新一代数据仓库: HAWQ

常雷 博士

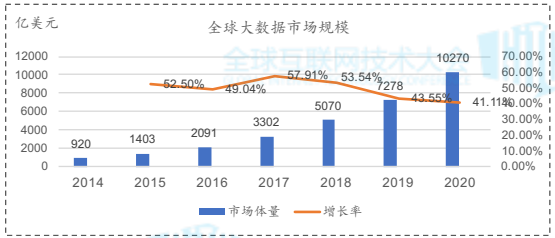
偶数科技CEO, Apache HAWQ创始人

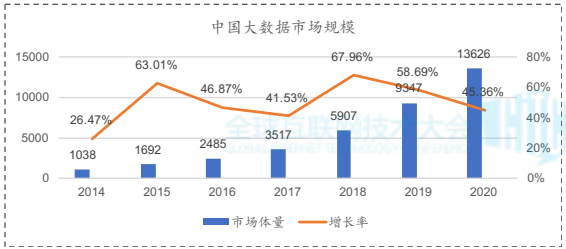
www.oushu.io

oushu

目录

数据生态系统





全球数据仓库市场规模2016年达数百亿美金

应用
用户行为分析、反欺诈、用户画像、信用模型

数据安全

BI
tableau Qlik, Power BI

分析挖掘 / 机器学习/AI
SAS, SPSS,Tensorflow

ETL
Informatica
Talend
Kettle

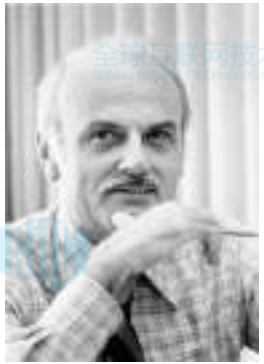
OLAP
数据仓库(Data Warehouse)
MPP, SQL-on-Hadoop, New Data Warehouse

OLTP
关系数据库，NoSQL，NewSQL

数据治理

Cloud (公有云和私有云)

数据库：55年

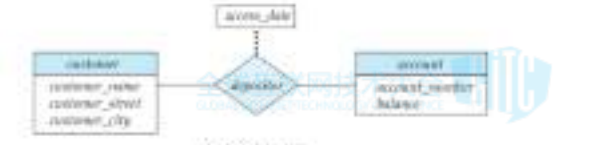


- Database: 1962年出现
  - Inverted File Database System
  - System Development Corporation
- 数据库的几个阶段
  - 1960s: Navigational DBMS (网状 & 层次模型)
    - Integrated Data Store (IDS)
    - Information Management System (IMS)
  - 1970s - 1990s: SQL/Relational DBMS
    - OLTP, Data warehouse, MPP
  - 2000s - Present: Post Relational
    - NoSQL (XML, KV, Graph, Tree), NewSQL, NewDW

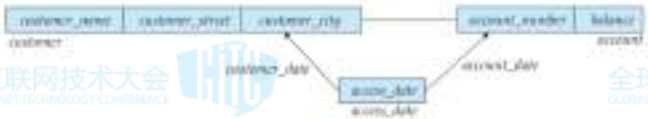
数据库的核心

- 数据模型 & 查询语言
- 查询优化和执行
- 索引与存储
- 事务处理

网状/层次模型



(a) E-R diagram



(b) Network diagram


```
customer.customer_city := "Harrison";
find any customer using customer_city;
while DB-status = 0 do
    begin
        get customer;
        print (customer.customer_name);
        find duplicate customer using customer_city;
    end;
```

找出住在Harrison的所有客户

Charles Bachman
1973 Turing Award

关系模型

(a) E-R diagram

Edgar F. Codd
1981 Turing Award

```
Select customer_name
From customer
Where customer_city  = 'Harrison';
```

