初识HBase

- HBase 是一个面向列式存储的分布式数据库，其设计思想来源于 Google 的 BigTable 论文。
- HBase 底层存储基于 HDFS 实现，集群的管理基于 ZooKeeper 实现。
- HBase 良好的分布式架构设计为海量数据的快速存储、随机访问提供了可能，基于数据副本机制和分区机制可以轻松实现在线扩容、缩容和数据容灾。
- 是大数据领域中 Key-Value 数据结构存储最常用的数据库方案。

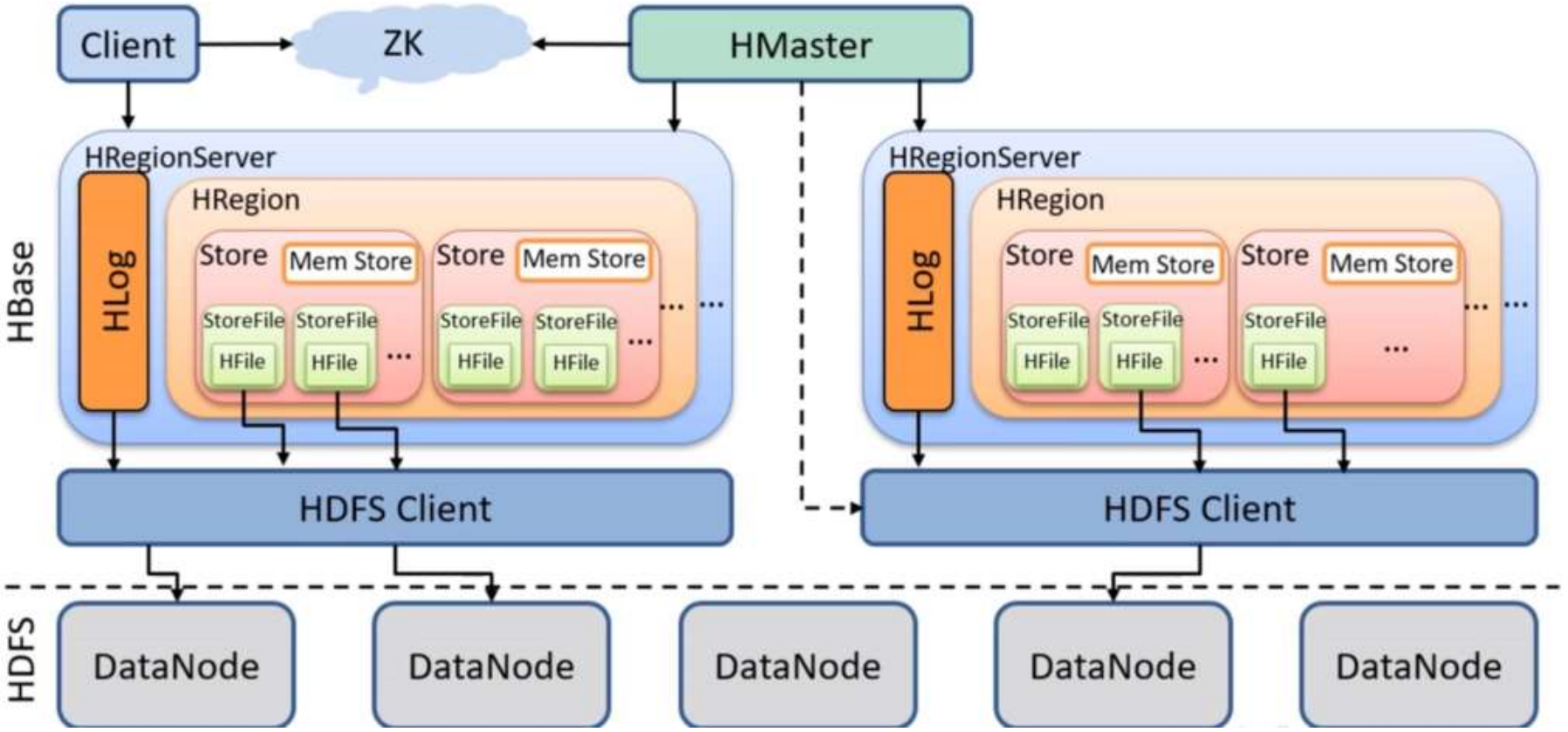
HBase特点(1/2)

- ◆易扩展
  - Hbase 的扩展性主要体现在两个方面，一个是基于运算能力（RegionServer） 的扩展，通过增加 RegionSever 节点的数量，提升 Hbase 上层的处理能力。
  - 另一个是基于存储能力的扩展（HDFS），通过增加 DataNode 节点数量对存储层的进行扩容，提升 HBase 的数据存储能力。
- ◆海量存储
  - HBase 作为一个开源的分布式 Key-Value 数据库，其主要作用是面向 PB 级别数据的实时入库和快速随机访问。这主要源于上述易扩展的特点，使得 HBase 通过扩展来存储海量的数据。

HBase特点(2/2)

- 高可靠性
  - WAL 机制保证了数据写入时不会因集群异常而导致写入数据丢失，Replication 机制保证了在集群出现严重的问题时，数据不会发生丢失或损坏。而且 Hbase 底层使用 HDFS，HDFS 本身也有备份。
- 稀疏性
  - 在 HBase 的列族中，可以指定任意多的列，为空的列不占用存储空间，表可以设计得非常稀疏。

HBase体系结构



北京理工大學
BEIJING INSTITUTE OF TECHNOLOGY

HMaster

- 负责管理 RegionServer，实现其负载均衡；
- 管理和分配 Region，比如在 Region split时分配新的 Region，在 RegionServer 退出时迁移其内的 Region 到其他 RegionServer上；
- 管理namespace和table的元数据（实际存储在HDFS上）；
- 权限控制（ACL）

RegionServer

- 存放和管理本地 Region；
- 读写HDFS，管理Table中的数据；
- Client 从 HMaster 中获取元数据，找到 RowKey 所在的 RegionServer 进行读写数据。

ZooKeeper

- 存放整个 HBase集群的元数据以及集群的状态信息；
- 实现HMaster主从节点的failover。

HBase 数据模型

- HBase 表中，一条数据拥有一个全局唯一的键(RowKey)和任意数量的列(Column)，一列或多列组成一个列族，HBase 会将表按主键划分为多个 Region 存储在不同 Region Server 上，以完成数据的分布式存储和读取。

Column Family

- Column Family 即列族，HBase 基于列划分数据的物理存储，一个列族可以包含包意多列。
- 一般同一类的列会放在一个列族中，每个列族都有一组存储属性：
  - 是否应该缓存在内存中；
  - 数据如何被压缩或行键如何编码等。
- HBase 在创建表的时候就必须指定列族。HBase的列族不是越多越好，官方荐一个表的列族数量最好小于或者等于3，过多的列族不利于 HBase 数据的管理和索引。

RowKey

- RowKey的概念与关系型数据库中的主键相似，HBase 使用 RowKey 来唯一标识行数据。
- 访问 HBase 数据的方式有三种:
  - 基于 RowKey的单行查询;
  - 基于RowKey的范围查询;
  - 全表扫描查询。

Region

- HBase 将表中的数据基于 RowKey 的不同范围划分到不同 Region 上，每个Region负责一定范围的数据存储和访问。
- 表开始只有一个 Region，随着数据增加，Region 不断增大，当增大到一个阀值的时候，Region 就会等分成两个新的 Region。
- Region 是 HBase 中分布式存储和负载均衡的最小单元，不同的 Region 可以分布在不同的 HRegion Server上，但同一个Region不会拆分到多个HRegion Server上的
- 由于数据被划分到不同的 Region上，每个 Region 都可以独立地进行写入和查询。

TimeStamp

- TimeStamp 是实现 HBase 多版本的关键。在HBase 中，使用不同 TimeStamp 来标识相同RowKey对应的不同版本的数据。
- 相同 RowKey的数据按照 TimeStamp 倒序排列。默认查询的是最新的版本，当然用户也可以指定 TimeStamp 的值来读取指定版本的数据。

