Microsoft

PolyBase vNext

Presenters

Travis Wright
Program Manager

Agenda

Background

Future of PolyBase

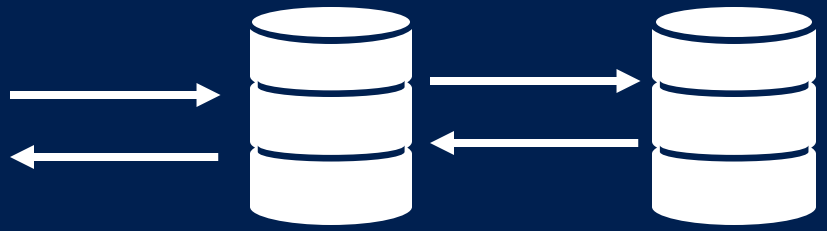
Data Integration is key to realizing business value

Different forms of data are better suited for different types of database engines

Data inertia makes consolidation difficult

Data Movement

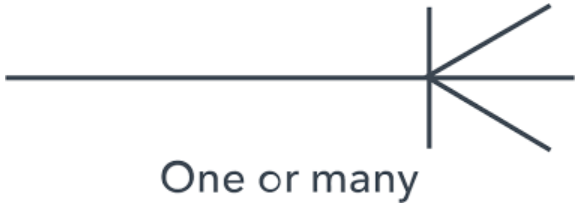
Data Virtualization



Data Movement Scenarios

$$(1+x)^n = 1 + \frac{nx}{1!} + \frac{n(n-1)x^2}{2!} + \cdots$$

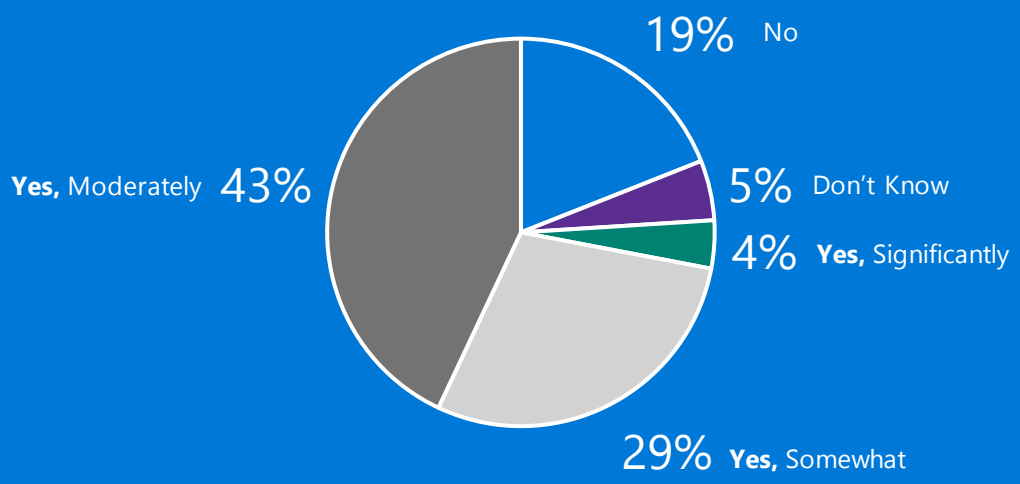
Aggregate/transform one time. Query result many times.



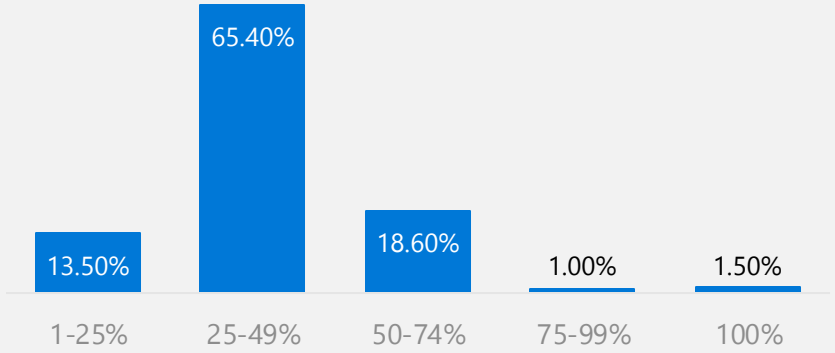
Joining data sets from multiple sources ***frequently*** and the performance needs to be ***super fast***.

Data Movement Challenges

More than 3/4 of Respondents Say That Untimely Data Has Inhibited Business Opportunities

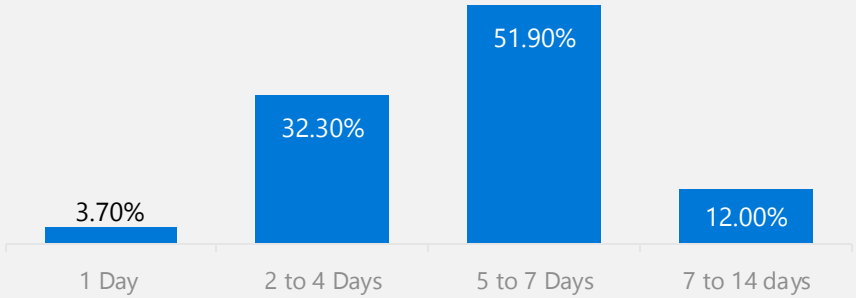


Large Volumes of Data Movement



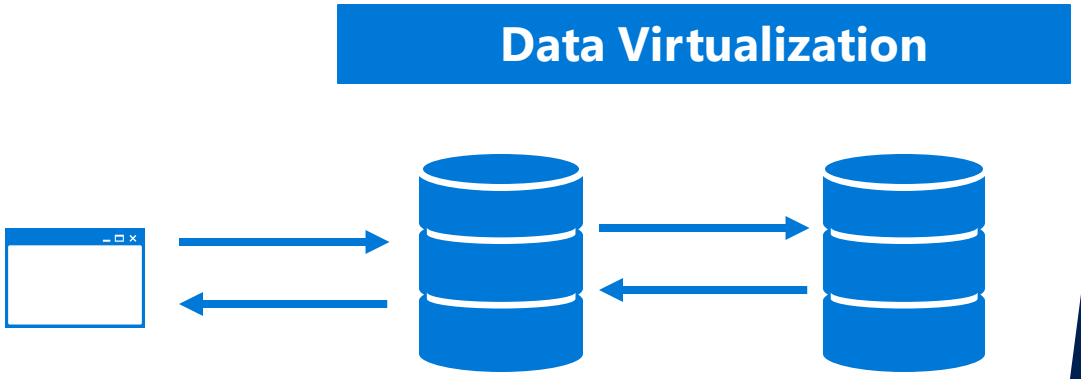
What percent of data is moved between transactional and analytical systems via ETL?