找出住在Harrison的所有客户

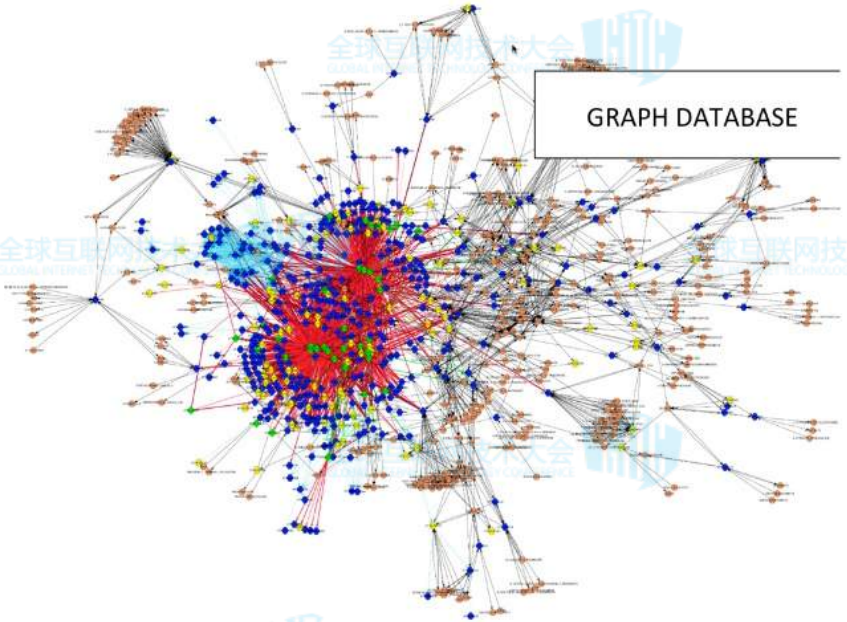
A Relational Model of Data for Large Shared Data Banks.

Jim Gray
1998 Turing Award

Michael Stonebraker
2014 Turing Award

Graph/Tree/KV模型

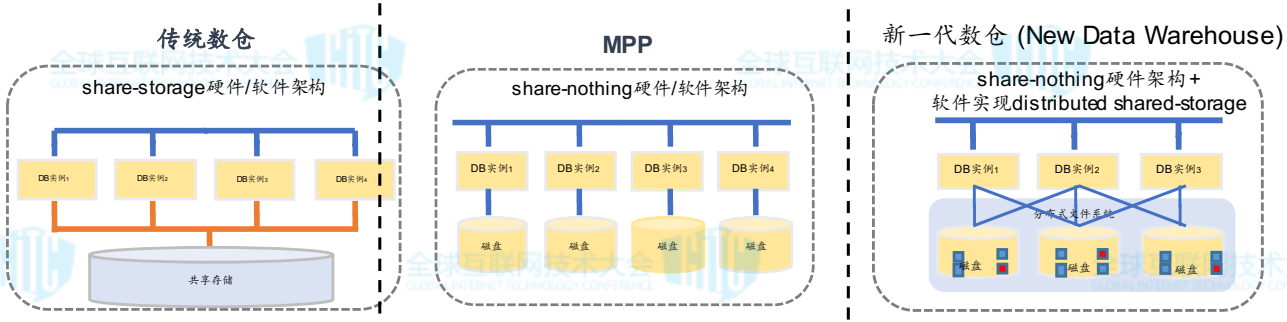
- Key-Value
  - Cassandra: CQL
  - HBase: API
- Graph Model
  - Neo4j
  - Giraph/Pregel
- Tree
  - XML Database
  - MongoDB
- Streaming



其他分类方法

- 事务处理 vs 分析处理
- 并行 vs 串行
- 硬件： CPU vs GPU vs FPGA vs Memory
- 云数据库 vs 非云数据库？

数据仓库的演进



	传统数仓	MPP	新一代数仓 (New Data Warehouse)
硬件配置	大多专有硬件平台	大多工业标准的x86服务器	工业标准的x86服务器
适用场景	面向传统的BI分析	复杂的计算需求 / 面向传统BI分析	面向大数据和人工智能 / 支持数据湖
架构	缺乏弹性 / 不易调整	缺乏弹性 / 不易调整	弹性伸缩，支持CaaS平台 / 灵活配置
可扩展性	十几个节点	几十个节点	上千个节点
数仓代表	Oracle, DB2	Teradata, Vertica, Greenplum, Redshift	Hive, HAWQ, SparkSQL, Snowflake

数据仓库引擎比较

NewDW的细分类别

- SQL on Hadoop
  - SparkSQL, Hive, HAWQ 2.x, Presto
- SQL on Object Store
  - Snowflake (on S3), Amazon Athena (on S3)
- Hybrid:有自己的存储，对外部存储可插拔
  - HAWQ 3.x, Oushu Database
  - Impala