行式存储与列式存储

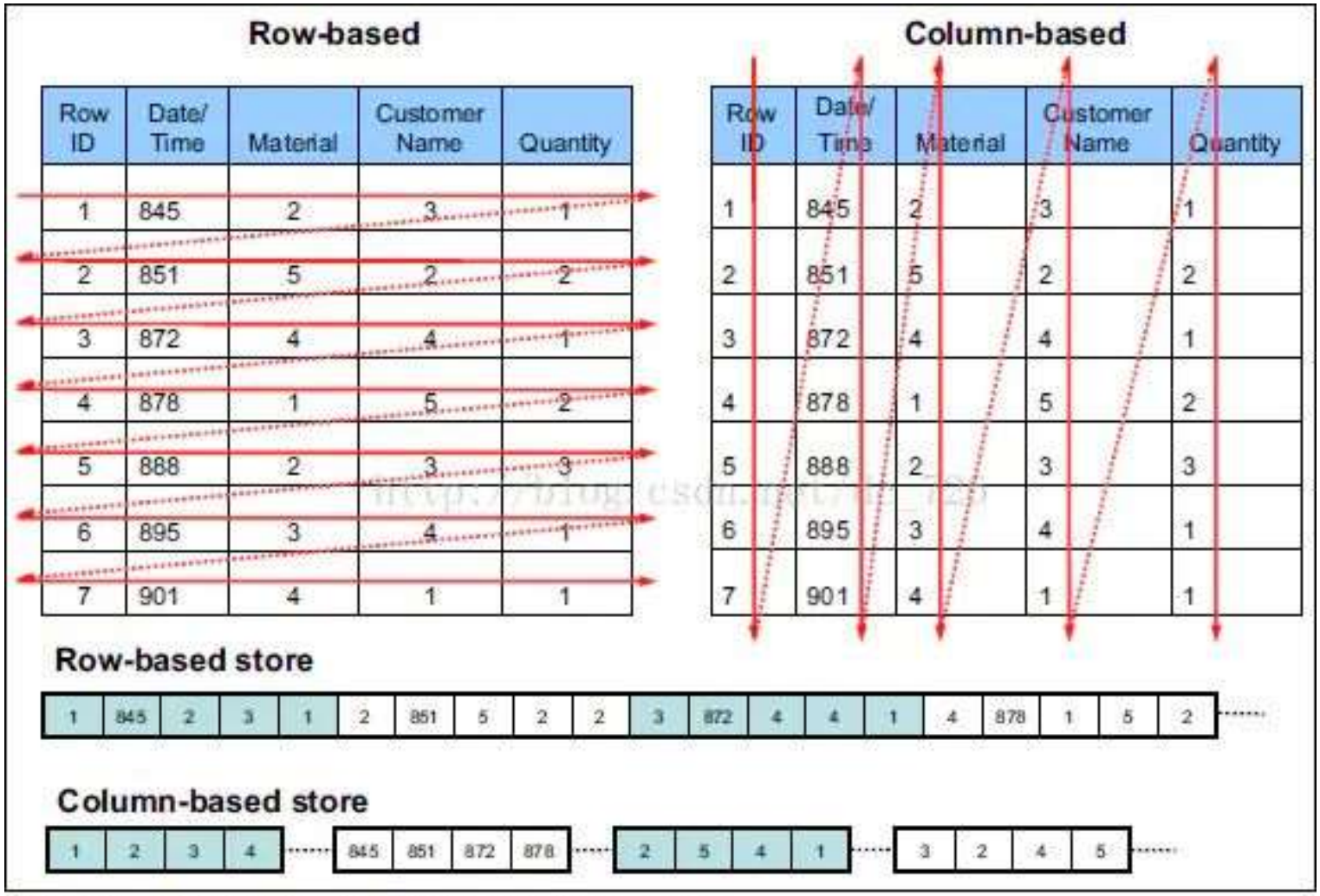


Figure 1-4 Row-based and column-based storage models

北京理工大學
BEIJING INSTITUTE OF TECHNOLOGY

行式与列式存储示例

SSN	Name	Age	Addr	City	St
101259797	SMITH	88	899 FIRST ST	JUNO	AL
892375862	CHIN	37	16137 MAIN ST	POMONA	CA
318370701	HANDU	12	42 JUNE ST	CHICAGO	IL

Block 1 Block 2 Block 3

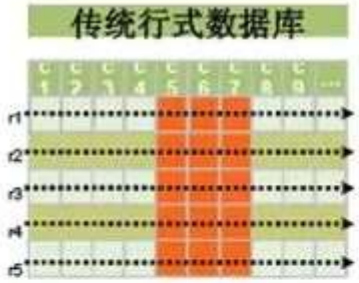
SSN	Name	Age	Addr	City	St
101259797	SMITH	88	899 FIRST ST	JUNO	AL
892375862	CHIN	37	16137 MAIN ST	POMONA	CA
318370701	HANDU	12	42 JUNE ST	CHICAGO	IL

Block 1

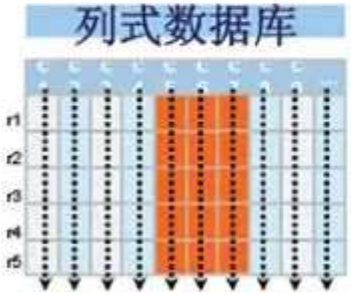
北京理工大學
BEIJING INSTITUTE OF TECHNOLOGY

行式存储与列式存储

	行式存储	列式存储
优点	Ø数据被保存在一起 ØINSERT/UPDATE容易	Ø 查询时只有涉及到的列会被读取 Ø 投影(projection)很高效 Ø 任何列都能作为索引
缺点	Ø 选择(Selection)时即使只涉及某几列，所有数据也都会被读取	Ø 选择完成时，被选择的列要重新组装 Ø INSERT/UPDATE比较麻烦



- 数据是按行存储的
- 没有索引的查询使用大量I/O
- 建立索引和物化视图需要花费大量时间和资源
- 面对查询的需求，数据库必须被大量膨胀才能满足性能要求



- 数据按列存储 - 每一列单独存放
- 数据即是索引
- 只访问查询涉及的列 - 大量降低系统IO
- 每一列由一个线索来处理 - 查询的并发处理
- 数据类型一致，数据特征相似 - 高效压缩

数据压缩

Row ID	Date/ Time	Material	Customer Name	Quantity
1	14:05	Radio	Dubois	1
2	14:11	Laptop	Di Dio	2
3	14:32	Stove	Miller	1
4	14:38	MP3 Player	Newman	2
5	14:48	Radio	Dubois	3
6	14:55	Refrigerator	Miller	1
7	15:01	Stove	Chevrier	1

#	Customers
1	Chevrier
2	Di Dio
3	Dubois
4	Miller
5	Newman

#	Material
1	MP3 Player
2	Radio
3	Refrigerator
4	Stove
5	Laptop

Row ID	Date/ Time	Material	Customer Name	Quantity
1	845	2	3	1
2	851	5	2	2
3	872	4	4	1
4	878	1	5	2
5	888	2	3	3
6	895	3	4	1
7	901	4	1	1

