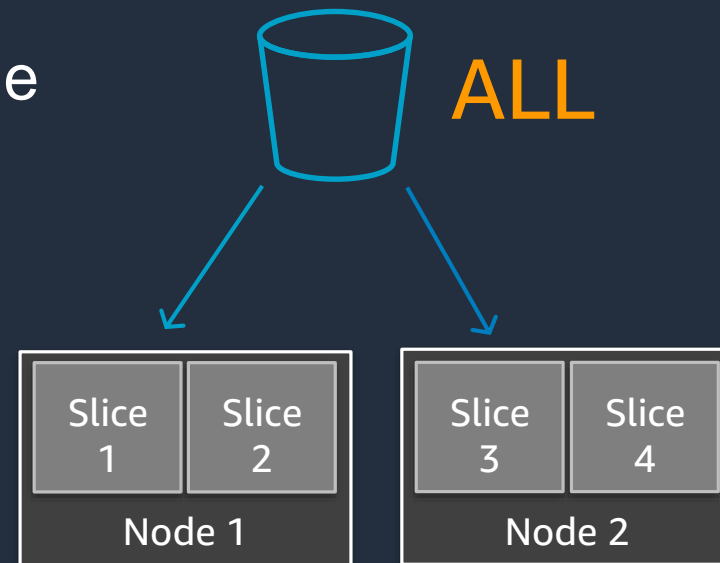
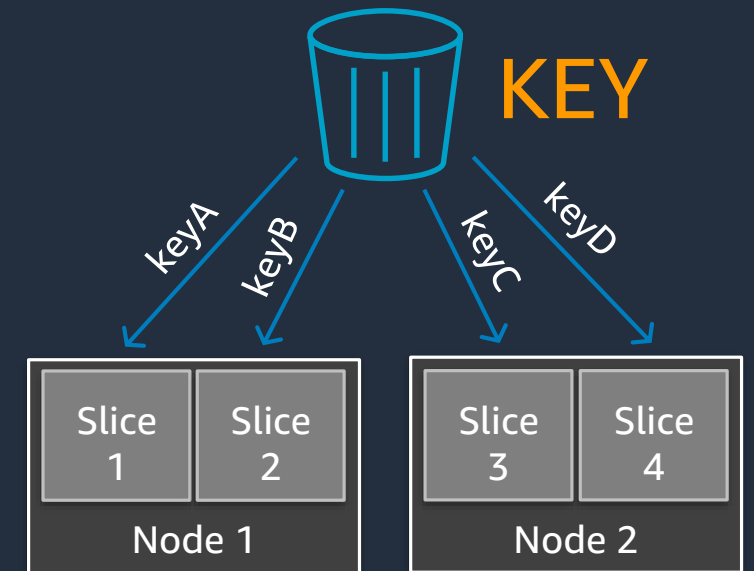
**EVEN:** Round robin

**AUTO:** Redshift chooses the distribution style

## Goals

Distribute data evenly for parallel processing

Minimize data movement during query processing



# Sort Keys

## Goal

Make queries run faster by increasing the effectiveness of zone maps and reducing I/O

## Impact

Enables range-restricted scans to prune blocks by leveraging zone maps

---

Achieved with the table property **SORTKEY** defined on one or more columns

Optimal sort key is dependent on:

- Query patterns

- Business requirements

- Data profile

# (AUTO) VACUUM

The VACUUM process runs either manually or automatically in the background

## Goals

VACUUM will remove rows that are marked as deleted

VACUUM will globally sort tables

- For tables with a sort key, ingestion operations will locally sort new data and write it into the unsorted region

# (AUTO) ANALYZE

- The ANALYZE process collects table statistics for optimal query planning
- In the vast majority of cases **AUTO ANALYZE** automatically handles statistics gathering

# Workload Management (WLM) modes

## Manual WLM

- Define different queues based on user groups or query groups
- Define how many concurrent queries can run within each queue
- Allocate a certain percentage of the cluster memory to each queue

## Auto WLM

- Define different queues based on user groups or query groups
- Redshift will determine the number of concurrent queries to run at any given point in time
- Redshift will also decide how much memory to allocate for each query based on the need.



# Why is Redshift your modern Data warehouse?

# Performance at Scale

# RA3 nodes with managed storage

## SCALE COMPUTE AND STORAGE INDEPENDENTLY



Managed storage

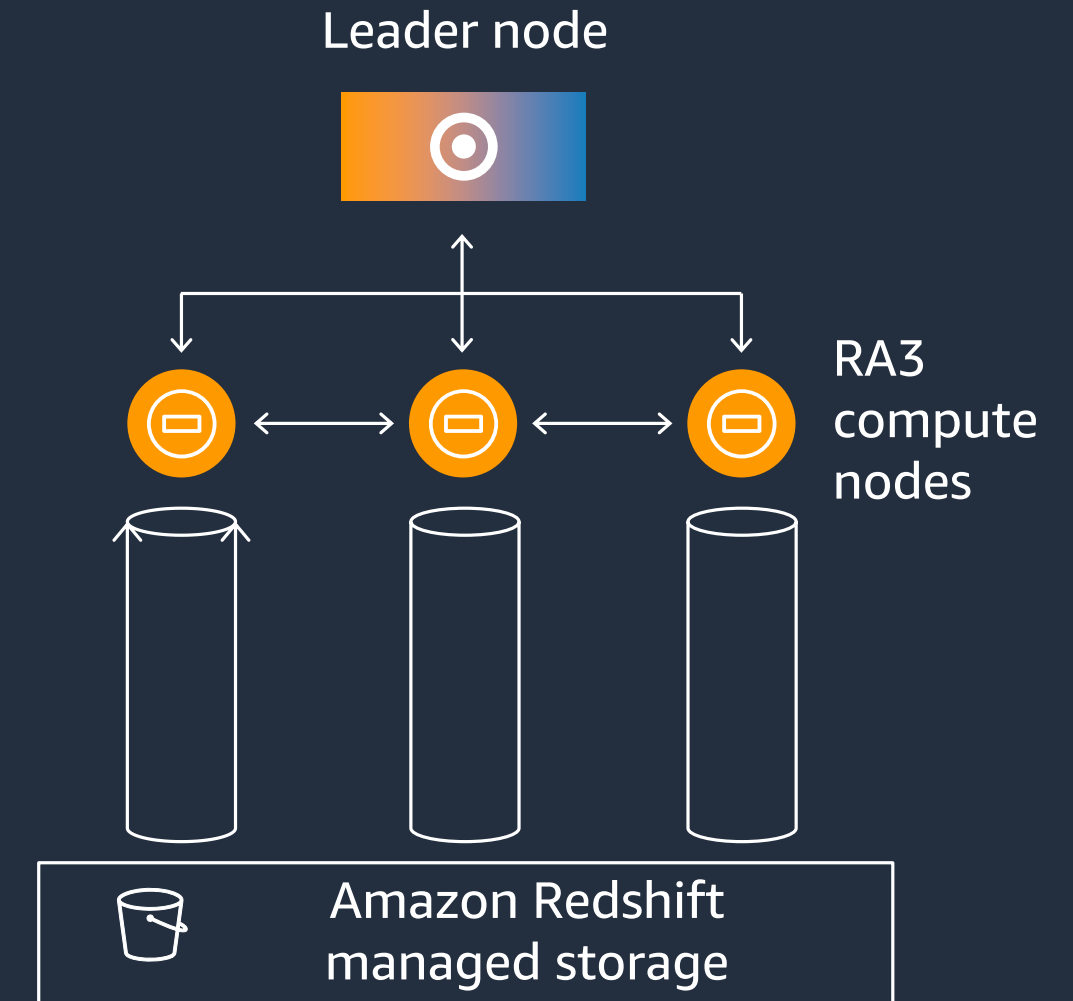


Large high-speed cache



High-bandwidth networking

- ▶ Size of data warehouse only based on steady state compute needs
- ▶ Scale and pay independently for compute and storage
- ▶ Automatic, no changes to any workflows, no need to manage storage