NewDW特性比较

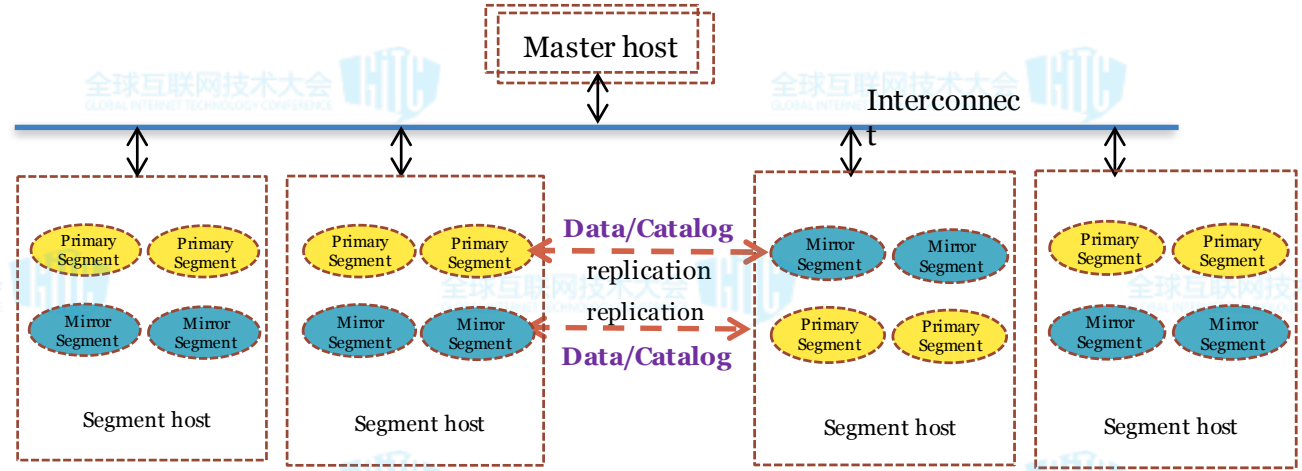
	SQL on Hadoop			SQL on Object Store		SQL on Hybrid Storage		
Features	Hive	SparkSQL	Presto	Snowflake	Athena	HAWQ	Oushu	Impala
性能	low	middle	low	low	low	high	top	middle
可扩展性	high	high	high	high	high	high	high	high
Update/Delete	bad	N/A	N/A	weak	N/A	N/A	Good	weak
索引	bad	N/A	N/A	N/A	N/A	N/A	Yes	weak
SQL兼容性	middle	middle	bad	middle	bad	good	good	middle
高并发查询	no	no	no	no	no	no	yes	no