Figure 1-3 Illustration of dictionary compression

列式存储查询性能示例

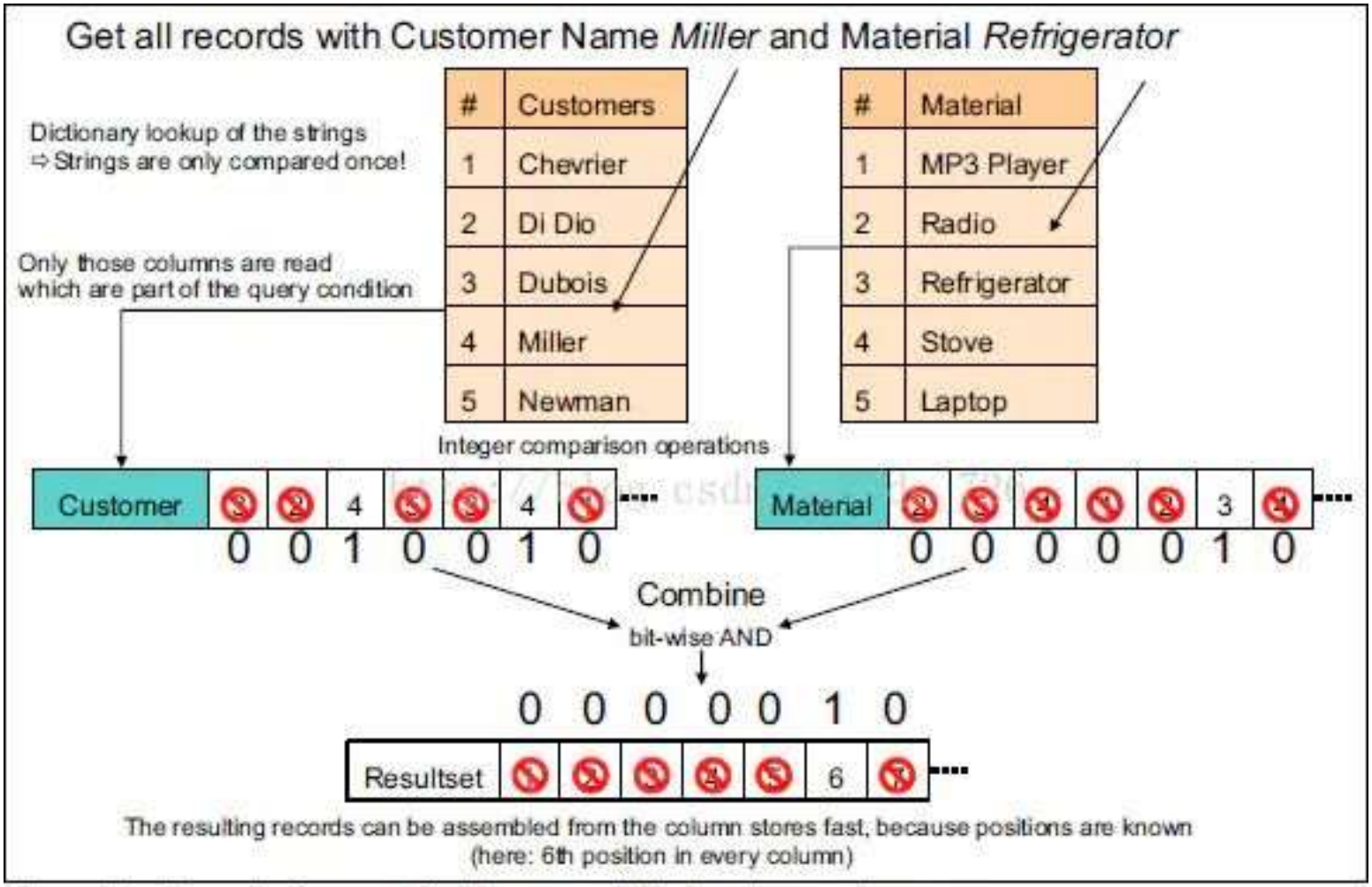
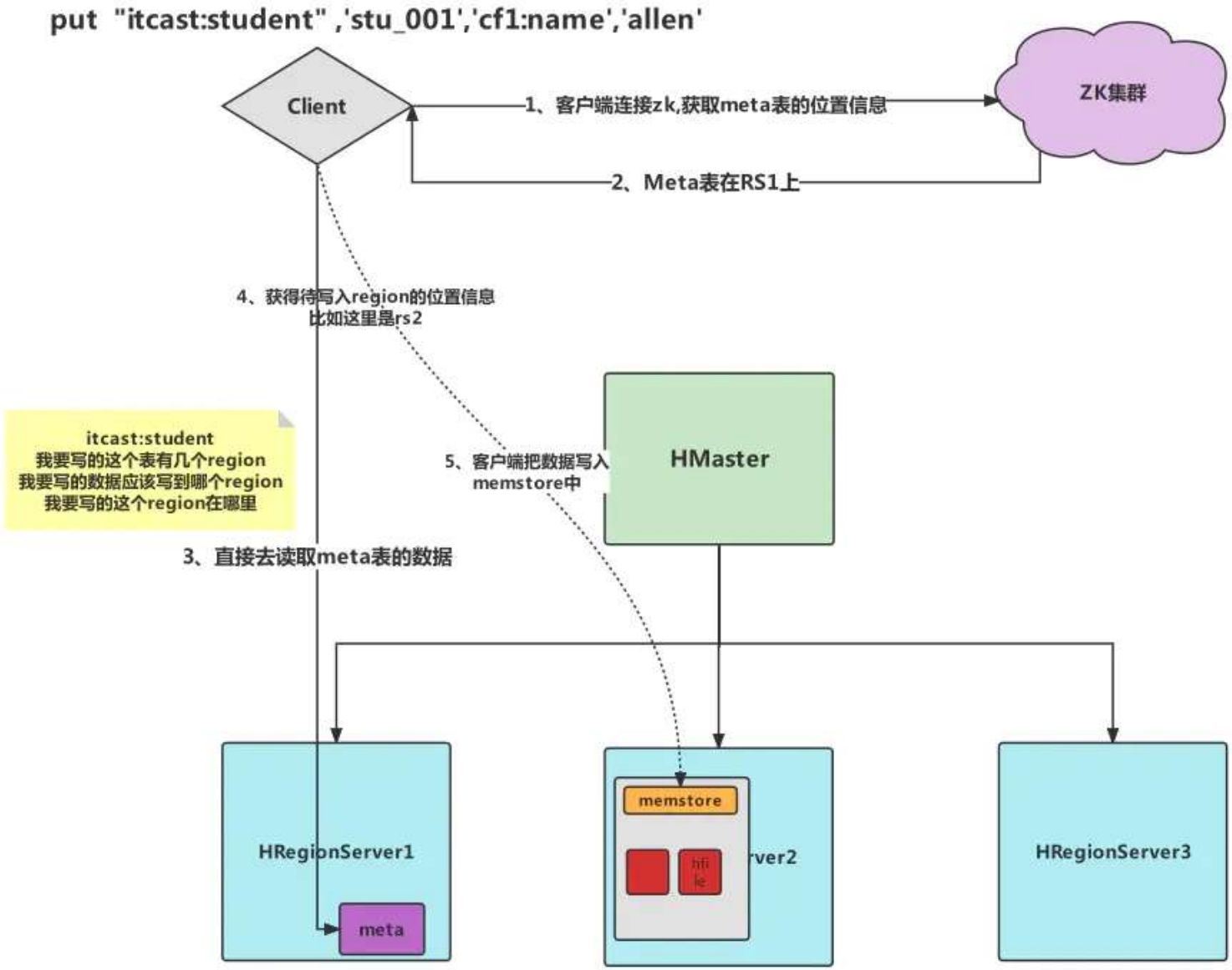