Data Movement is Slow



On average, for each ETL task, how old is the data by the time it reaches the analytical database?

Data Virtualization



Data virtualization integrates data from disparate sources, locations and formats, *without replicating or moving the data*, to create a single "virtual" data layer that delivers unified data services to support multiple applications and users.

Data Movement vs. Data Virtualization

	Development & Operations Costs	Time to Solution	Security	Data Freshness & Quality	Compliance
Data Movement	Costly to build and maintain ETL jobs. Duplicated data storage costs.	Takes time to build and run jobs.	Creating copies of the data makes it more vulnerable to hackers.	ETL pipelines make the data "stale". ETL introduces data errors.	Moving data in and out of compliance boundaries can cause data governance issues.
Data Virtualization	Reduced ongoing maintenance and change management. Minimize storage costs.	Rapid iterations and prototyping can be done.	Data is kept in one secure place minimizing the attack surface area.	Data being queried is always "fresh" and accurate from the source.	Data virtualization helps meet compliance requirements as there are less copies of data and movement.

Data Virtualization in SQL Server Today

Oracle

Sybase

Excel

DB2

SQL Server

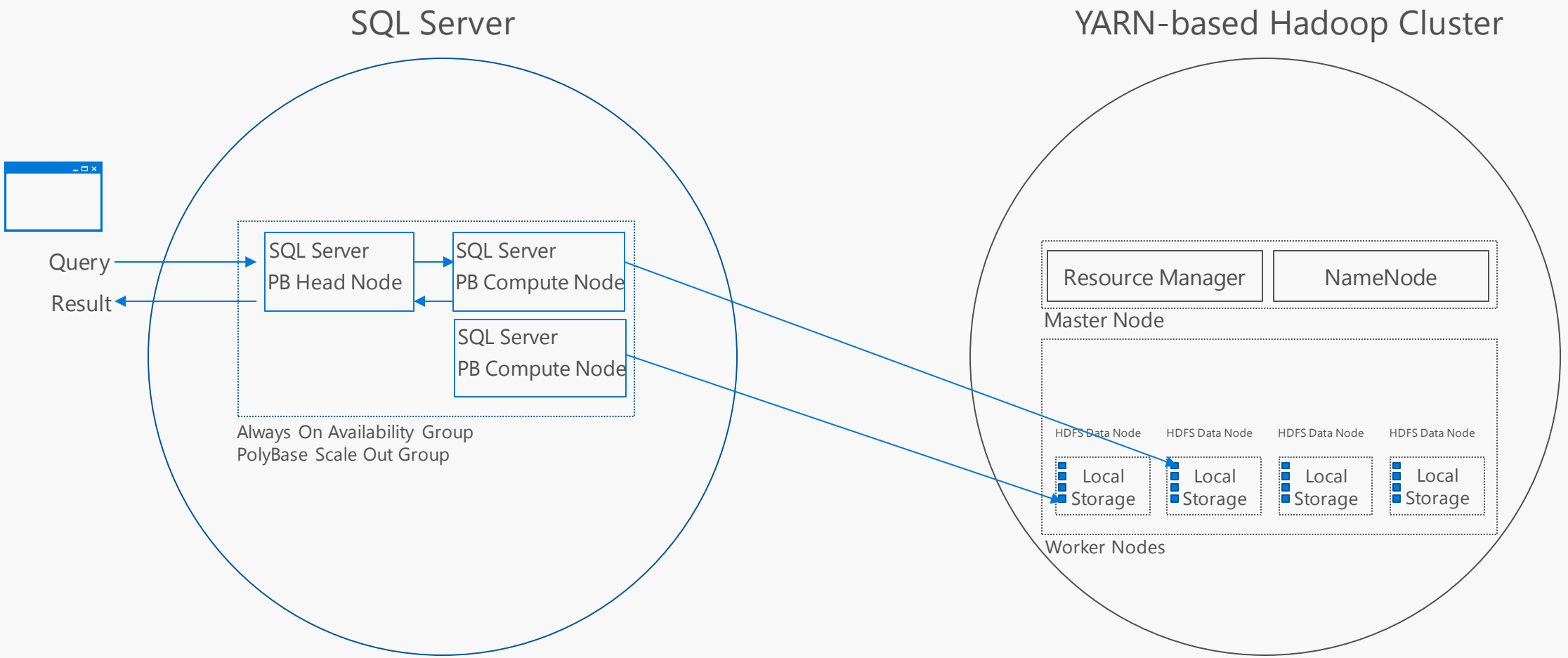
Integrate SQL Server with other databases with **Distributed Query**

No changes

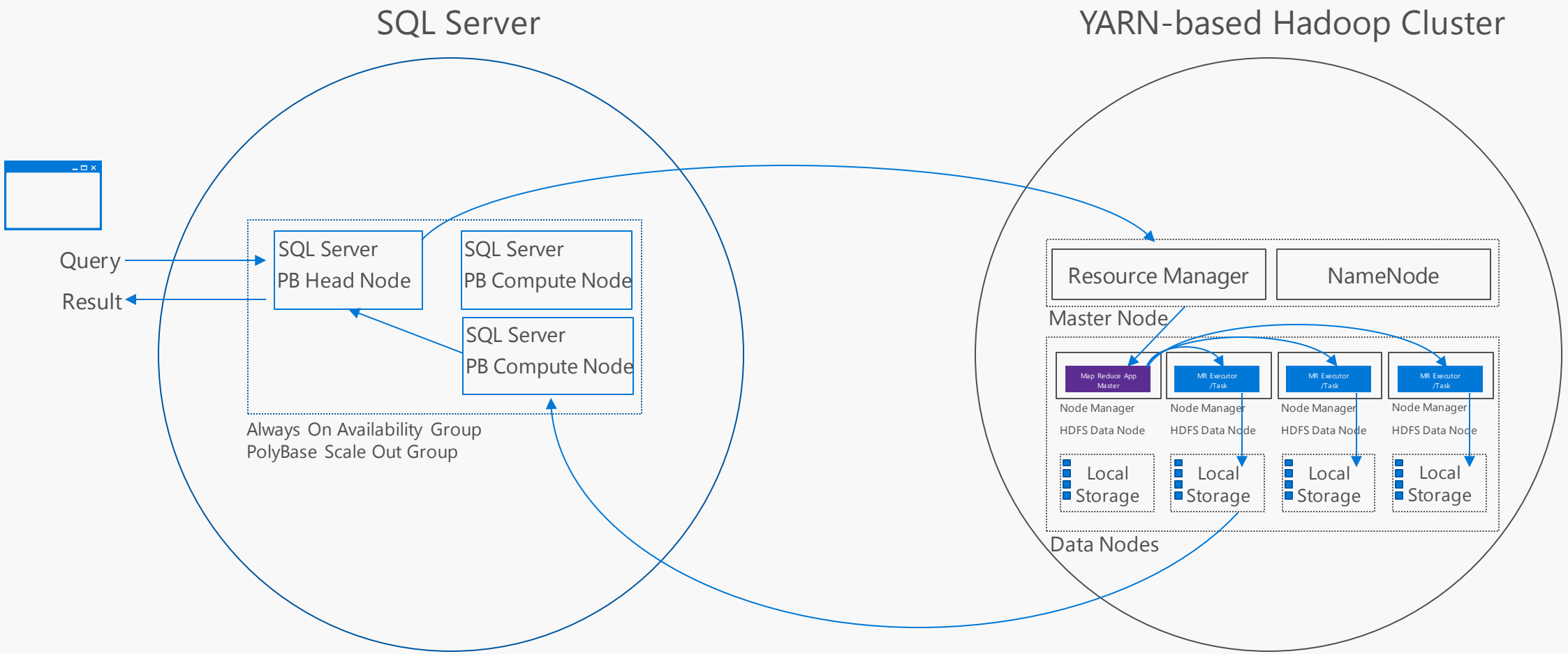
Connect your relational data to big data with **PolyBase**