HAWQ

HAWQ主要发展历程

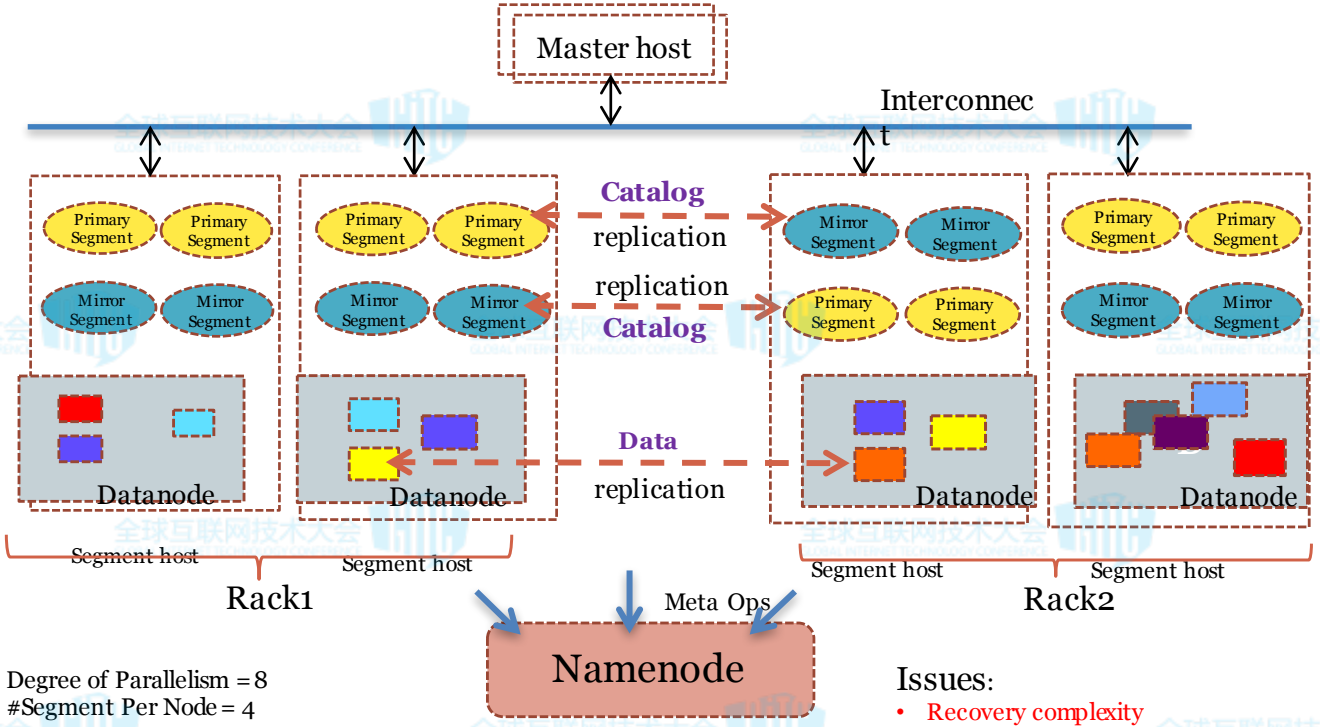
- 2011年 - 常雷博士在EMC/Pivotal提出创意，HAWQ项目启动。
- 2013年 - HAWQ 1.0发布，性能是Hive的数百倍。
- 2014年 - HAWQ SIGMOD论文发表，得到国际数据库界认可。
- 2014年 - HAWQ为全球多家大型企业客户采用。
- 2015年 - HAWQ开源成为Apache项目。
- 2016年 - 常雷博士及HAWQ核心团队创立偶数科技。
- 2017年 - 偶数得到国际顶级VC投资，致力于HAWQ的发展。
- 2017年 - Oushu Database 3.0 企业版本发布，全新执行器，世界上最快的数据仓库 **10倍性能提升**

Greenplum database (2003)



Degree of Parallelism = 8
#Segment Per Node = 4

HAWQ Alpha: Greenplum Database on HDFS (2011)