# Redshift Node types

## Amazon Redshift analytics—RA3

- Amazon Redshift Managed Storage (RMS)—Solid-state disks + Amazon S3

## Dense compute—DC2

- Solid-state disks

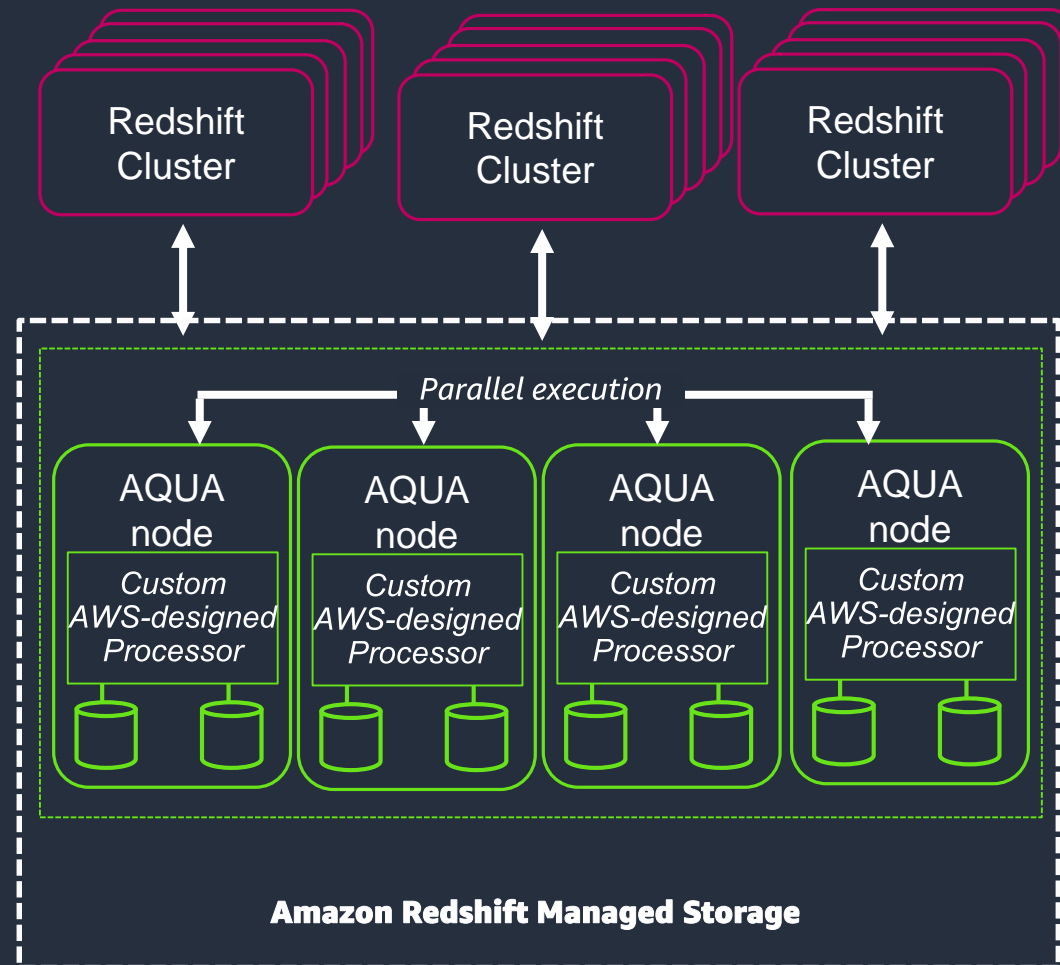
## Dense storage—DS2

- Magnetic disks

Instance type	Disk type	Size	Memory(GiB)	vCPUs	Slices
RA3.xlplus	RMS	Upto 32 TB RMS	32	4	2
RA3 4xlarge	RMS	Upto 128 TB RMS	96	12	4
RA3 16xlarge	RMS	Upto 128 TB RMS	384	48	16
DC2 large	SSD	160 GB	15	2	2
DC2 8xlarge	SSD	2.56 TB	244	32	16
DS2 xlarge	Magnetic	2 TB	31	4	2
DS2 8xlarge	Magnetic	16 TB	244	36	16



# AQUA (Advanced Query Accelerator)



New distributed & hardware-accelerated processing layer

With AQUA, Amazon Redshift is multiple times **faster** than any other cloud data warehouse, no extra cost

AQUA Nodes with custom AWS-designed analytics processors to make operations (compression, encryption, filtering, and aggregations) faster than traditional CPUs

*Available with RA3 and no code changes required*

# Data sharing builds on Redshift managed storage

HIGH PERFORMANCE DATA ACCESS WHILE PRESERVING WORKLOAD ISOLATION

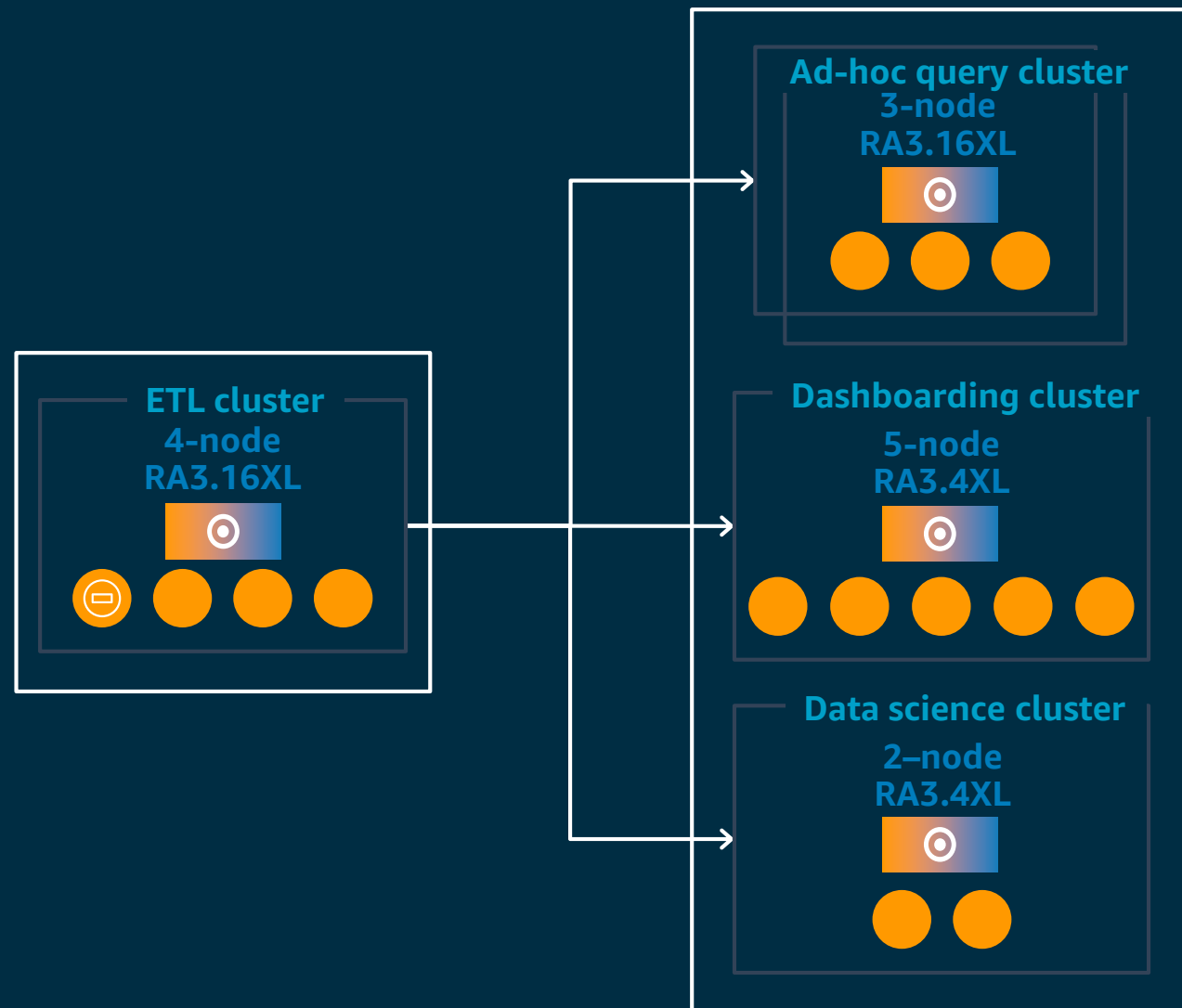


Producer pays for Amazon Redshift managed storage and consumers pay for consumer cluster