Improvements

PolyBase in SQL Server 2016 – Stream Query



PolyBase in SQL Server 2016 – Push Down Query



Customer feedback

Customers love the *idea* of PolyBase but…

Security

No support for:

- Encryption Zones
- Apache Knox
- Ranger
- TLS/SS
- Active Directory authentication
- User-level auditing

Limited DBMS Integration

No support for:

- Oracle
- MongoDB
- Teradata
- SQL Server
- Generic ODBC
- OLEDB

Performance

Queries are too slow

- Time to spin up YARN containers
- Remote file access is slow
- Limited predicate pushdown
- Single threaded
- Converting data
- MapReduce is old

Metadata Integration

No integration with Hcatalog

No awareness of ORC/parquet file headers

Loading

No support for:

- JSON
- AVRO
- Extended chars
- Different date & time formats
- Evolving schemas

Business Model

- High upfront CapEx cost for hardware for scale out nodes + SQL Server licenses creates adoption friction and risk
- Perceived low value for price due to idle SQL Server and HW compute capacity

Project “Aris”

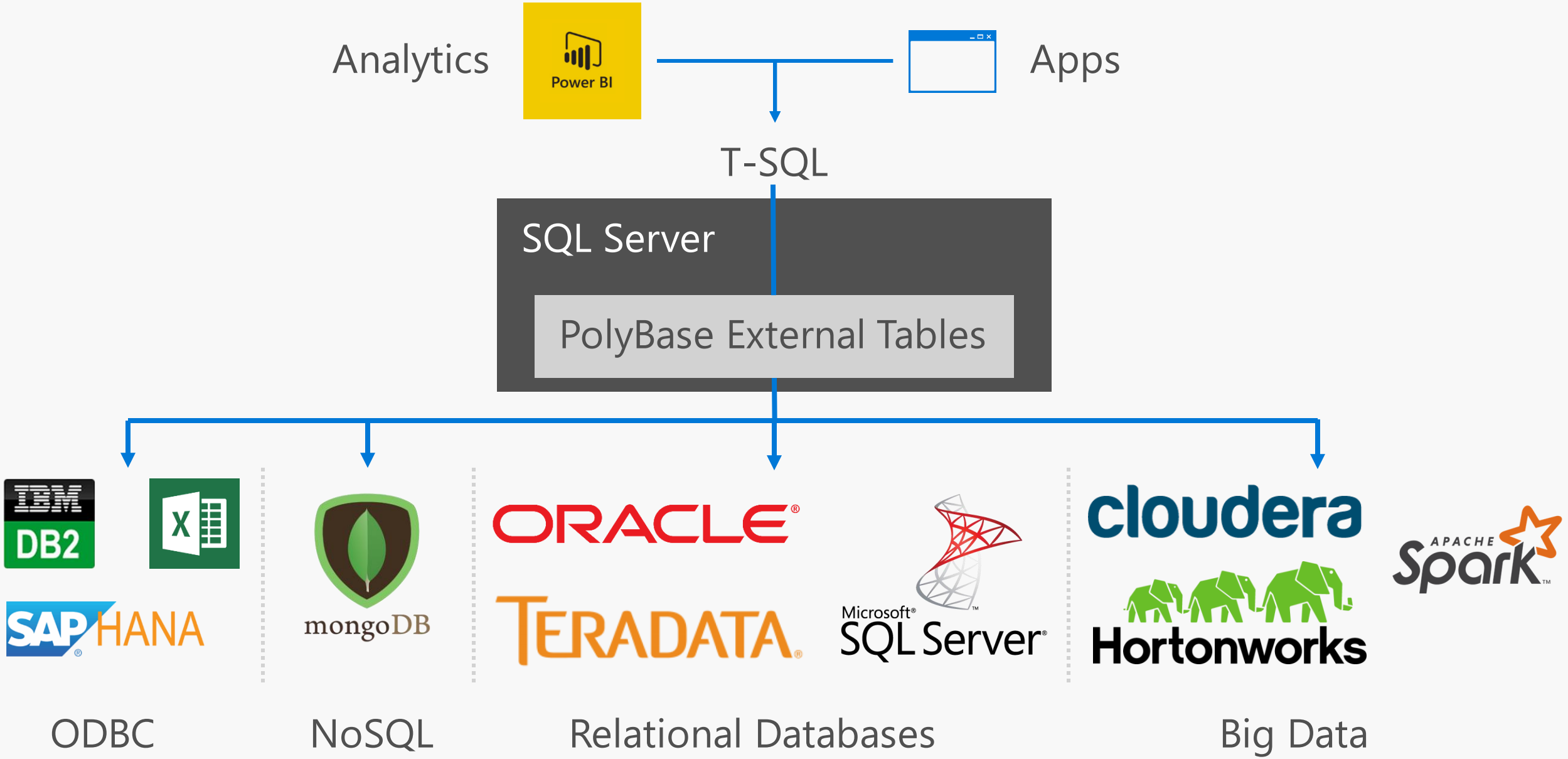
Goals

1. Bring SQL Server and Big Data Hadoop and Spark systems closer together.
   Based on customer feedback, rearchitect PolyBase to be faster, more secure, and bi-directionally integrated with Hadoop/Spark.