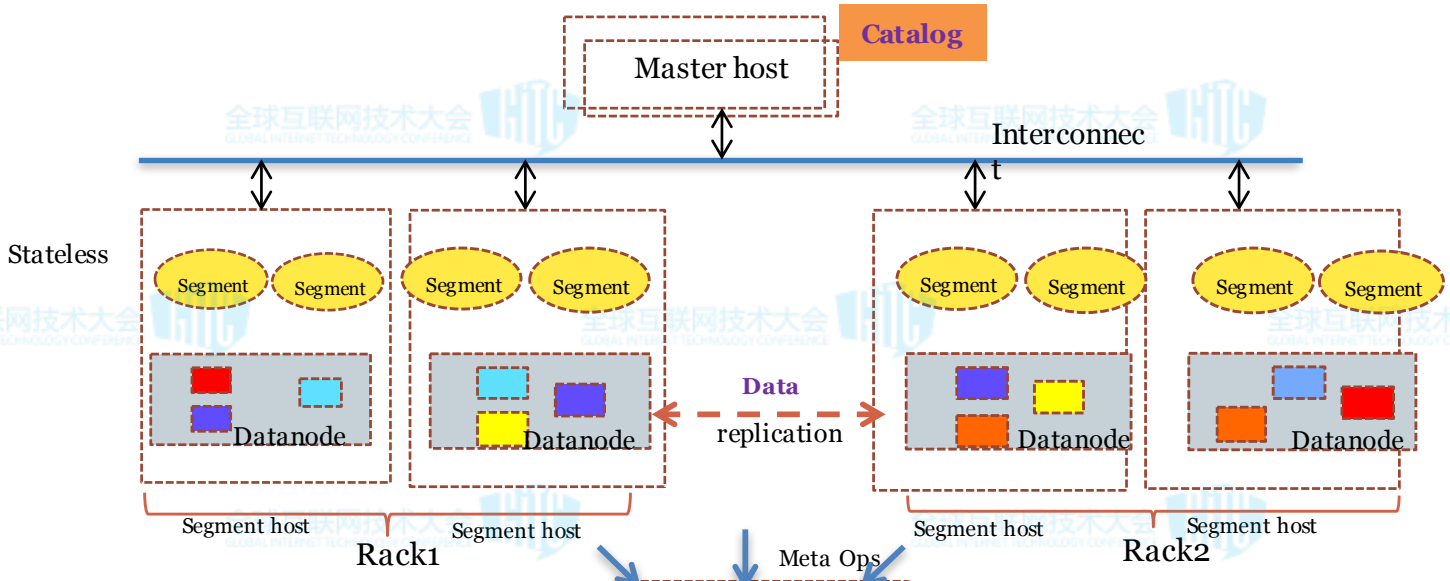


- Degree of Parallelism = 8
- #Segment Per Node = 4

Issues:

- Recovery complexity
- Expansion complexity
- Management complexity (many segments per node)
- Fixed Degree of Parallelism

HAWQ 1.0 GA Architecture (2013)



- Degree of Parallelism = 8
- **#Segment Per Node = 2**

Issues:

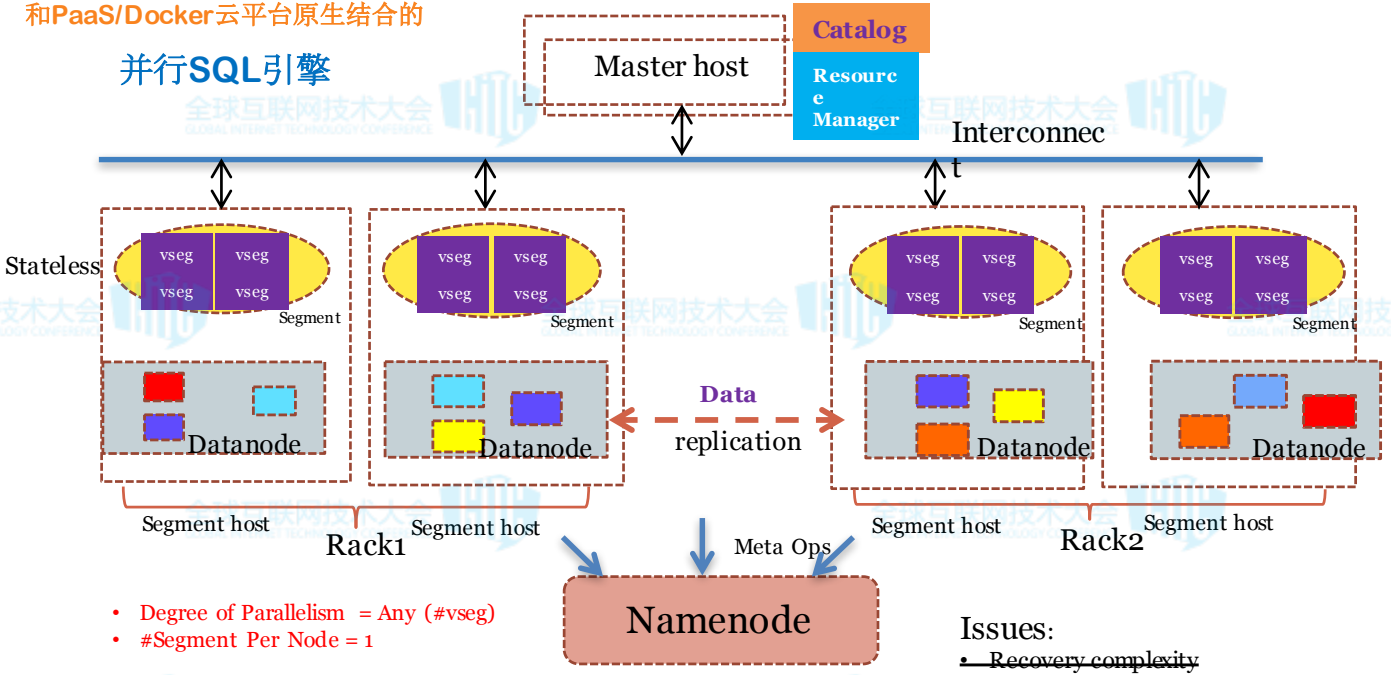
- ~~Recovery complexity~~
- Expansion complexity
- Management complexity (many segments per node)
- Fixed Degree of Parallelism