Workloads accessing shared data are isolated from each other and the producer

# Data sharing use cases (1)

## SEPARATE ETL VS BI WORKLOADS



Rapidly onboard new analytics workloads

---

Size and scale individual workloads according to their performance requirements

---

Provide workload isolation while sharing common datasets

---

Offer chargeback for individual workloads

---

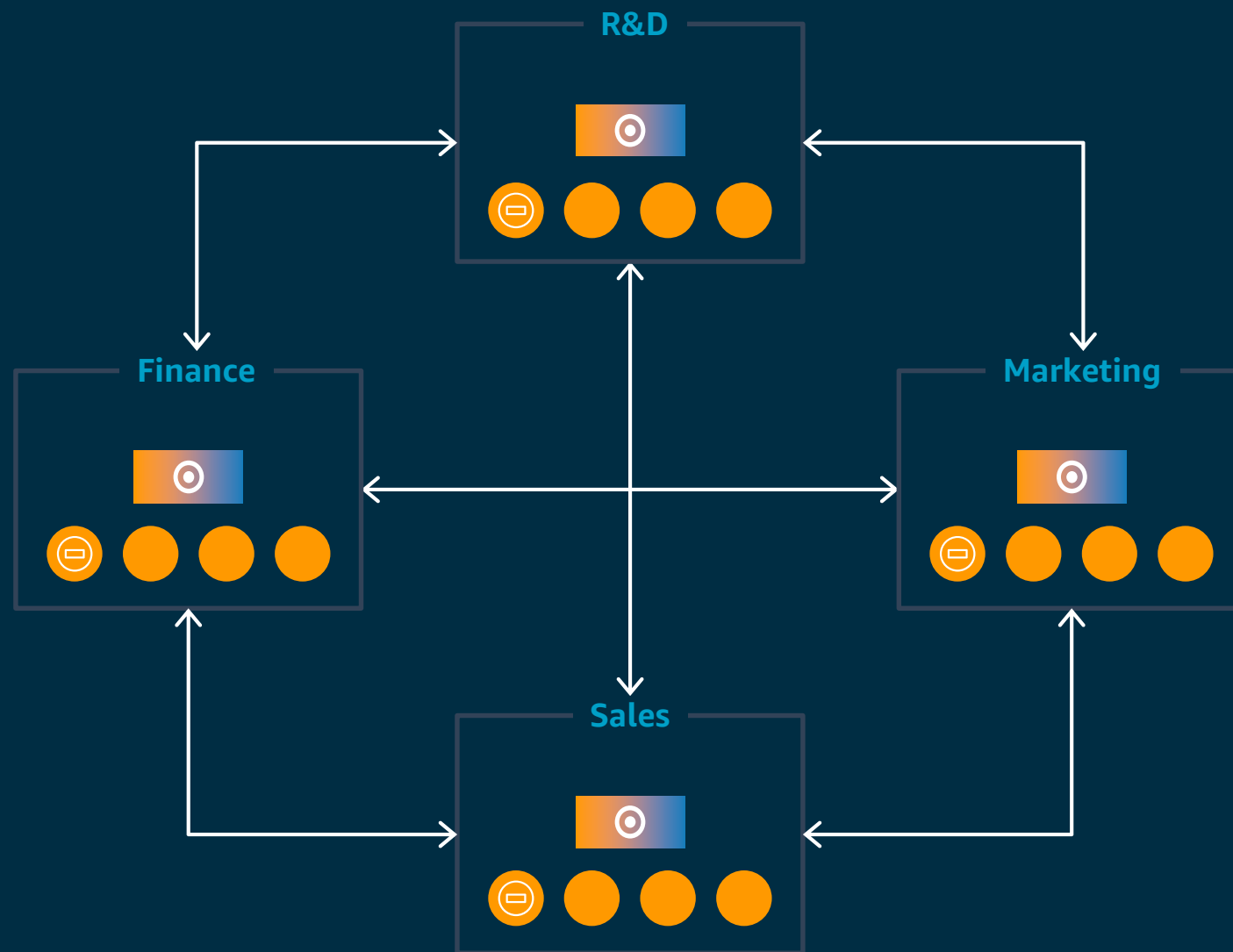
Pause / resume consumer clusters as necessary

---



# Data sharing use cases (2)

## CROSS-GROUP COLLABORATION



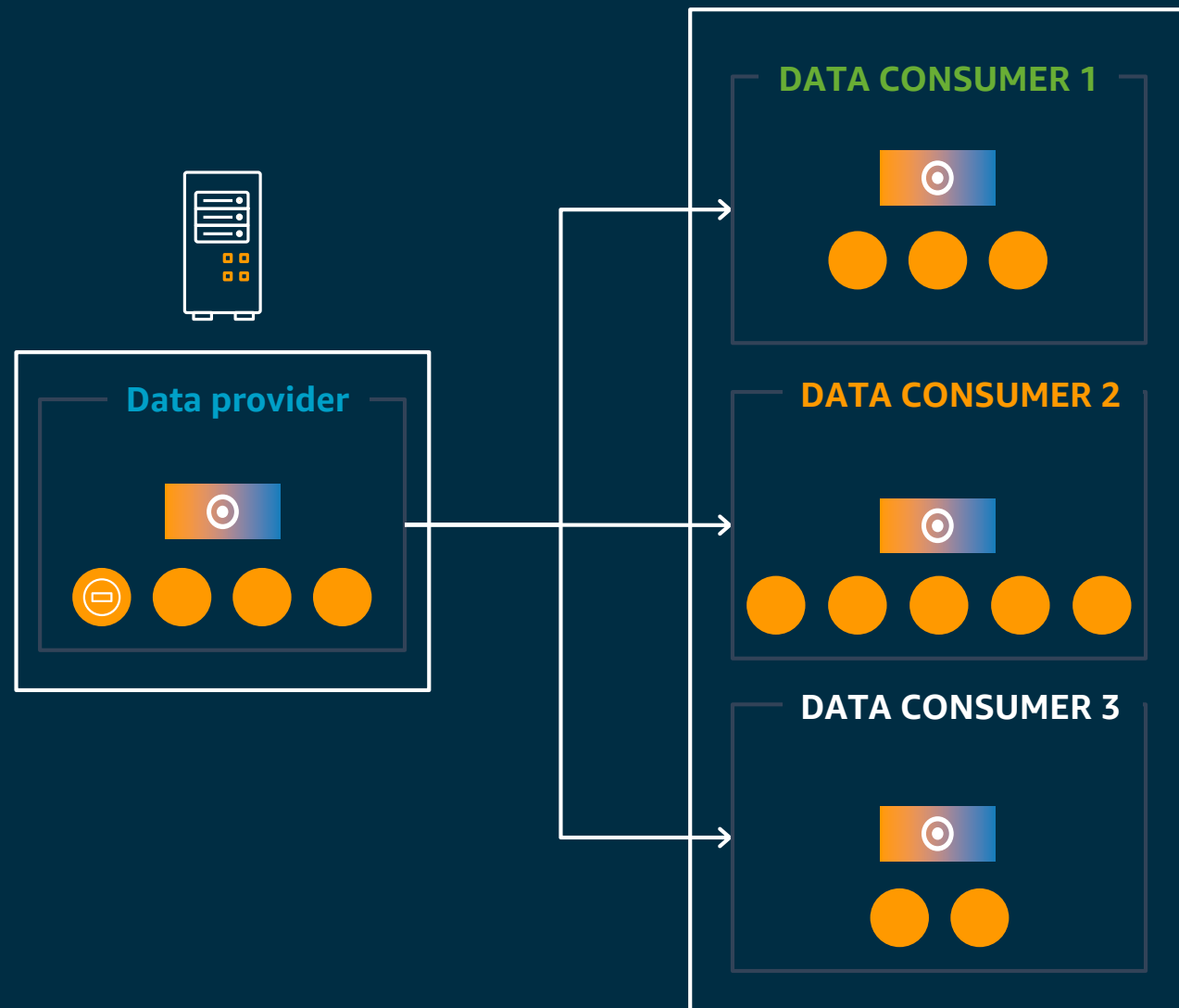
Seamlessly collaborate across business groups for broader analytics and data science and analyze cross-product impact

Eliminate compliance concerns with moving data

Each Amazon Redshift cluster can be producer and consumer of the data

# Data sharing use cases (3)

## DELIVER DATA AS A SERVICE



Offer data and analytics as a service within and across organizations and with external parties

---

Securely share live data with Amazon Redshift clusters in same or different AWS accounts

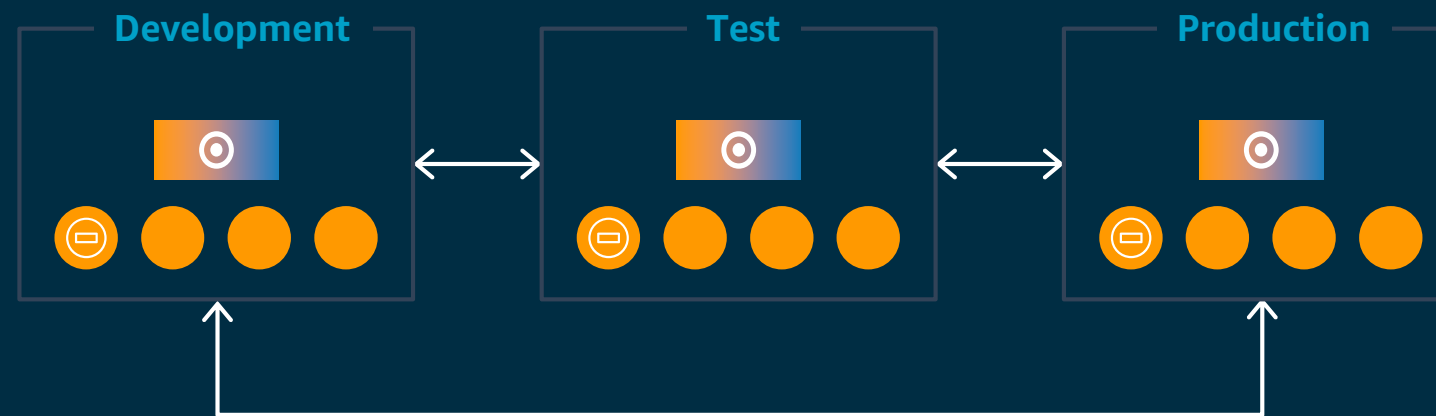
---

Monitor / track usage of the data and retain control of the data sets

---

# Data sharing use cases (4)

## SHARE DATA BETWEEN ENVIRONMENTS



Improve agility of teams by sharing data between **development, test, and production** environments at any granularity