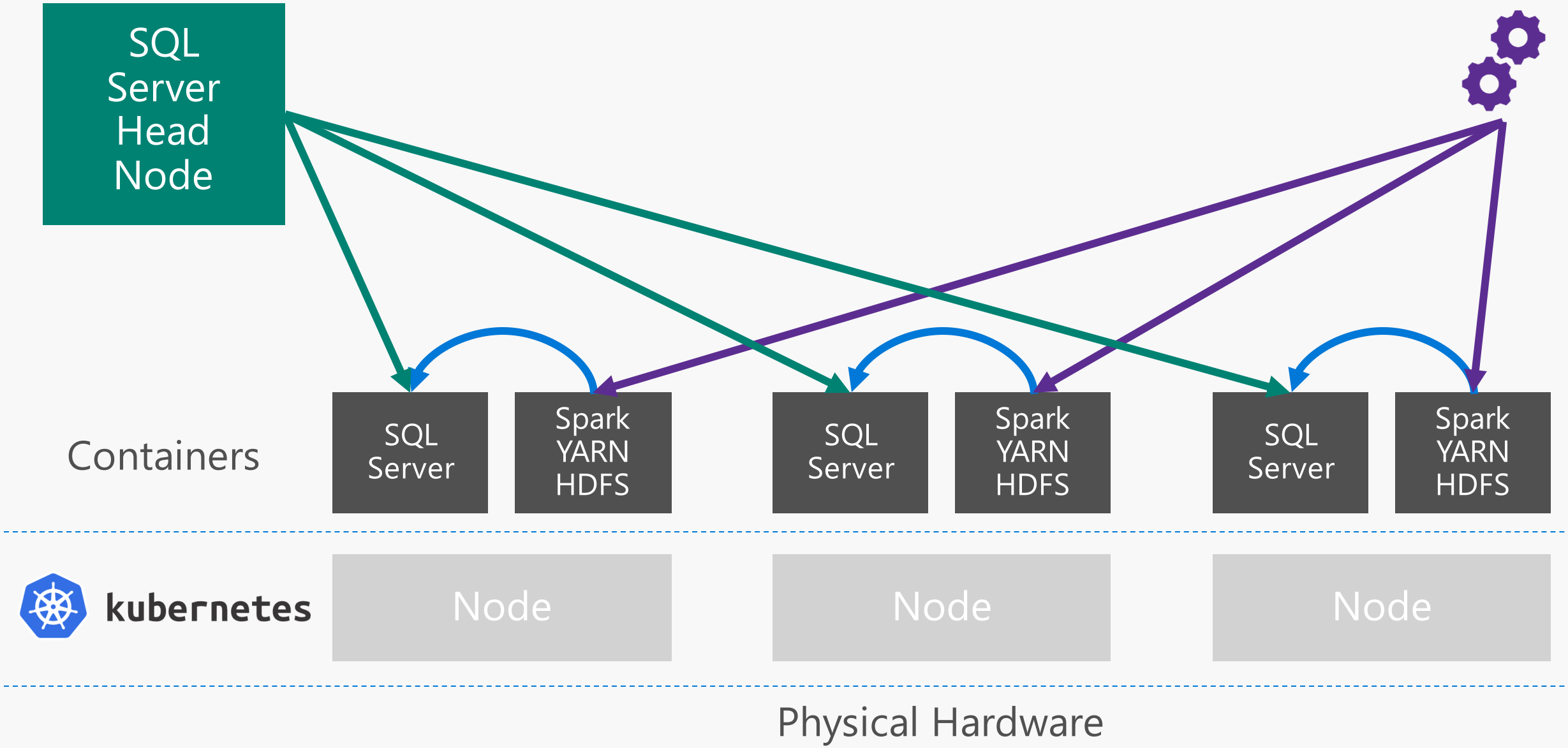
2. Improve the data virtualization experience between SQL Server and other DB engines
   Oracle, MongoDB, Teradata, SQL Server, and other DB engines via a generic ODBC connector.

3. Integrate existing distributed architecture technologies under PolyBase brand

SQL Server as a "Data Hub"



Phase 1: CTP1 – scale-out storage and query




CTP 1 “Demo”

Clone the GitHub repo with the deployment scripts in it

```
git clone https://github.com/annashres/sqleapvnext.git
```

Set environment variables for credentials to access the private Docker registry

export DOCKER_USERNAME=<your_user_name>
export DOCKER_PASSWORD=<your_password>
export DOCKER_EMAIL=<your_email>

Set environment variables for the passwords to be used within Aris

export MSSQL_SA_PASSWORD=<sa_password_of_sql_instances>
export MSSQL_IMPORT_PASSWORD=<import_account_password_of_sql_instances>
export MSSQL_EXTERNAL_PASSWORD=<external_account_password_of_sql_instances>

Deploy the compute cluster, Hadoop and head node containers to Kubernetes

deploy-**mssql**-**cluster**

Create a service for the SQL compute pool to be accessed from outside of

Create logical compute pool

```
USE high_value_data
GO
DECLARE @compute_pool_name NVARCHAR(max) = 'mssql-compute-pool'
DECLARE @cluster_node_count INT = 2

PRINT 'STEP 1: Initialize compute cluster'
--     sp_compute_pool_create will do the following:
--         1. setup up 'map' management
--         2. create credential for head node --> compute node authentication
--         3. create compute node databases and configure 'map'
--         4. create EXTERNAL DATASOURCE on head node for compute cluster
--
EXEC sp_compute_pool_create @compute_pool_name, @cluster_node_count
GO
```

Spark job to create schema

```
USE high_value_data
GO
PRINT 'STEP 2: Derive table schema from sample file (using Spark) and create SQL Server table schema'
--      sp_compute_pool_create_table will do the following:
--     1. Submit Resource Negotiator request to submit Spark job to derive T/SQL schema
--     2. Create TABLE on each compute node
--     3. Create EXTERNAL TABLE on the head node
--
DECLARE @compute_pool_name NVARCHAR(max) = 'mssql-compute-pool'
DECLARE @table_name NVARCHAR(max) = 'airlinedata'
DECLARE @sample_file_name NVARCHAR(max) = 'hdfs:///airlinedata/airlinedata_sample.csv'
EXEC sp_compute_pool_create_table @compute_pool_name, @table_name, @sample_file_name
GO
```

Start streaming job to ingest data

```
USE high_value_data
GO
PRINT 'STEP 3: Start/Stop Spark stream (Submit Spark job to Resource Negotiator)'
--  Submit Spark job to ingest data from specified folder (files) into compute
instances
--
DECLARE @compute_pool_name NVARCHAR(max) = 'mssql-compute-pool'
DECLARE @table_name NVARCHAR(max) = 'airlinedata'
DECLARE @source_folder NVARCHAR(max) = 'hdfs:///airlinedata'
EXEC sp_compute_pool_start_import @compute_pool_name, @table_name,
@source_folder;
GO
```