Figure 1-5 Example of a query that is run on a table in columnar storage

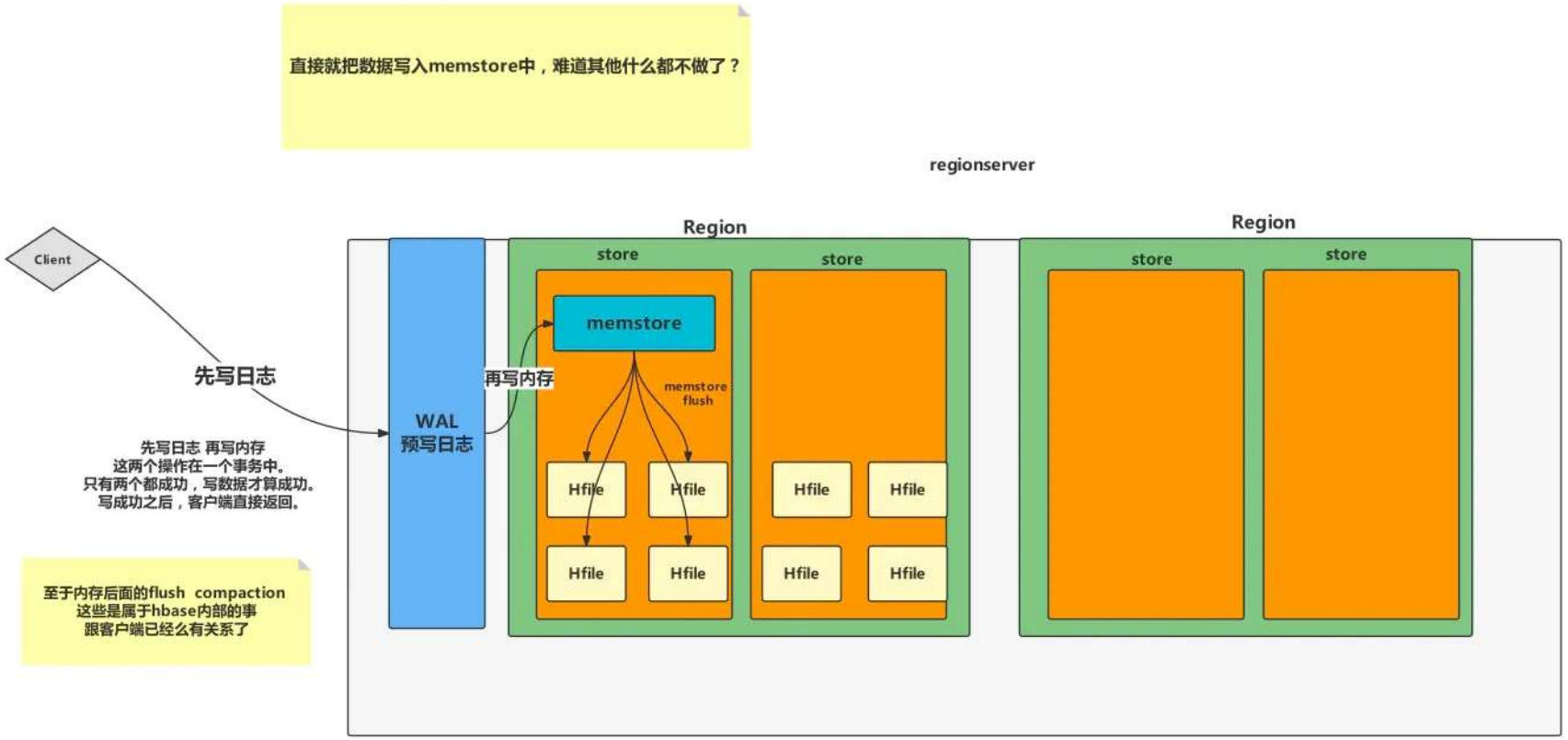
北京理工大學
BEIJING INSTITUTE OF TECHNOLOGY

Hbase写流程



北京理工大學
BEIJING INSTITUTE OF TECHNOLOGY

预写日志WAL（Write-Ahead-Log）

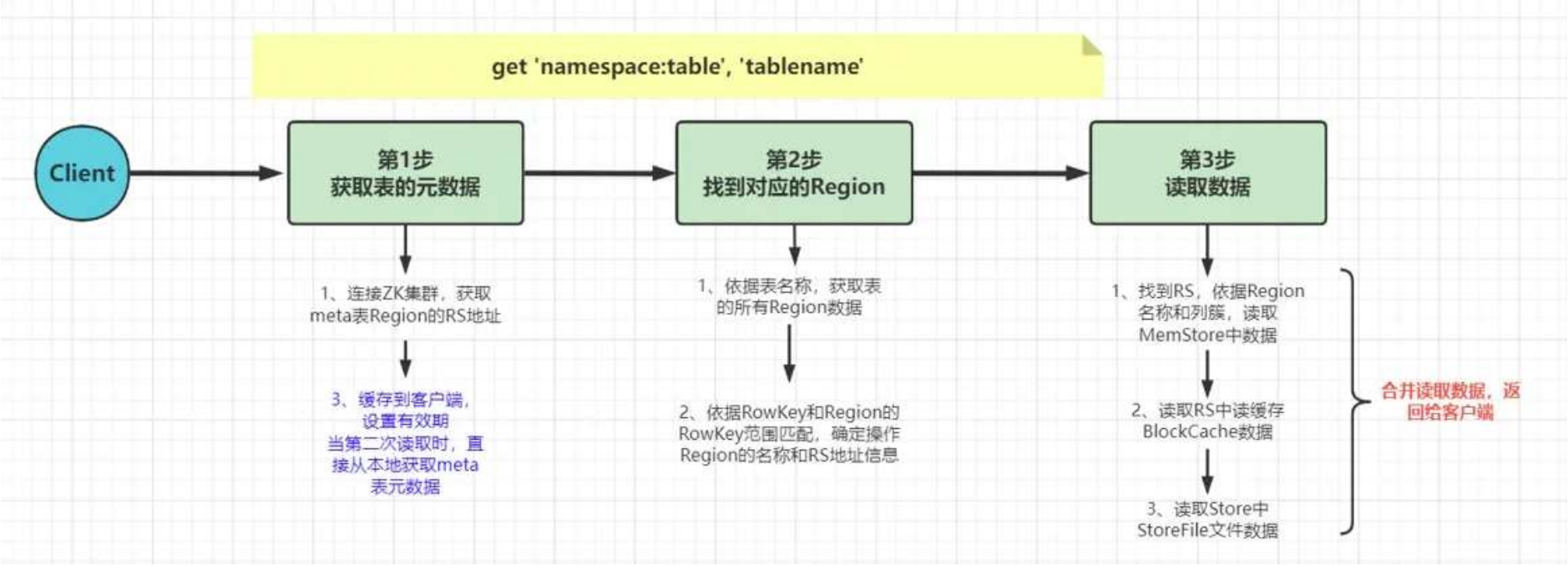


北京理工大學
BEIJING INSTITUTE OF TECHNOLOGY

预写日志WAL（Write-Ahead-Log）

- WAL最重要的作用是灾难恢复。
- 和关系数据库的日志类似，它记录所有的数据改动。
- 通过重做log中所记录的改动，可以恢复崩溃之前的数据。
- 如果写入WAL失败，整个操作将认为失败。