# Data Sharing in Action

## - Demo

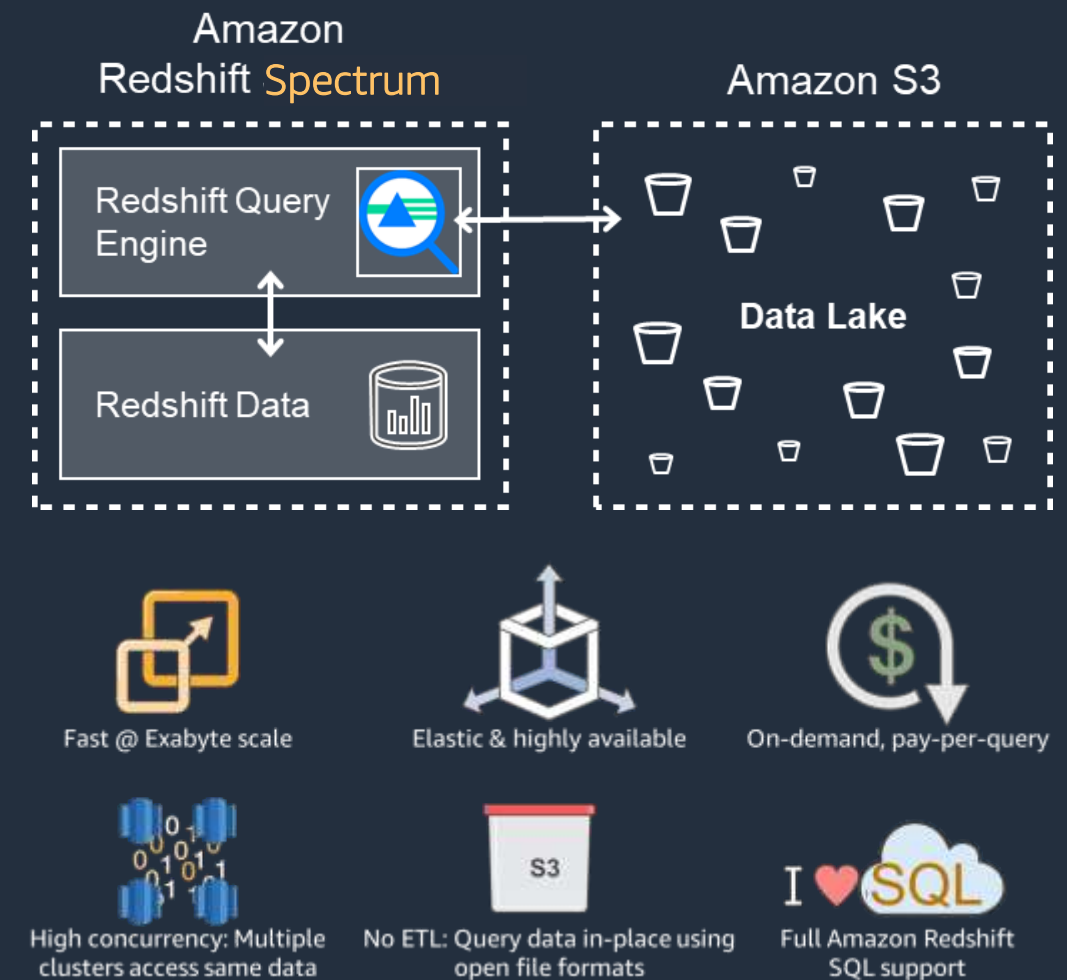
# Redshift Spectrum

Redshift Spectrum is a feature of Redshift that allows Redshift SQL queries to reference external data on Amazon S3 as they would any other table in Amazon Redshift

## Benefits

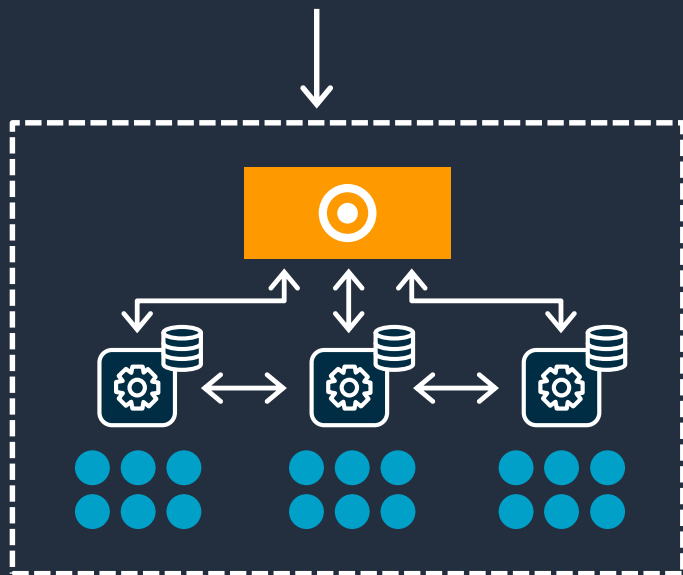
- Enables the Lake House pattern out-of-the-box
- Allows for querying of potentially exabytes of data in an S3 data lake from within Amazon Redshift
- Data is queried in-place, so no loading of data into your Redshift cluster is required
- Keeps your data warehouse lean by ingesting warm data locally while keeping other data in the data lake within reach
- Powered by a separate fleet of powerful Amazon Redshift Spectrum nodes

Run SQL queries directly against data in S3 using thousands of nodes



# Concurrency scaling

Amazon Redshift automatically adds transient clusters, in seconds, to serve sudden spike in concurrent requests with consistently fast performance



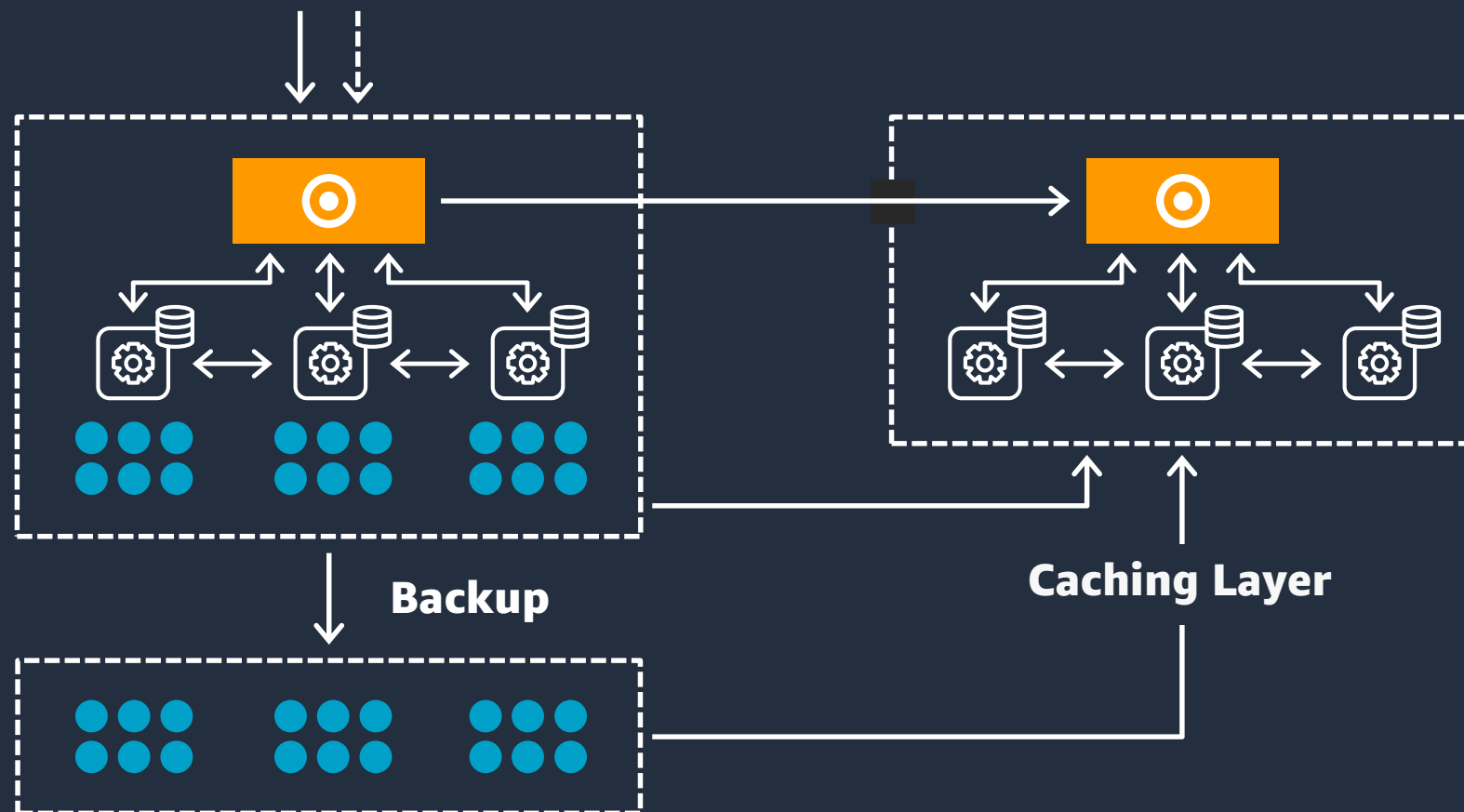
For every 24 hours that your main cluster is in use, you accrue a one-hour credit for Concurrency Scaling. This means that Concurrency Scaling is free for > 97% of customers.