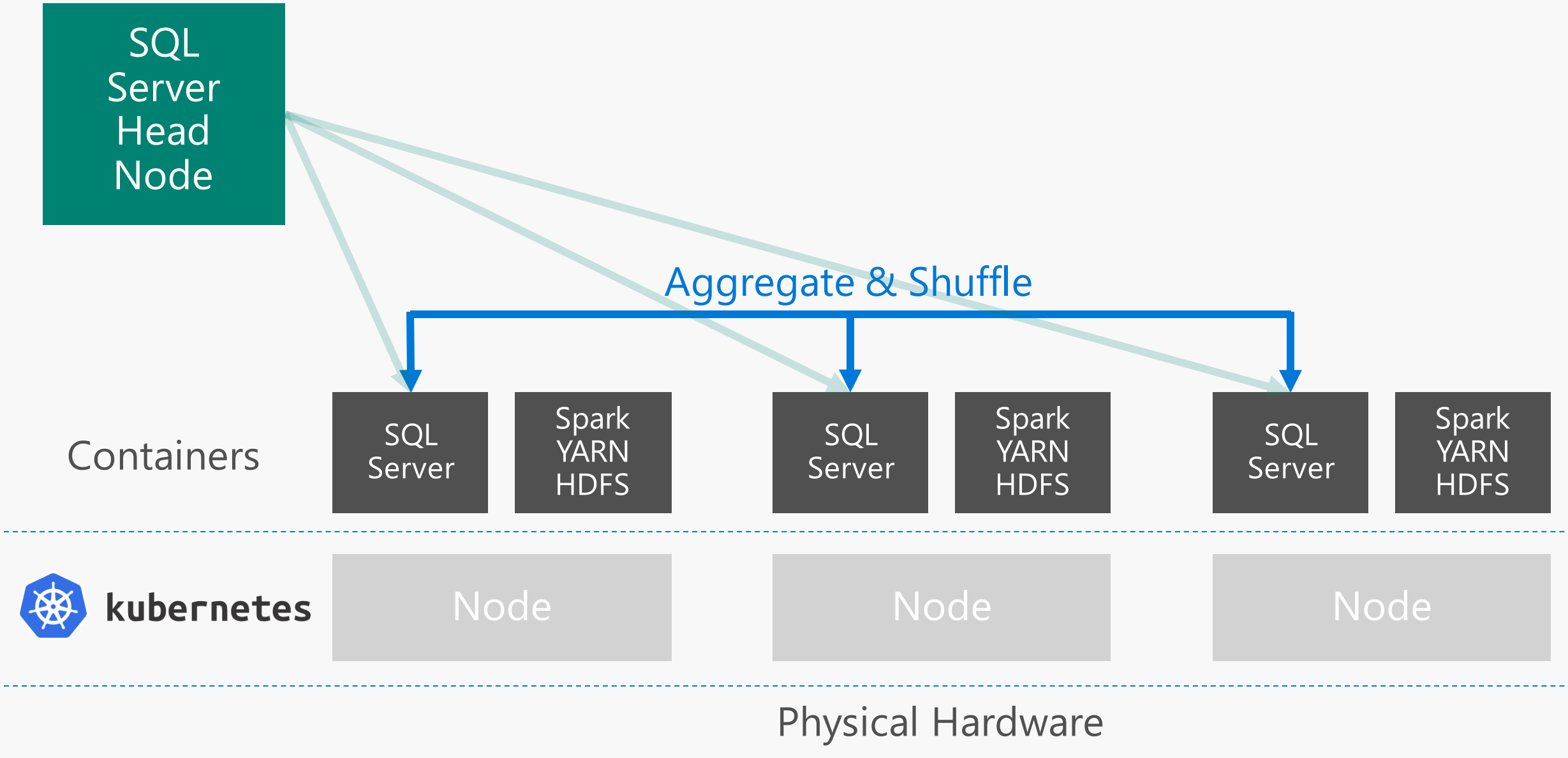
Query compute pool from head node

```
USE high_value_data
GO
-- View data via fan-out queries
--
SELECT count(*) FROM airlinedata
SELECT TOP 10 * FROM airlinedata
GO
```

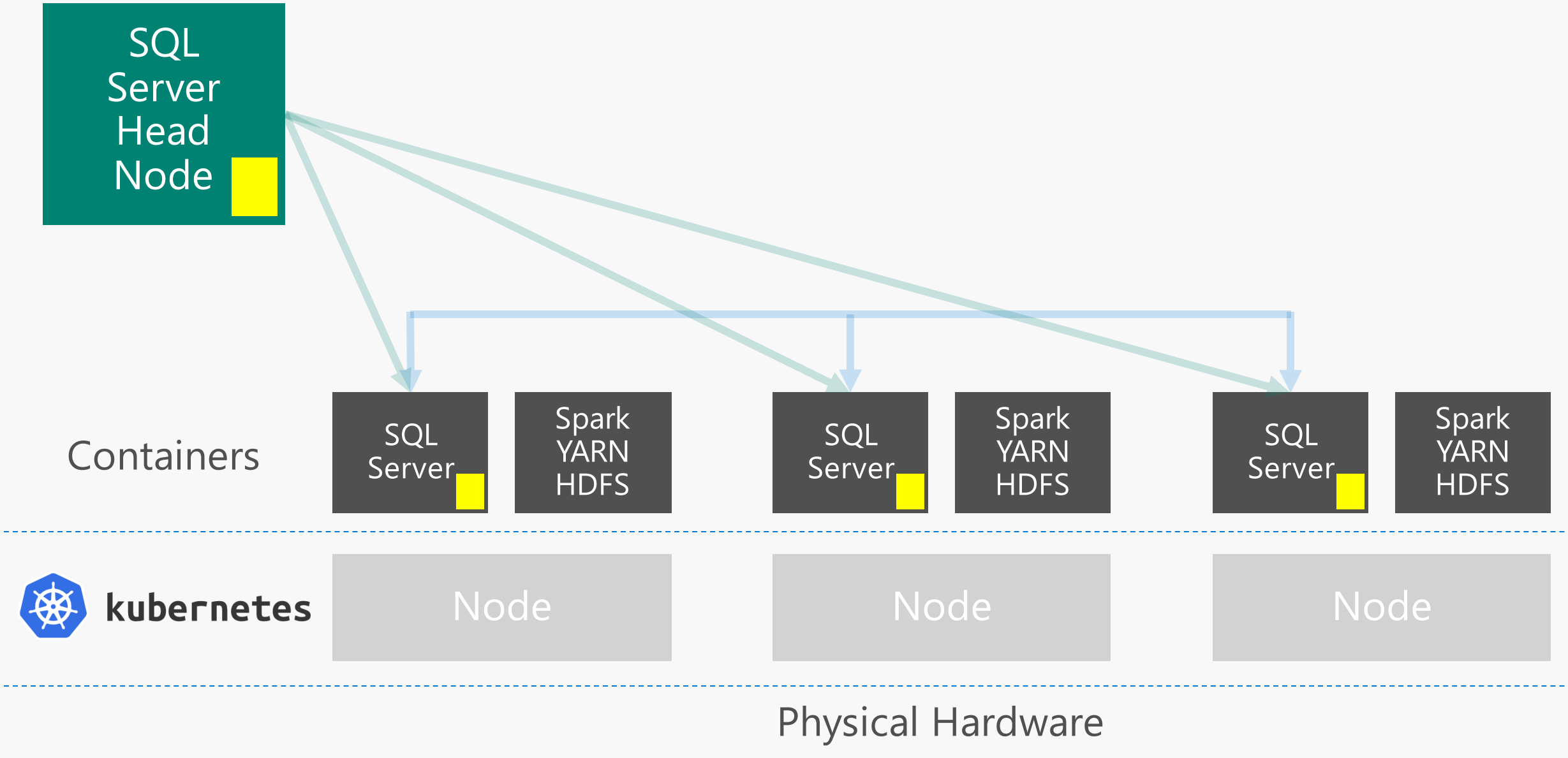
Join head node data and compute pool data

```
USE high_value_data
GO

SELECT E.AircraftRegistration, F.Origin, F.Destination, A.*
  FROM AirlineEngineSensorDataNorm AS A          ← View over high volume external table
  JOIN high_value_data.dbo.AirlineEngines AS E   ← High value data in head node
    ON E.EngineId = A.EngineId
  JOIN high_value_data.dbo.FlightRoutes AS F     ← High value data in head node
    ON F.AircraftRegistration = E.AircraftRegistration AND F.EngineId = E.EngineId
 WHERE A.EngineId IN (9, 48);

GO
```