HAWQ 2.0: Architecture Change (2016 Q2)

世界上第一个

和PaaS/Docker云平台原生结合的

并行SQL引擎

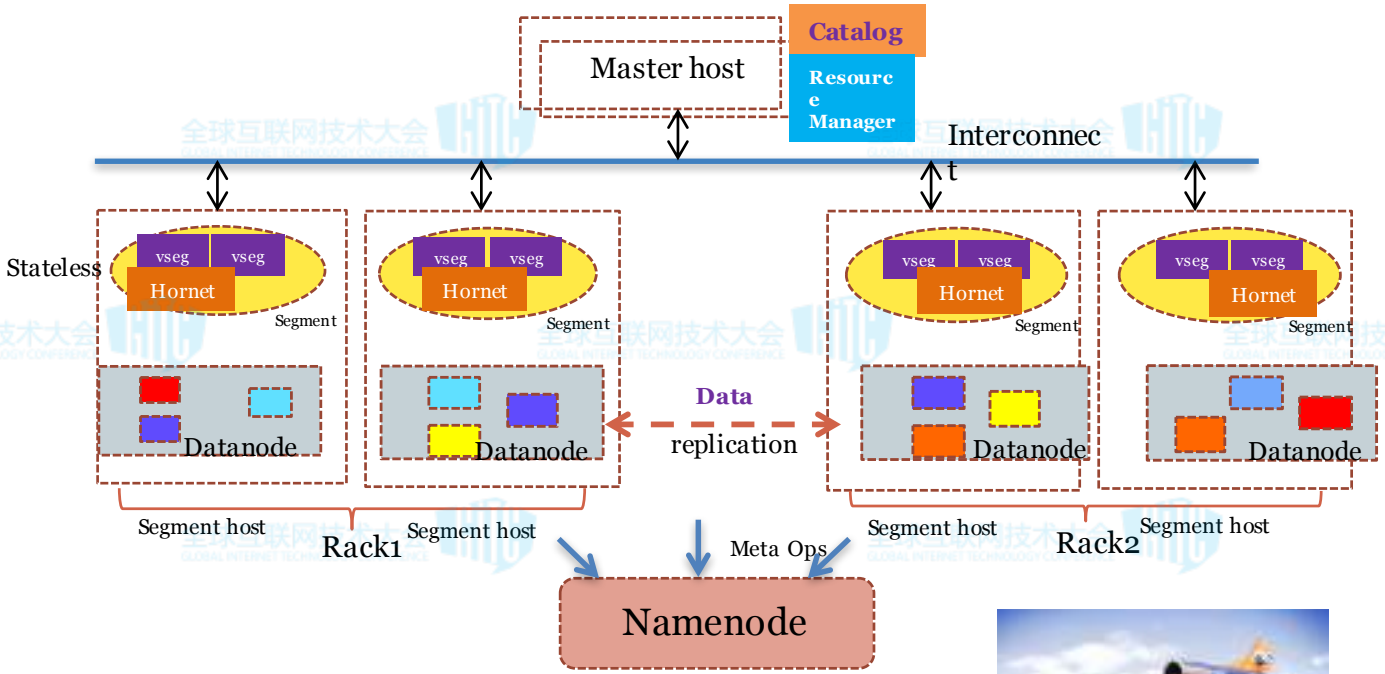


- Degree of Parallelism = Any (#vseg)
- #Segment Per Node = 1

Issues:

- ~~Recovery complexity~~
- ~~Expansion complexity~~
- ~~Management complexity (many segments per node)~~
- ~~Fixed Degree of Parallelism~~

HAWQ++ 3.0: Hornet Execution Engine (2017 Q3)



Hornet Execution Engine: SIMD/New hardware

The Fastest Engine in the World

10 times faster

Oushu Database 3.0 vs SparkSQL 2.2

单位（毫秒ms）	Oushu	Spark	ratio
select count(*) from lineitem;	21.28	2555	120.06
select count(*) from lineitem;	22.77	2440	107.16
AVERAGE	22.03	2497.50	113.61