## How it works:

- 1 All queries go to the leader node, user only sees less wait for queries
- 2 When queries in designated WLM queue begin queuing, Amazon Redshift automatically routes them to the new clusters, enabling Concurrency Scaling automatically
- 3 Amazon Redshift automatically spins up a new cluster, processes waiting queries and automatically shuts down the Concurrency Scaling cluster

# Concurrency scaling

Amazon Redshift automatically adds transient clusters, in seconds, to serve sudden spike in concurrent requests with consistently fast performance



For every 24 hours that your main cluster is in use, you accrue a one-hour credit for Concurrency Scaling. This means that Concurrency Scaling is free for > 97% of customers.

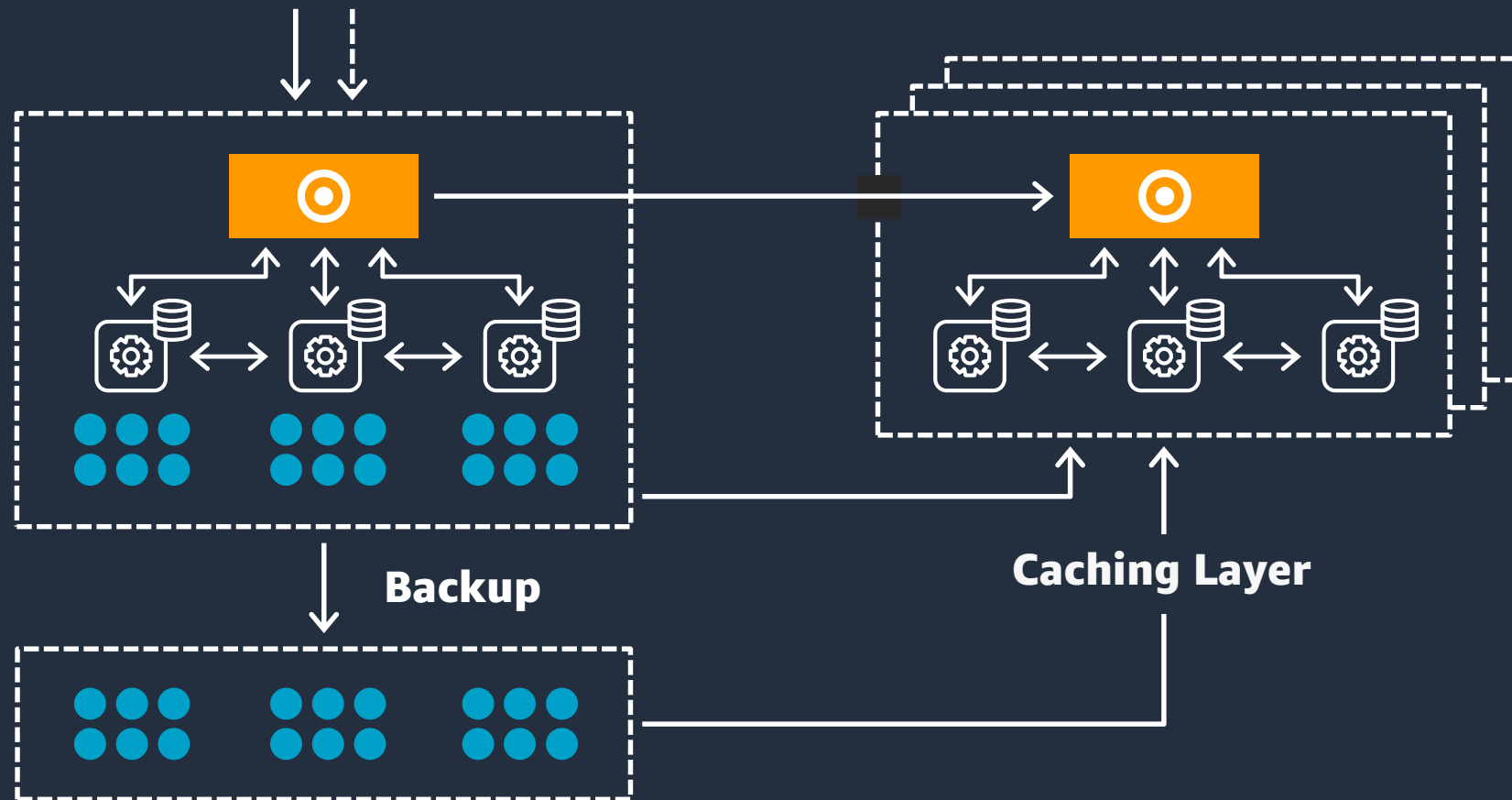
## How it works:

- 1 All queries go to the leader node, user only sees less wait for queries
- 2 When queries in designated WLM queue begin queuing, Amazon Redshift automatically routes them to the new clusters, enabling Concurrency Scaling automatically
- 3 Amazon Redshift automatically spins up a new cluster, processes waiting queries and automatically shuts down the Concurrency Scaling cluster



# Concurrency scaling

Amazon Redshift automatically adds transient clusters, in seconds, to serve sudden spike in concurrent requests with consistently fast performance



For every 24 hours that your main cluster is in use, you accrue a one-hour credit for Concurrency Scaling. This means that Concurrency Scaling is free for > 97% of customers.

## How it works:

- 1 All queries go to the leader node, user only sees less wait for queries
- 2 When queries in designated WLM queue begin queuing, Amazon Redshift automatically routes them to the new clusters, enabling Concurrency Scaling automatically
- 3 Amazon Redshift automatically spins up a new cluster, processes waiting queries and automatically shuts down the Concurrency Scaling cluster

# Resizing Amazon Redshift

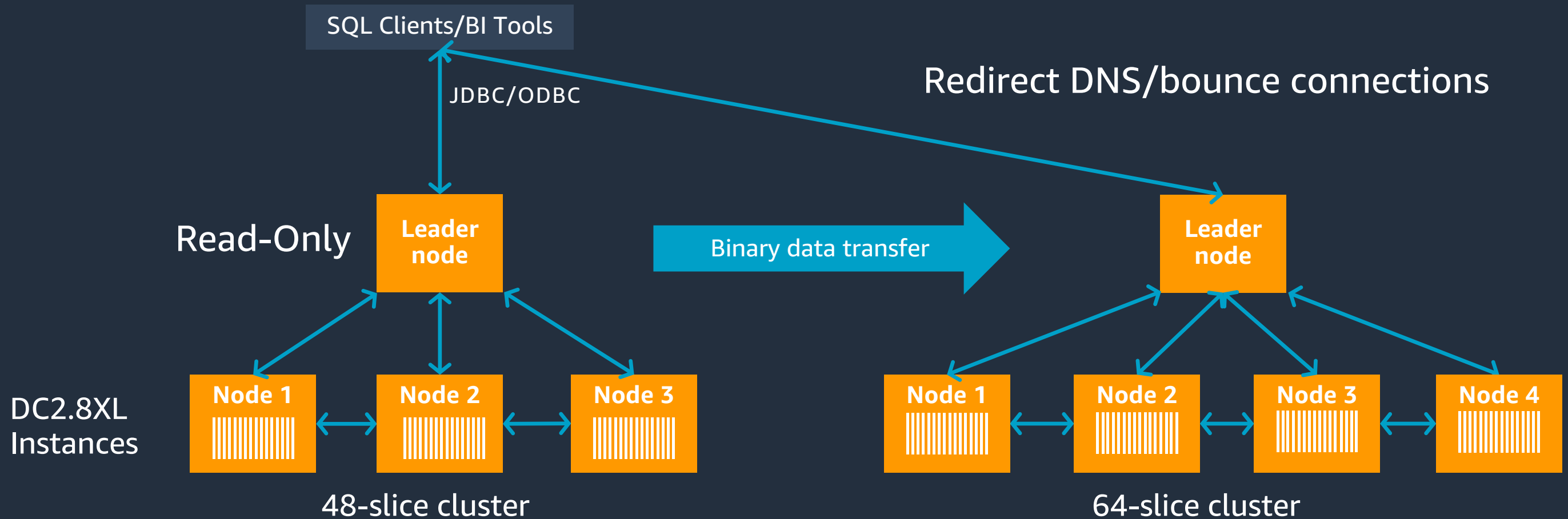
## Classic resize

- Data is transferred from old cluster to new cluster (within hours)