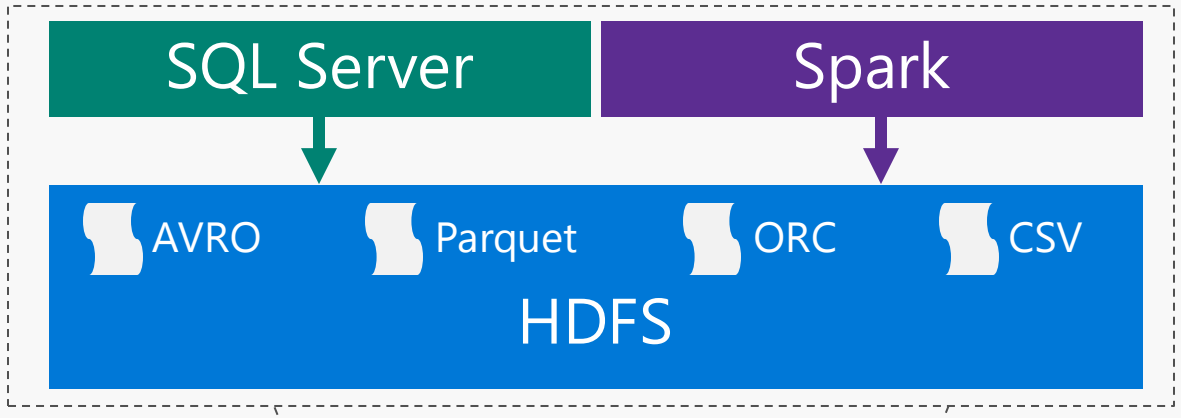
Scale out aggregation and shuffle



Push down dimensional data



SQL Server reads Hadoop file types



Containers

SQL Server | Spark YARN HDFS

SQL Server | Spark YARN HDFS

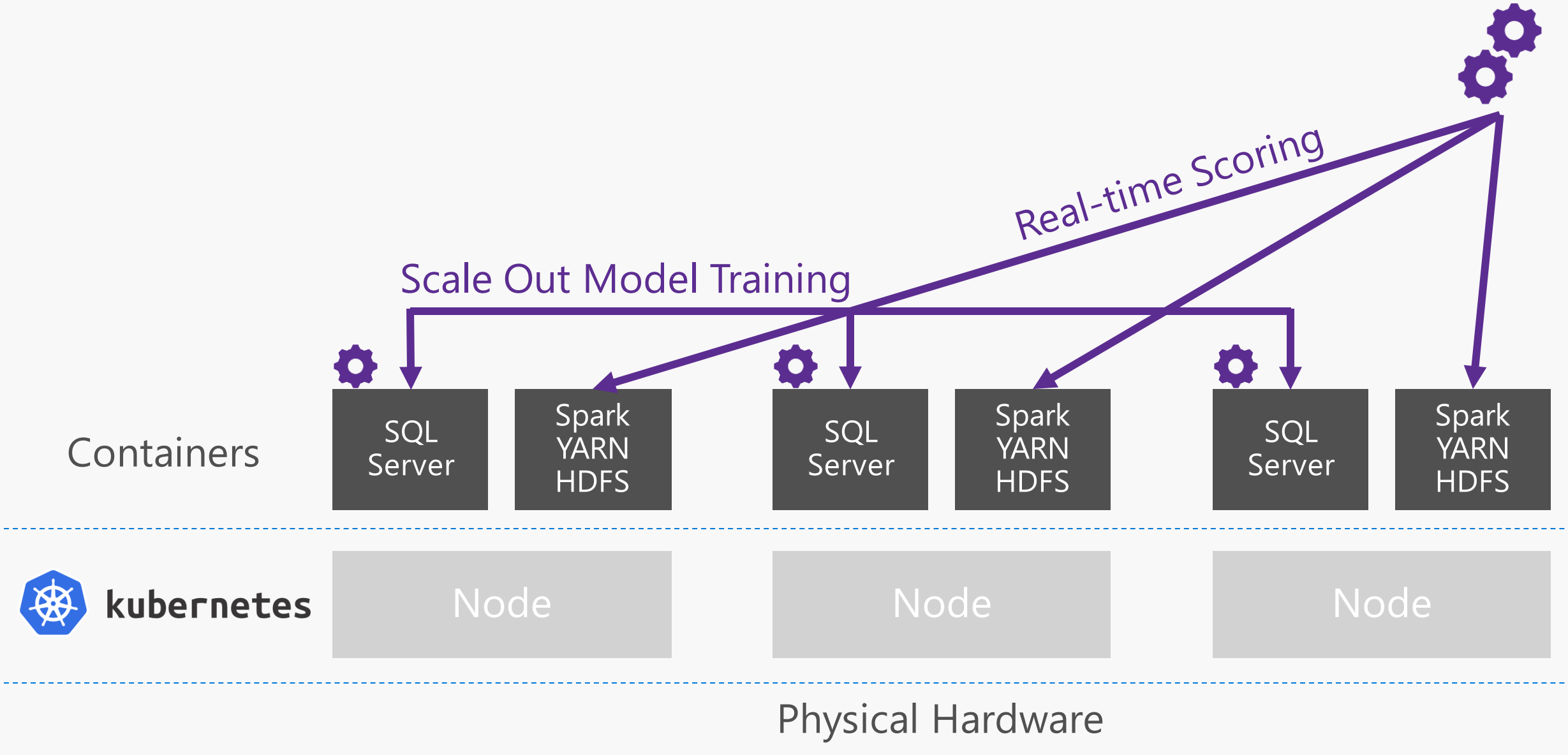
SQL Server | Spark YARN HDFS

kubernetes

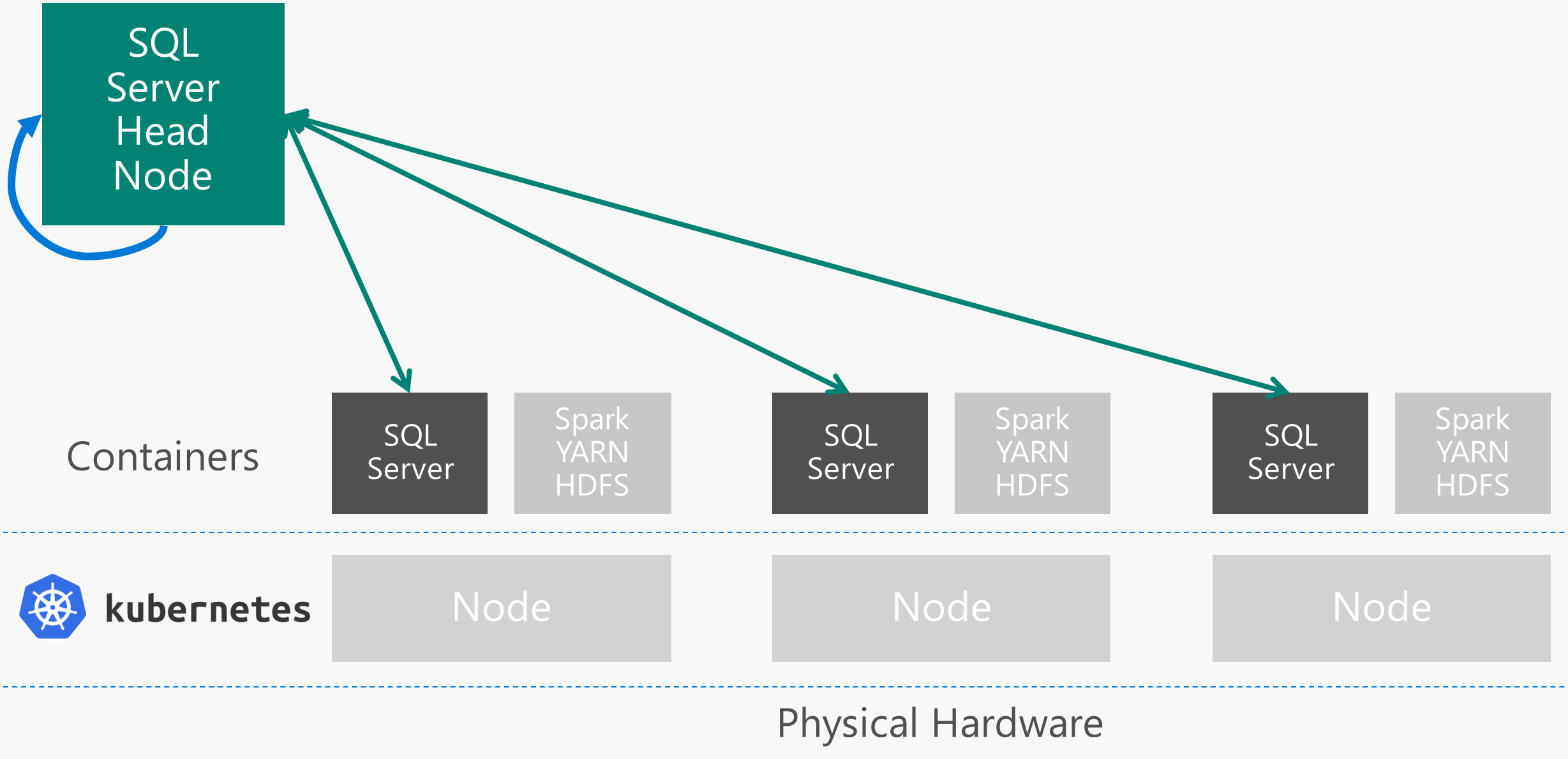
Node | Node | Node

Physical Hardware

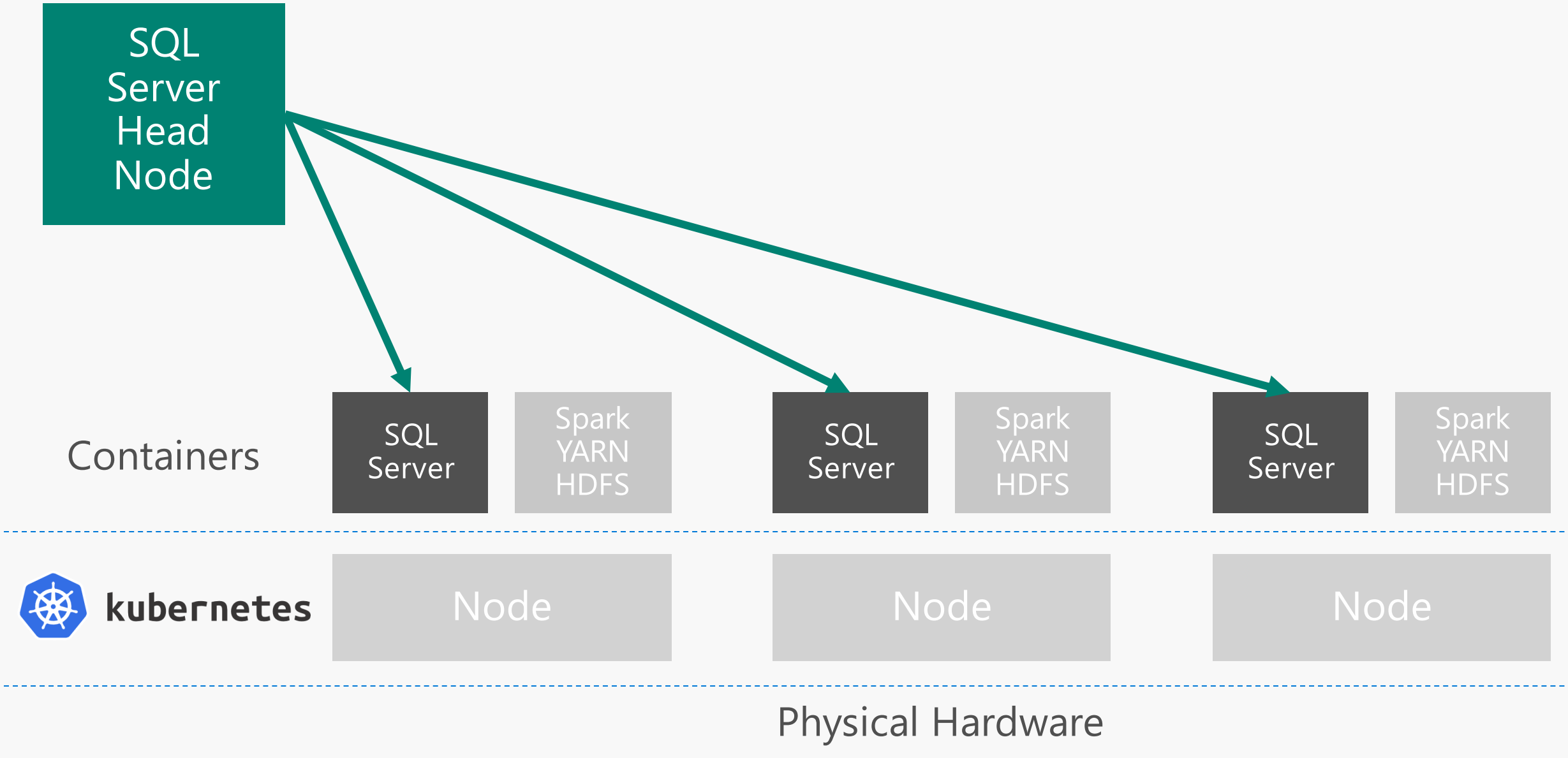
Scale out machine learning



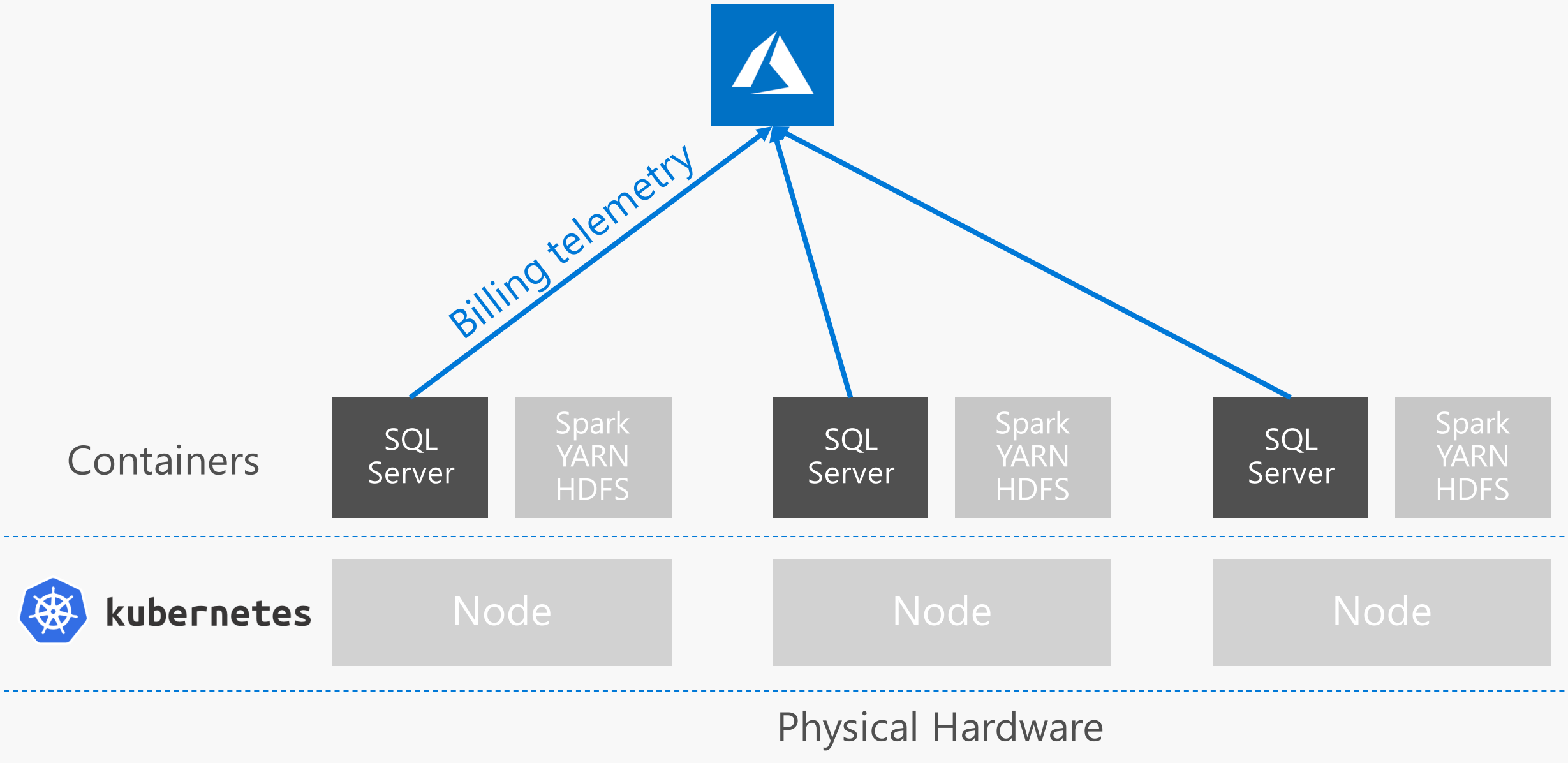
ETL with nothing but T-SQL – roll up data



ETL with nothing but T-SQL – archive data



PAYG, consumption-based subscription



SQL Server Early Adoption Program

Customer Benefits

- Direct access to engineering team via PM buddy and Yammer group
- Visibility into roadmap
- Provide feedback and input into design of new features and functionality
- Full production support from Microsoft Support via special support channel
- License amendment to allow running vNext in production prior to GA
- Release to release upgrade support

Microsoft Goals

- Real world usage of SQL Server in production to verify quality, scale and performance
- Discover bugs
- Discover issues preventing customer adoption in production
- Document customer evidence

Requirements

- NDA
- Sign license agreement amendment
- Complete pre-deployment questionnaire
- Meet with Microsoft Support

ISVs, service providers and hosters with customers in EAP will also be added to EAP so they can participate there.

Apply to join the

SQL Server Early Adoption Program

https://aka.ms/eapsignup

Q&A