HBase读流程

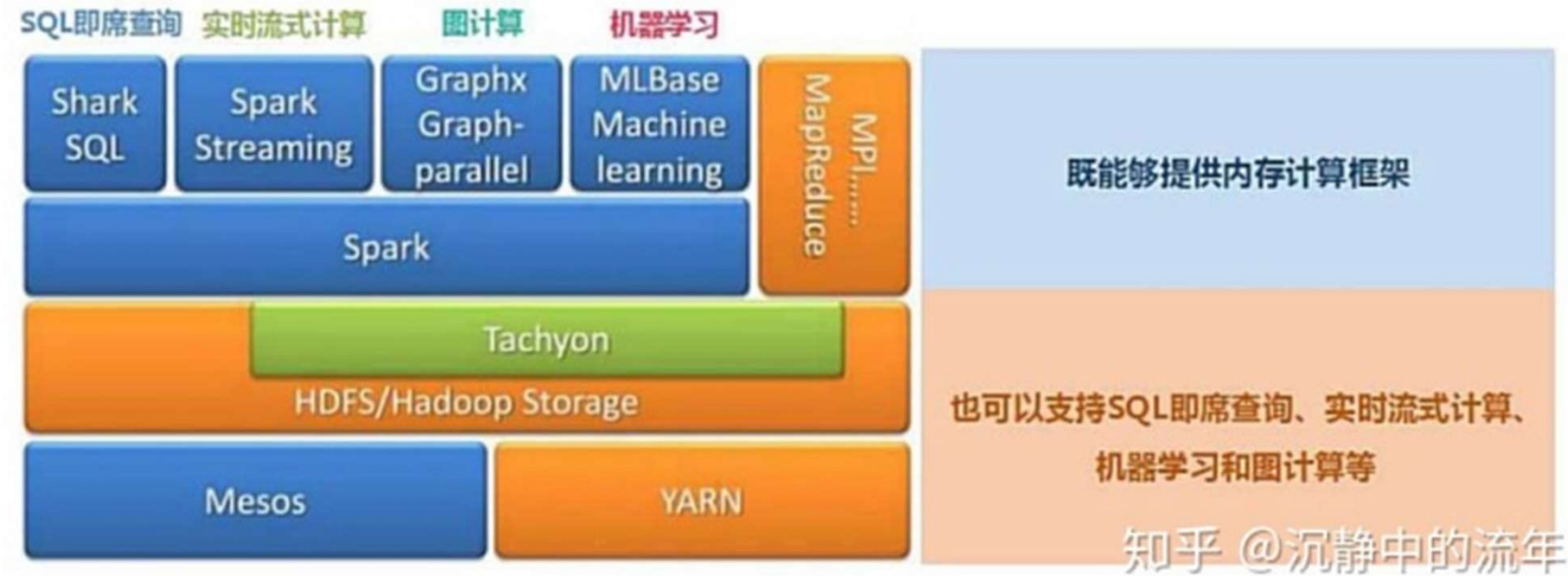


北京理工大學
BEIJING INSTITUTE OF TECHNOLOGY

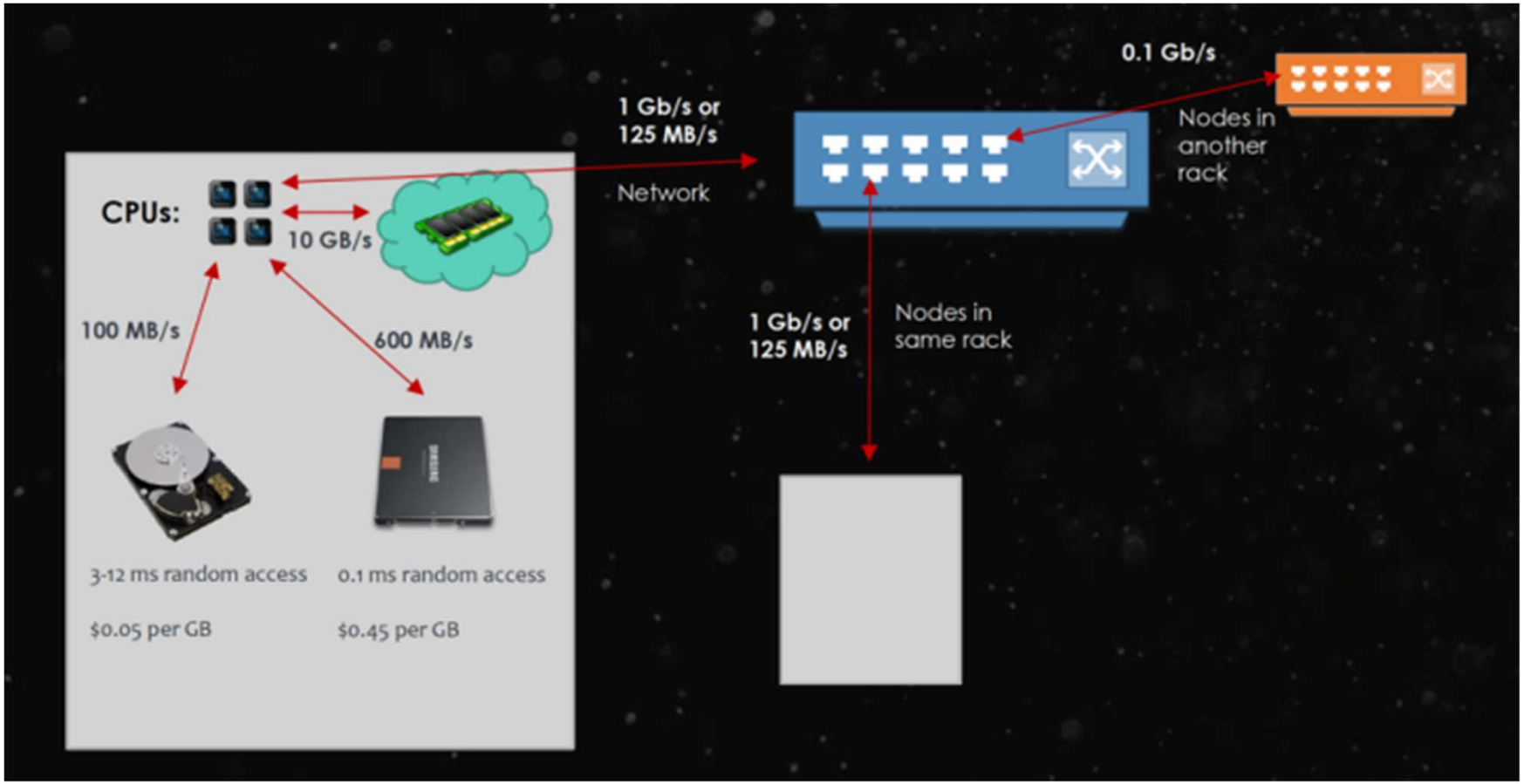
Apache Spark

通用的分布式数据处理引擎
大数据计算框架MapReduce的继任者

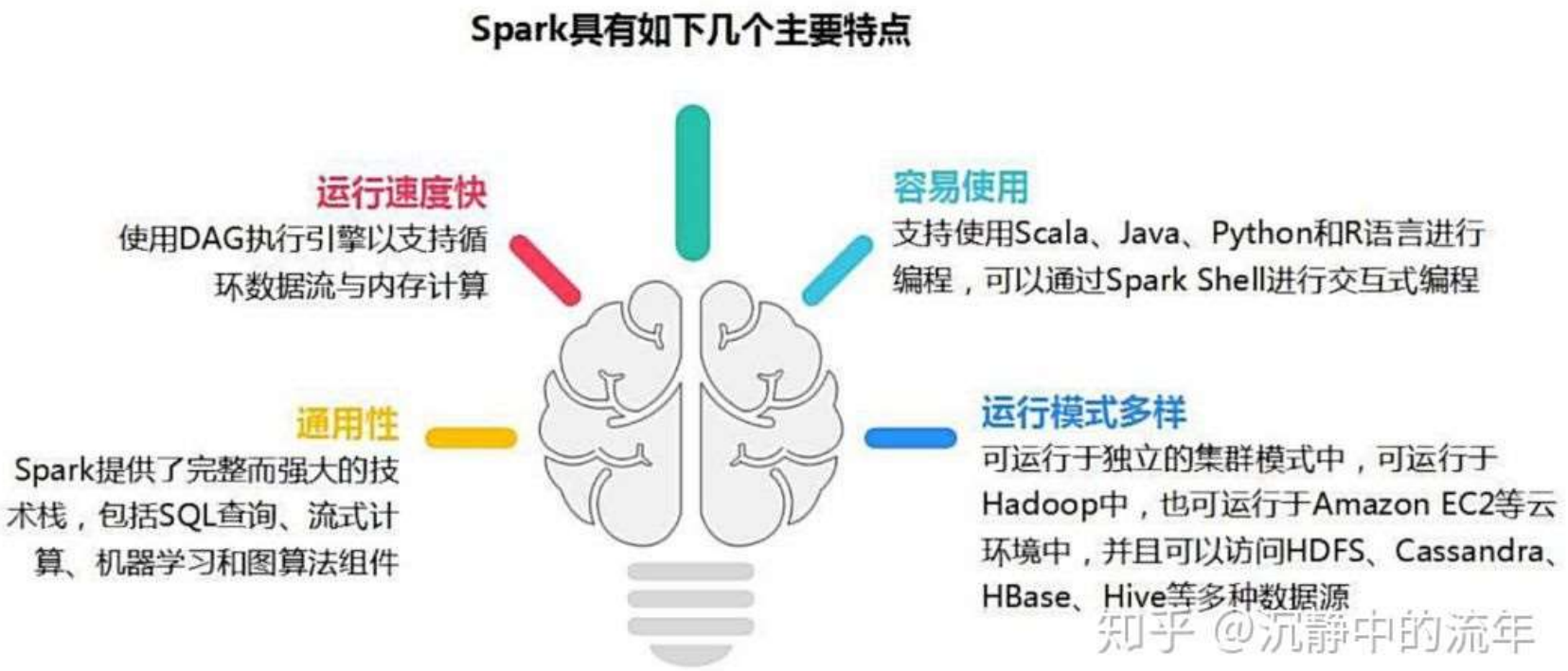
Spark结构


SQL即席查询
实时流式计算
图计算
机器学习
Shark SQL
Spark Streaming
Graphx Graph-parallel
MLBase Machine learning
MPI,..... MapReduce