## Elastic resize

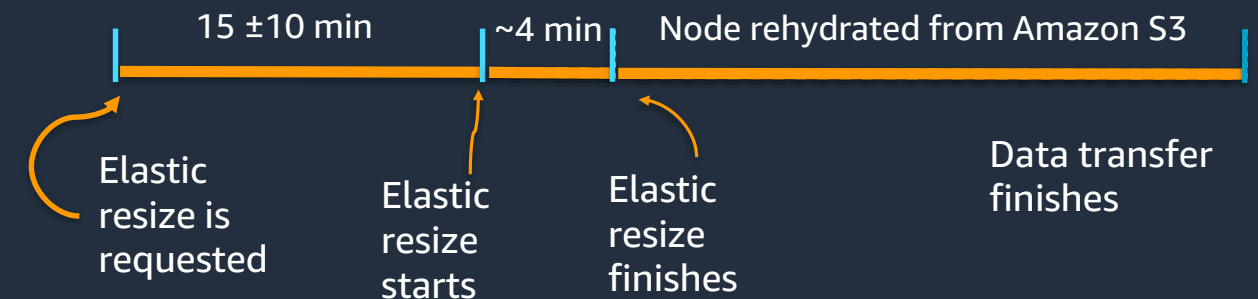
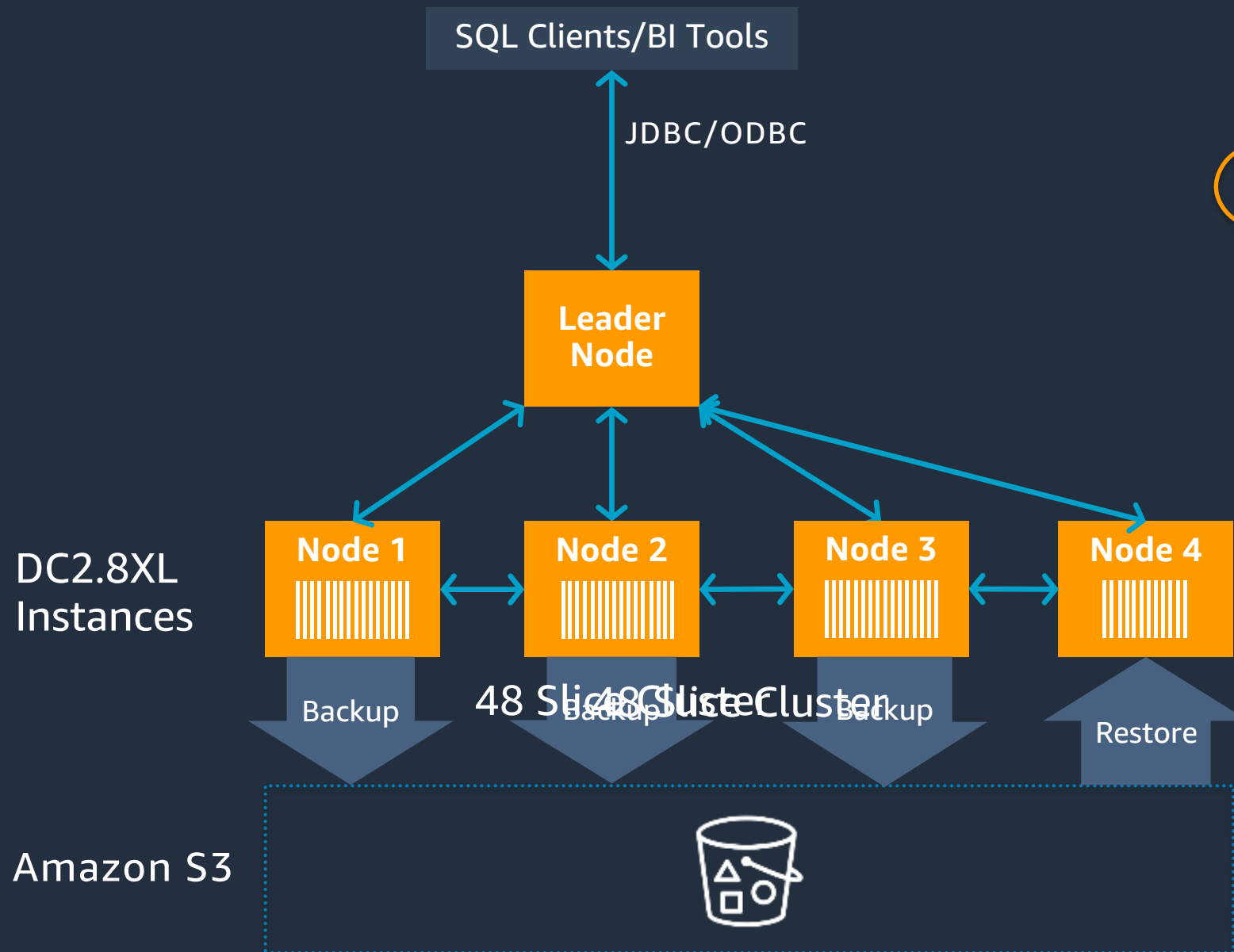
- Nodes are added/removed to/from existing cluster (within minutes)

# Classic resize



- Source cluster is placed into read-only mode during resize
- All data is copied and redistributed on the target cluster
- Allows for changing node types

# Elastic resize



- ~~After the site fully available nodes, a new node is added~~ take  
 are as fast as existing nodes in the background
- S3 and provision the new node(s)
- ~~Hotfixes are used for corrections~~ are put
- ~~Cluster~~ is fully available for reads
- and writes
- Some queries within transactions  
 maybe rolled back

# When to use what kind of scaling?

Scenario	Concurrency Scaling	Elastic Resize	Classic Resize
Unpredictable spiky Workload	✓		
Scale Up and down for known spikes		✓	
Permanently scaling out or scaling in			✓

# Auto-Everything

# Automated Performance Tuning

ML-BASED OPTIMIZATIONS TO GET  
STARTED EASILY AND GET THE FASTEST  
PERFORMANCE QUICKLY

Automates physical data design  
and optimization

---

Optimizes for peak performance  
as data and workloads scale

---

Leverages machine learning to adapt  
to shifting workloads

---

Automated performance tuning

---



Automatic  
vacuum delete



Automatic  
distribution keys



Automatic  
sort keys



Auto workload  
manager



Automatic  
table sort



MV auto-refresh  
and rewrite

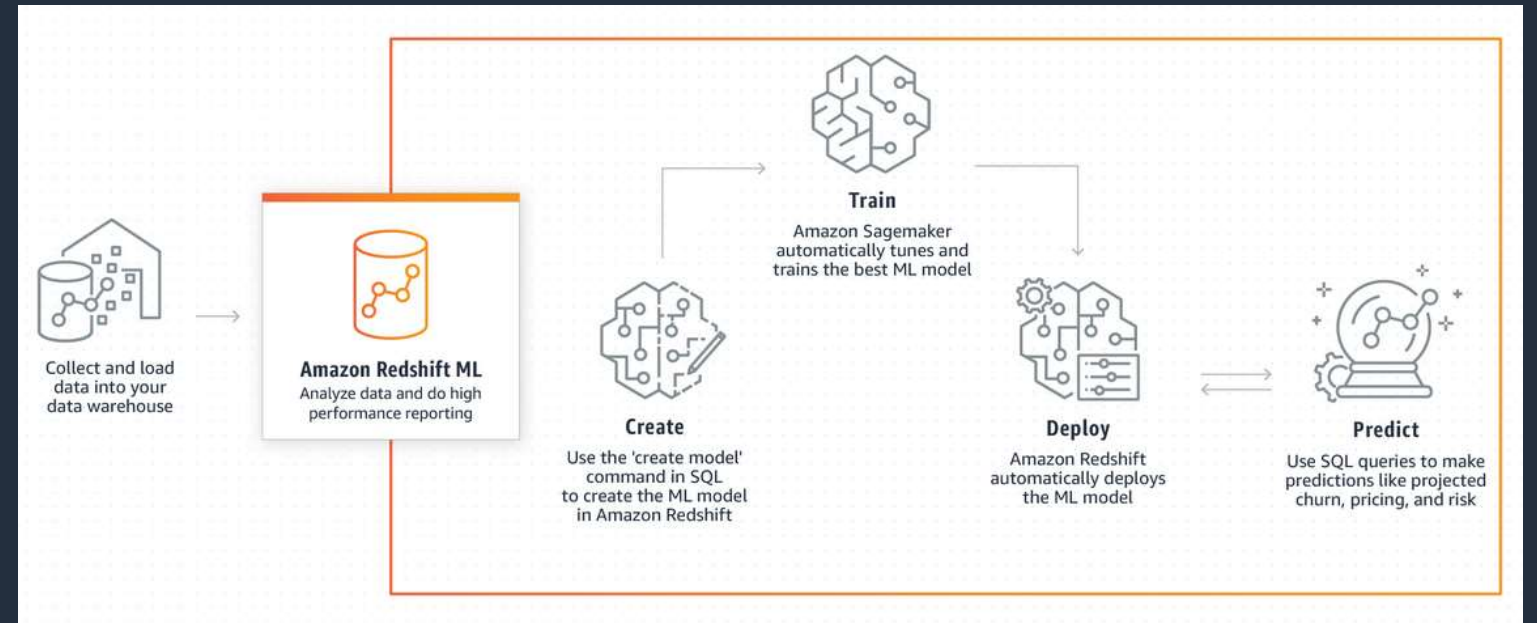
# Advanced Analytics Capabilities



# Amazon Redshift ML

EASILY CREATE AND TRAIN ML MODELS USING SQL QUERIES WITH AMAZON SAGEMAKER

- ✓ Use case: Product recommendations, fraud prevention, reduce customer churn
- ✓ Train and apply ML models using SQL
- ✓ From fully automated training to partially or fully guided training
- ✓ Automatic pre-processing, creation, training, deployment of your model
- ✓ Deploy inference models locally in Amazon Redshift
- ✓ Run an inference as invoking a user-defined function as part of SQL statements



```
CREATE MODEL customer_churn
FROM (SELECT c.age, c.zip, c.monthly_spend, c.monthly_cases,
c.active FROM customer_info_table c)
TARGET c.active
FUNCTION predict_customer_churn
...;

SELECT n.id, n.firstName, n.lastName,
predict_customer_churn(n.age,c.zip,..)
AS activity_prediction
FROM new_customers n
WHERE n.marital_status = 'single'
...;
```