count 不同数据类型的列

单位（毫秒ms）	Oushu	Spark	Ratio
select count(l_orderkey) from lineitem;	306.70	3925	12.80
select count(l_partkey) from lineitem;	274.35	3674	13.39
select count(l_suppkey) from lineitem;	244.77	3466	14.16
select count(l_linenumber) from lineitem;	133.67	3265	24.43
select count(l_quantity) from lineitem;	110.12	3689	33.50
select count(l_extendedprice) from lineitem;	112.05	3627	32.37
select count(l_discount) from lineitem;	108.64	3886	35.77
select count(l_tax) from lineitem;	115.14	3723	32.33
select count(l_returnflag) from lineitem;	70.41	4591	65.20
select count(l_linestatus) from lineitem;	73.01	4208	57.64
select count(l_shipdate) from lineitem;	127.12	4218	33.18
select count(l_commitdate) from lineitem;	135.43	4506	33.27
select count(l_receiptdate) from lineitem;	134.36	4193	31.21
select count(l_shipinstruct) from lineitem;	236.63	4311	18.22
select count(l_shipmode) from lineitem;	177.66	4173	23.49
select count(l_comment) from lineitem;	344.94	5885	17.06
AVERAGE	169.06	4083.75	29.88

sum/avg 不同数据类型的列

单位（毫秒ms）	Oushu	Spark	Ratio
select sum(l_orderkey) from lineitem;	323.16	3414	10.56
select sum(l_partkey) from lineitem;	298.30	3321	11.13
select sum(l_suppkey) from lineitem;	263.69	3243	12.30
select sum(l_linenumber) from lineitem;	154.20	3193	20.71
select sum(l_quantity) from lineitem;	128.39	4004	31.19
select sum(l_extendedprice) from lineitem;	138.48	4042	29.19
select sum(l_discount) from lineitem;	141.68	3500	24.70
select sum(l_tax) from lineitem;	143.07	3536	24.72
select avg(l_orderkey) from lineitem;	327.68	3511	10.71
select avg(l_partkey) from lineitem;	303.51	3583	11.81
select avg(l_suppkey) from lineitem;	269.36	3331	12.37
select avg(l_linenumber) from lineitem;	161.41	3196	19.80
select avg(l_quantity) from lineitem;	131.92	3614	27.40
select avg(l_extendedprice) from lineitem;	138.48	3554	25.66
select avg(l_discount) from lineitem;	134.01	3618	27.00
select avg(l_tax) from lineitem;	137.92	3549	25.73
AVERAGE	199.70	3513.06	20.31

group by (某一列) 取count

单位（毫秒ms）	Oushu	Spark	Ratio
select l_orderkey, count(*) from lineitem group by l_orderkey;	14314.14	OOM	NAN
select l_partkey, count(*) from lineitem group by l_partkey;	4127.98	29299	7.10
select l_suppkey, count(*) from lineitem group by l_suppkey;	1142.61	18181	15.91
select l_linenumber, count(*) from lineitem group by l_linenumber;	363.51	9570	26.33
select l_quantity, count(*) from lineitem group by l_quantity;	370.15	11367	30.71
select l_extendedprice, count(*) from lineitem group by l_extendedprice;	4929.78	29736	6.03
select l_discount, count(*) from lineitem group by l_discount;	392.41	10371	26.43
select l_tax, count(*) from lineitem group by l_tax;	352.99	10371	29.38
select l_returnflag, count(*) from lineitem group by l_returnflag;	545.86	11346	20.79
select l_linestatus, count(*) from lineitem group by l_linestatus;	329.30	11217	34.06
select l_shipdate, count(*) from lineitem group by l_shipdate;	638.51	16077	25.18
select l_commitdate, count(*) from lineitem group by l_commitdate;	642.31	16161	25.16
select l_receiptdate, count(*) from lineitem group by l_receiptdate;	647.12	15649	24.18
select l_shipinstruct, count(*) from lineitem group by l_shipinstruct;	823.09	11539	14.02
select l_shipmode, count(*) from lineitem group by l_shipmode;	630.63	11371	18.03
select l_comment, count(*) from lineitem group by l_comment;	39032.16	OOM	NAN
AVERAGE(除去spark OOM 语句)	1138.30	15161.07	21.66