Spark
Tachyon
HDFS/Hadoop Storage
Mesos
YARN
既能够提供内存计算框架
也可以支持SQL即席查询、实时流式计算、机器学习和图计算等
知乎 @沉静中的流年

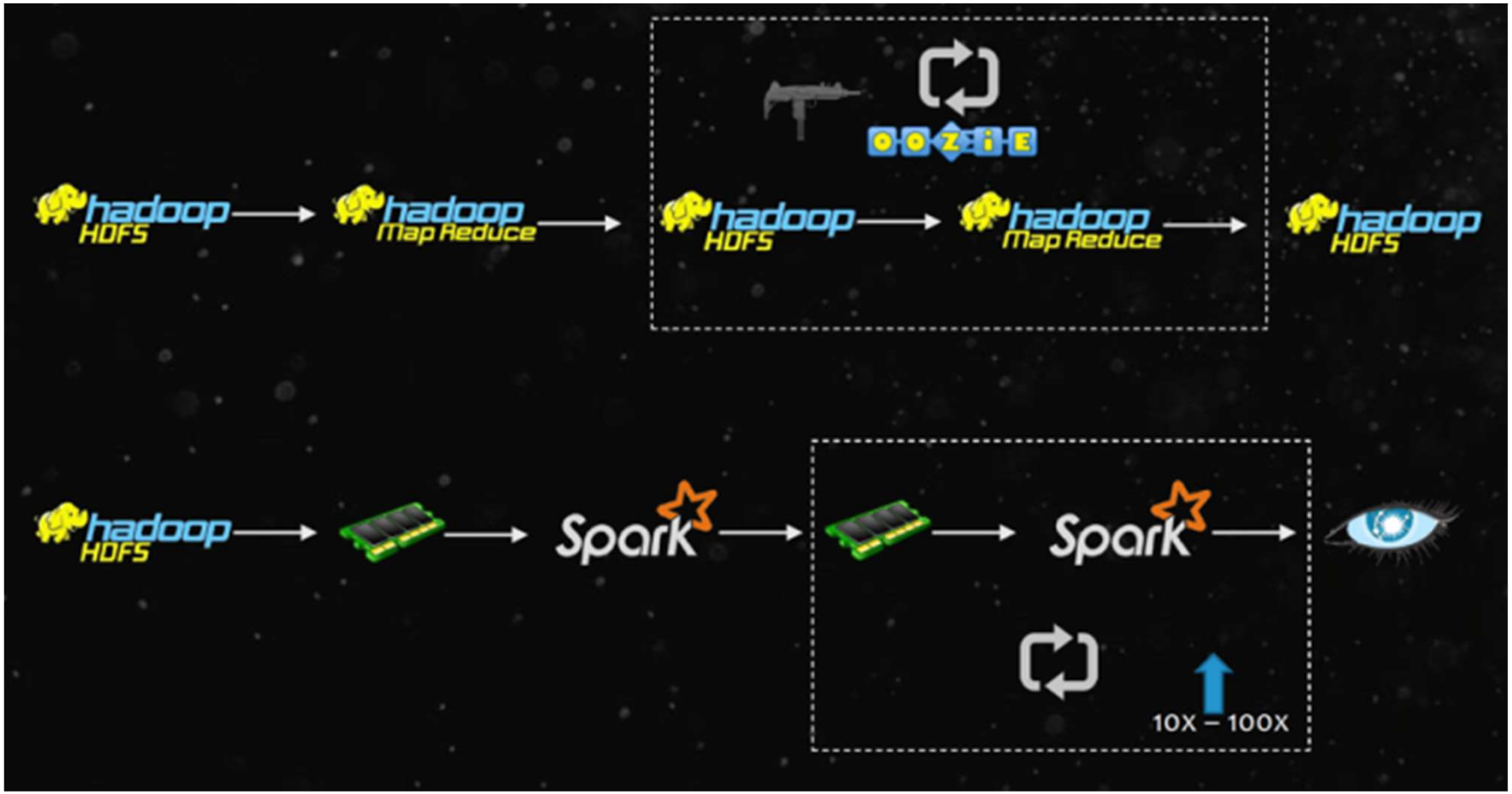
计算机不同资源的性能


0.1 Gb/s
1 Gb/s or
125 MB/s
Network
Nodes in
another
rack
CPUs:
10 GB/s
100 MB/s
600 MB/s
1 Gb/s or
125 MB/s
Nodes in
same rack
3-12 ms random access
$0.05 per GB
0.1 ms random access
$0.45 per GB