# Redshift ML in Action

## - Demo

# Native Semi Structured Data Support

New data type: **SUPER**

Easy, efficient, and powerful JSON processing

Fast row-oriented data ingestion

Fast column-oriented analytics with  
materialized views over SUPER/JSON

Access to schema-less nested data with  
easy-to-use SQL extensions powered  
by the PartiQL query language

id INTEGER	name SUPER	phones SUPER
1	{"given": "Jane", "family": "Doe"}	[{"type": "work", "num": "9255550100"}, {"type": "cell", "num": "6505550101"}]
2	{"given": "Richard", "family": "Roe"}	[{"type": "work", "num": "5105550102"}]

```
SELECT name.given AS firstname, ph.num  
FROM customers c, c.phones ph  
WHERE ph.type = 'cell';
```

```
firstname | num  
-----+-----  
"Jane"    | 6505550101
```

# Super Data type in Action

## - Demo

# Materialized Views

**SPEED UP QUERY PERFORMANCE BY ORDERS OF MAGNITUDE WITH PRECOMPUTED RESULTS**

Simplify and accelerate iterative and predictable workloads, such as ETL, BI/dashboarding queries

MVs can be based on one or more Amazon Redshift tables or external tables (Spectrum, Federated)

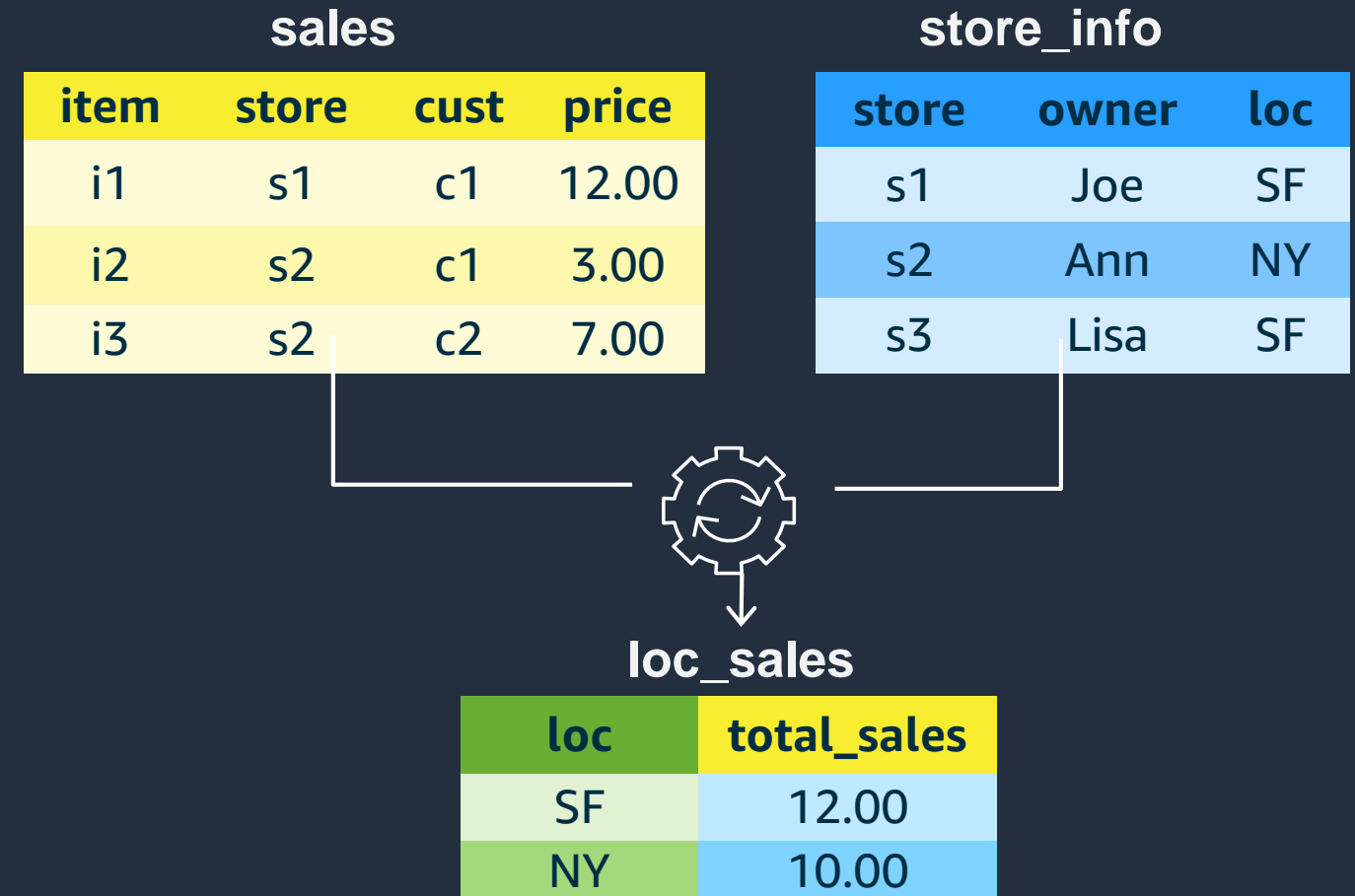
---

Efficient incremental maintenance

Scheduled, automatic, or manually timed refresh

---

Amazon Redshift auto query rewrite optimizes queries by replacing native tables with materialized views



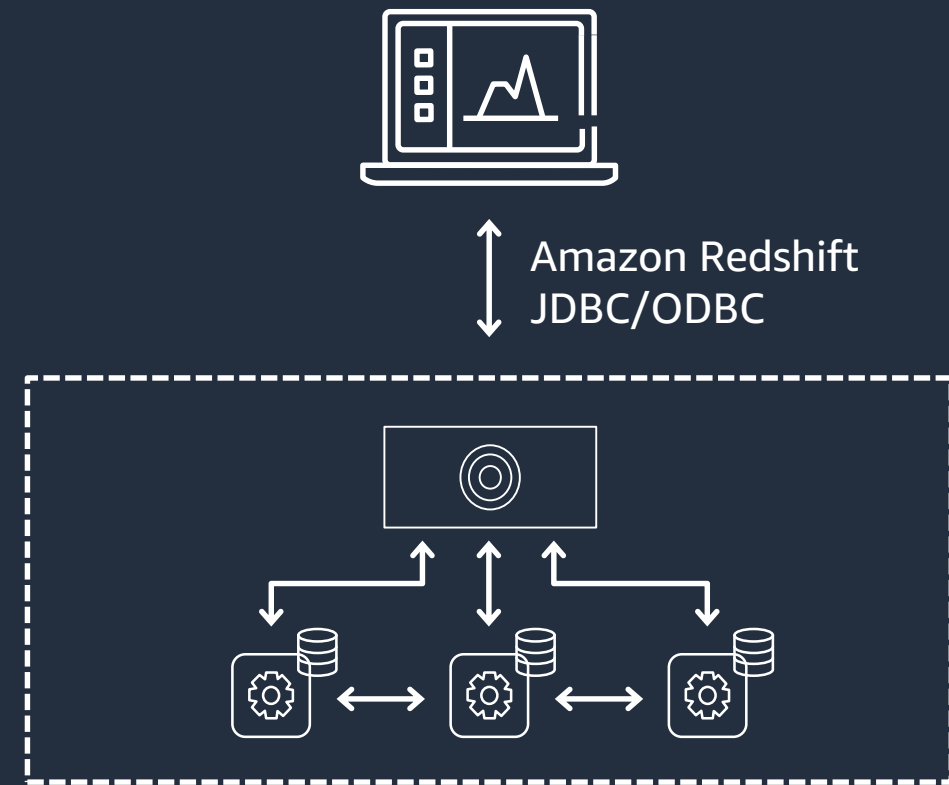
*"The Amazon Redshift materialized view auto query rewrite feature reduced dashboard load times from 8 minutes to just 500 ms. The best part is that this is completely transparent for Tableau and the business user."*

*Arman Nasrollahi, Home24*


# Amazon Redshift Federated Query

## UNIFIED ANALYTICS ACROSS DATABASES, DATA WAREHOUSE, AND DATA LAKE

- ✓ Use case: Integrate operational data with DW and data lake for real-time analytics
- ✓ Analytics on operational data without data movement and ETL delays
- ✓ Query and join data from one or more Amazon RDS and Aurora PostgreSQL databases
- ✓ Flexible and easy way to ingest data avoiding complex ETL pipelines
- ✓ Intelligent distribution of computation to remote sources to optimize performance
- ✓ **Preview** Amazon RDS and Aurora MySQL support



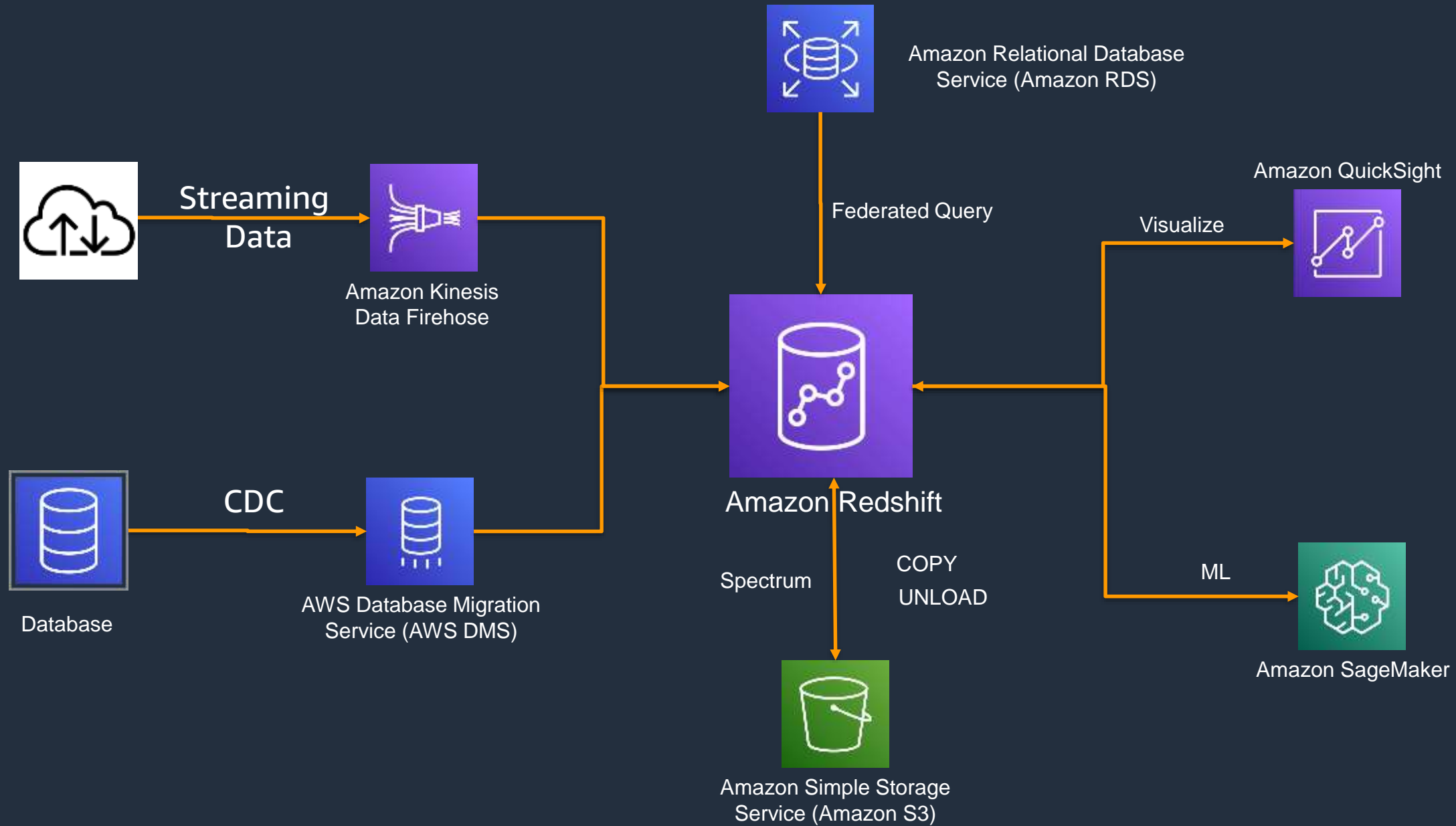
  
Amazon Aurora  
PostgreSQL,  
MySQL

  
Amazon S3  
data lake

  
Amazon RDS  
PostgreSQL,  
MySQL

# Integration with Analytics Services

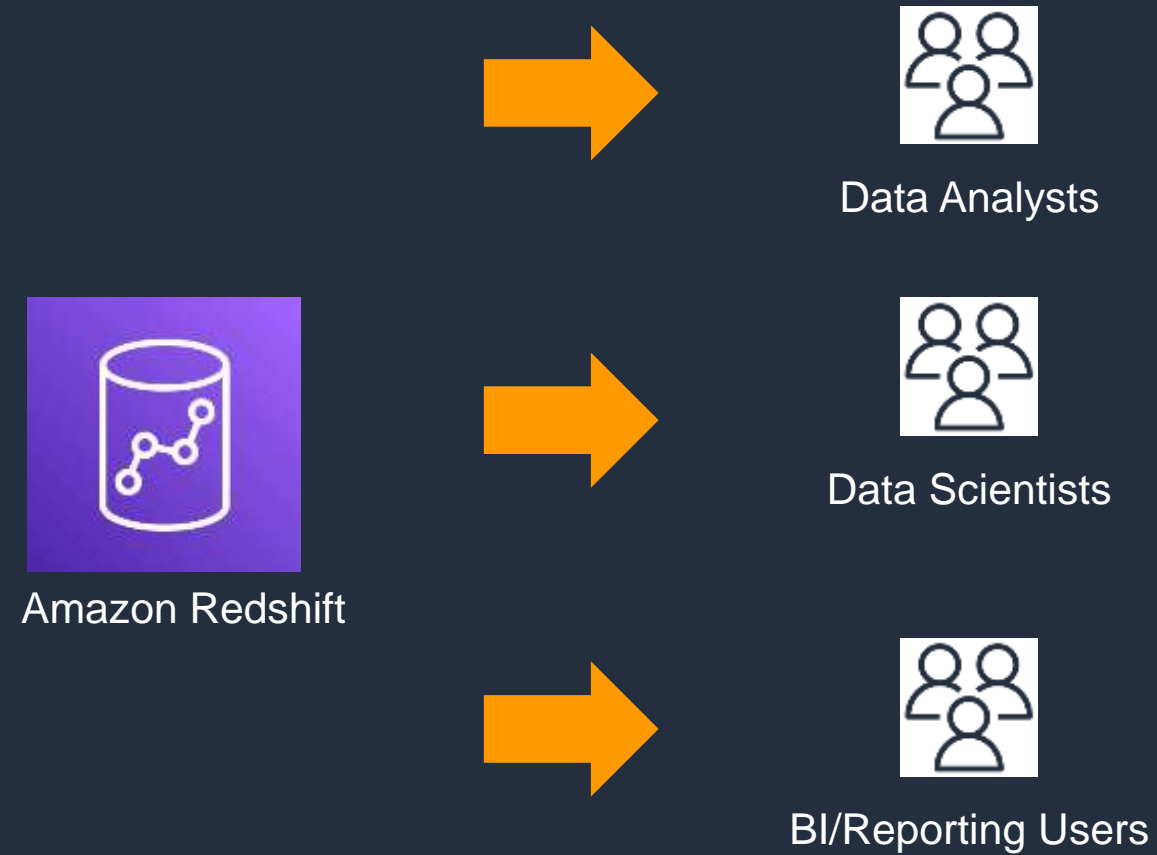
# Integration with Analytics Services





# Variety of Personas and Use cases

# Variety of Personas and Use Cases



# Customer Case studies

# Tens of thousands of customers process exabytes of data with Amazon Redshift daily



## NTT DOCOMO

Moved >10 PB of data from on-premises to cloud



## FOX Corp.

Taking a lake house approach with RA3 nodes and Amazon S3



## Dream11

Reporting for Finance and Product Teams for better decision making



## DSP Mutual Fund

Building a data platform using Amazon Redshift to democratize access to data



## Warner Bros. Games

Performance, scale, cost-effective



# Migration to Amazon Redshift

# AWS data migrations: broadest toolkit

AWS provides the broadest range of tools for easy, fast, and secure data movement to and from the AWS cloud



AWS Direct Connect



AWS Snowball



Amazon S3 Transfer  
Acceleration



AWS Storage  
Gateway



**AWS Schema Conversion Tool (SCT)**

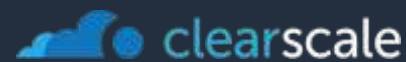


**AWS Database Migration Service (DMS)**

# Migrate with AWS partners

AWS consulting partners offer a wide range of migration services to help you move your data warehouse to Amazon Redshift. **AWS Data Warehouse Migration Partners** provide support to accelerate moving a data warehouse to the cloud with proven best practices and resources. The **AWS Service Delivery Partners** have deep understanding of specific AWS services, follow best practices and have proven success delivering AWS services to customers.

This is not a complete list; to view all Amazon Redshift partners, visit <https://aws.amazon.com/redshift/partners/>



# Thank You