Spark优点

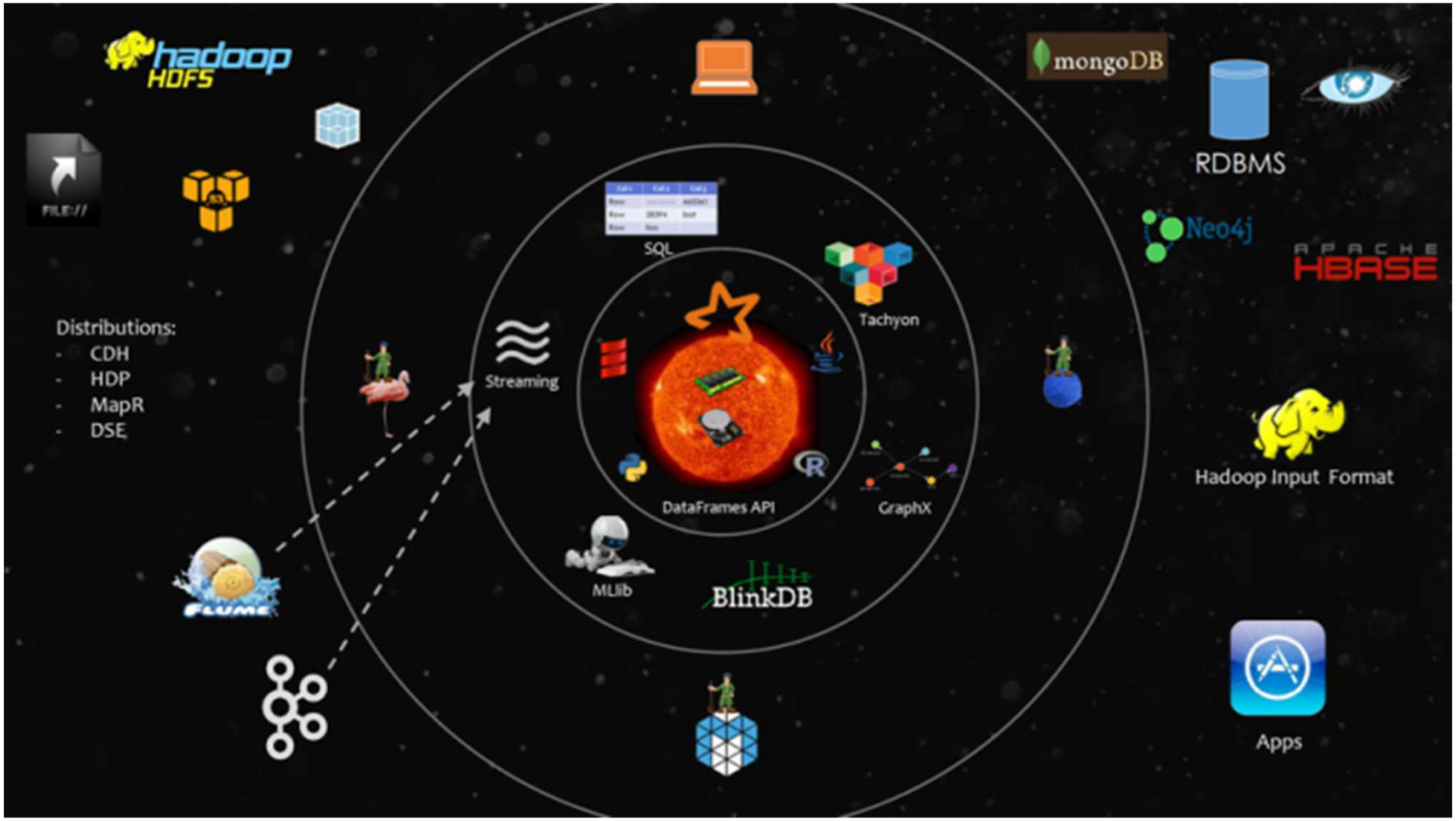

Spark具有如下几个主要特点
运行速度快
使用DAG执行引擎以支持循环数据流与内存计算
容易使用
支持使用Scala、Java、Python和R语言进行编程，可以通过Spark Shell进行交互式编程
通用性
Spark提供了完整而强大的技术栈，包括SQL查询、流式计算、机器学习和图算法组件
运行模式多样
可运行于独立的集群模式中，可运行于Hadoop中，也可运行于Amazon EC2等云环境中，并且可以访问HDFS、Cassandra、HBase、Hive等多种数据源
知乎 @沉静中的流年

Spark比MapReduce快

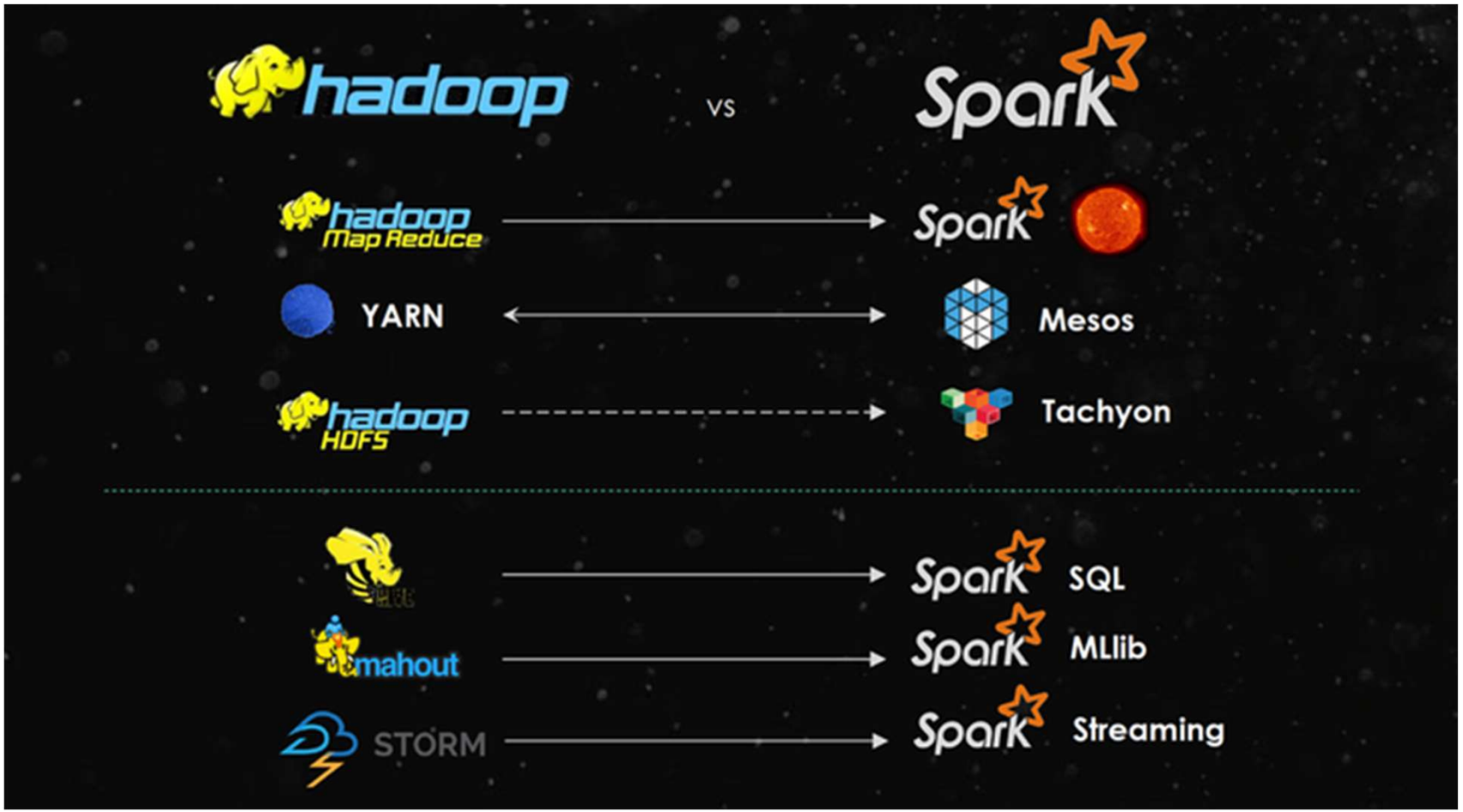


北京理工大學
BEIJING INSTITUTE OF TECHNOLOGY

Spark生态圈


hadoop
HDFS
FILE://
mongoDB
RDBMS
Neo4j
APACHE
HBASE
Distributions:
- CDH
- HDP
- MapR
- DSE
SQL
Tachyon
Streaming
DataFrames API
GraphX
MLlib
BlinkDB
FLUME
Hadoop Input Format
Apps

Spark与Hadoop



北京理工大學
BEIJING INSTITUTE OF TECHNOLOGY

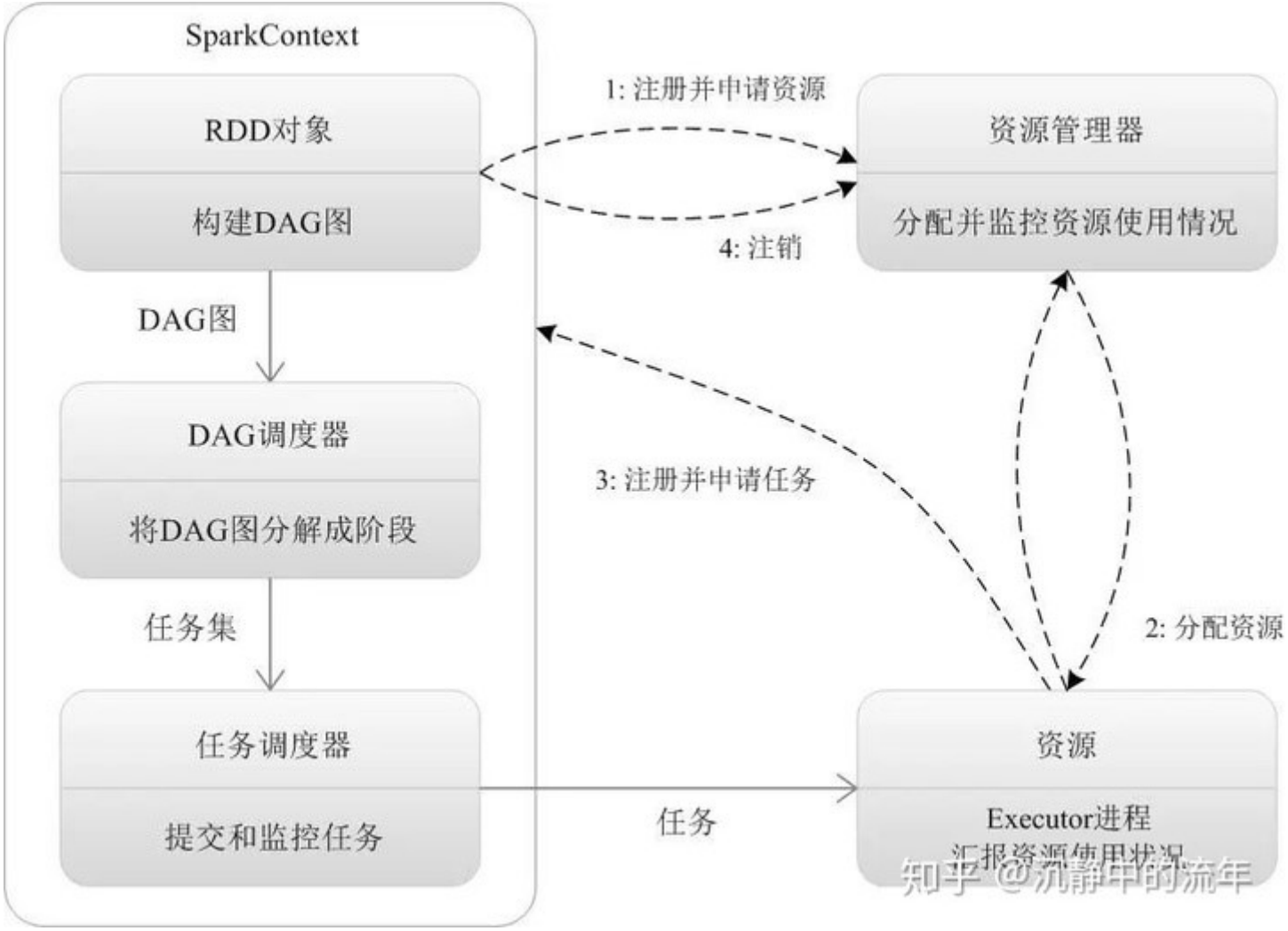
Spark基本概念（1/2）

- **RDD**：是弹性分布式数据集（Resilient Distributed Dataset）的简称，是分布式内存的一个抽象概念，提供了一种高度受限的共享内存模型。
- **DAG**：是Directed Acyclic Graph（有向无环图）的简称，反映RDD之间的依赖关系。
- **Driver Program**：控制程序，负责为Application构建DAG图。
- **Cluster Manager**：集群资源管理中心，负责分配计算资源。
- **Worker Node**：工作节点，负责完成具体计算。
- **Executor**：是运行在工作节点（Worker Node）上的一个进程，负责运行Task，并为应用程序存储数据。

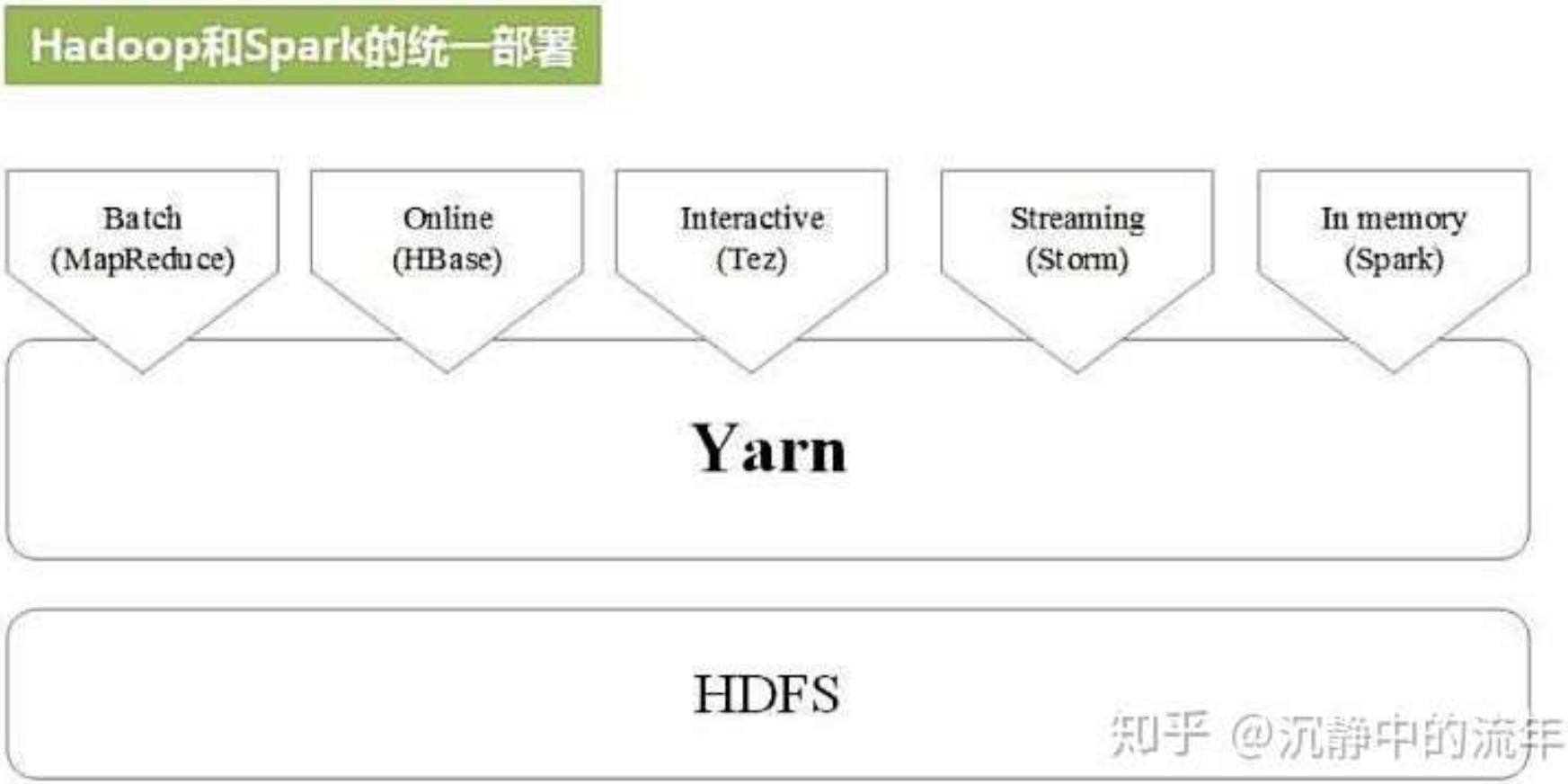
Spark基本概念（2/2）

- **Application**：用户编写的Spark应用程序，一个Application包含多个Job。
- **Job**：作业，一个Job包含多个RDD及作用于相应RDD上的各种操作。
- **Stage**：阶段，是作业的基本调度单位，一个作业会分为多组任务，每组任务被称为“阶段”。
- **Task**：任务，运行在Executor上的工作单元，是Executor中的一个线程。
- **总结**：Application由多个Job组成，Job由多个Stage组成，Stage由多个Task组成。Stage是作业调度的基本单位。

Spark运行流程



Spark部署


Hadoop和Spark的统一部署
Batch
(MapReduce)
Online
(HBase)
Interactive
(Tez)
Streaming
(Storm)
In memory
(Spark)
Yarn
HDFS
知乎 @沉静中的流年