group by 不同数据类型的列,取其sum和avg

单位（毫秒ms）	Oushu	Spark	Ratio
select l_partkey, sum(l_partkey), avg(l_partkey) from lineitem group by l_partkey;	8333.37	54470	6.54
select l_suppkey, sum(l_suppkey), avg(l_suppkey) from lineitem group by l_suppkey;	1527.32	19505	12.77
select l_linenumber, sum(l_linenumber), avg(l_linenumber) from lineitem group by l_linenumber;	416.03	9914	23.83
select l_quantity, sum(l_quantity), avg(l_quantity) from lineitem group by l_quantity;	390.82	11949	30.57
select l_extendedprice, sum(l_extendedprice), avg(l_extendedprice) from lineitem group by l_extendedprice;	9148.20	32005	3.50
select l_discount, sum(l_discount), avg(l_discount) from lineitem group by l_discount;	418.81	10757	25.68
select l_tax, sum(l_tax), avg(l_tax) from lineitem group by l_tax;	357.99	10733	29.98
AVERAGE	2941.79	21333.29	18.98

Group by 多列

单位（毫秒ms）	Oushu	Spark	Ratio
select l_partkey, l_suppkey, count(*) from lineitem group by l_partkey, l_suppkey;	13074.79	OOM	NAN
select l_partkey, l_linenumber, count(*) from lineitem group by l_partkey, l_linenumber;	18091.03	OOM	NAN
select l_suppkey,l_extendedprice, count(*) from lineitem group by l_suppkey,l_extendedprice;	145543.51	OOM	NAN
select l_partkey, l_shipmode, count(*) from lineitem group by l_partkey, l_shipmode;	21298.14	OOM	NAN
select l_partkey, l_shipdate, count(*) from lineitem group by l_partkey, l_shipdate;	71890.82	OOM	NAN
select l_suppkey, l_tax, count(*) from lineitem group by l_suppkey, l_tax;	3994.25	28334	7.09
select l_shipdate,l_commitdate, count(*) from lineitem group by l_shipdate,l_commitdate;	3159.43	32811	10.39
select count(l_orderkey) from lineitem group by l_linenumber , l_quantity , l_tax;	1179.85	18080	15.32
AVERAGE	2777.84	26408.33	10.93

Group by 表达式

单位（毫秒ms）	Oushu	Spark	Ratio
select l_partkey + l_suppkey, count(*) from lineitem group by l_partkey + l_suppkey;	4050.55	31601	7.80
select l_partkey + 1000 from lineitem group by l_partkey +1000;	2869.51	27083	9.44
select l_tax * 100 from lineitem group by l_tax*100;	426.14	10005	23.48
AVERAGE group by 表达式	2448.73	22896.33	13.57

多个聚集函数

单位（毫秒ms）	Oushu	Spark	Ratio
select l_partkey, count(*), count(l_orderkey),sum(l_orderkey), avg(l_orderkey) from lineitem group by l_partkey;	11878.22	OOM	NAN
select l_suppkey,count(*),count(l_orderkey) , sum(l_orderkey),avg(l_orderkey) from lineitem group by l_suppkey;	2399.98	23745	9.89
select l_linenumber,count(*),count(l_orderkey) ,sum(l_orderkey),avg(l_orderkey) from lineitem group by l_linenumber;	698.18	10943	15.67
select l_quantity,count(*),count(l_orderkey) , sum(l_orderkey),avg(l_orderkey) from lineitem group by l_quantity;	702.60	13496	19.21
select l_discount,count(*),count(l_orderkey) , sum(l_orderkey),avg(l_orderkey) from lineitem group by l_discount;	741.17	12668	17.09
select l_tax,count(*),count(l_orderkey) , sum(l_orderkey),avg(l_orderkey) from lineitem group by l_tax;	670.63	12046	17.96
select l_returnflag,count(*),count(l_orderkey) , sum(l_orderkey),avg(l_orderkey) from lineitem group by l_returnflag;	913.23	12812	14.03
select l_linestatus,count(*),count(l_orderkey) , sum(l_orderkey),avg(l_orderkey) from lineitem group by l_linestatus;	675.94	12444	18.41
select l_shipdate,count(*),count(l_orderkey), sum(l_orderkey),avg(l_orderkey) from lineitem group by l_shipdate;	1025.86	17846	17.40
select l_shipmode,count(*),count(l_orderkey) , sum(l_orderkey),avg(l_orderkey) from lineitem group by l_shipmode;	...	...	...
select l_comment,count(*),count(l_orderkey) , sum(l_orderkey),avg(l_orderkey) from lineitem group by l_comment;	117636.74	OOM	NAN
AVERAGE	1722.58	17189.46	14.97

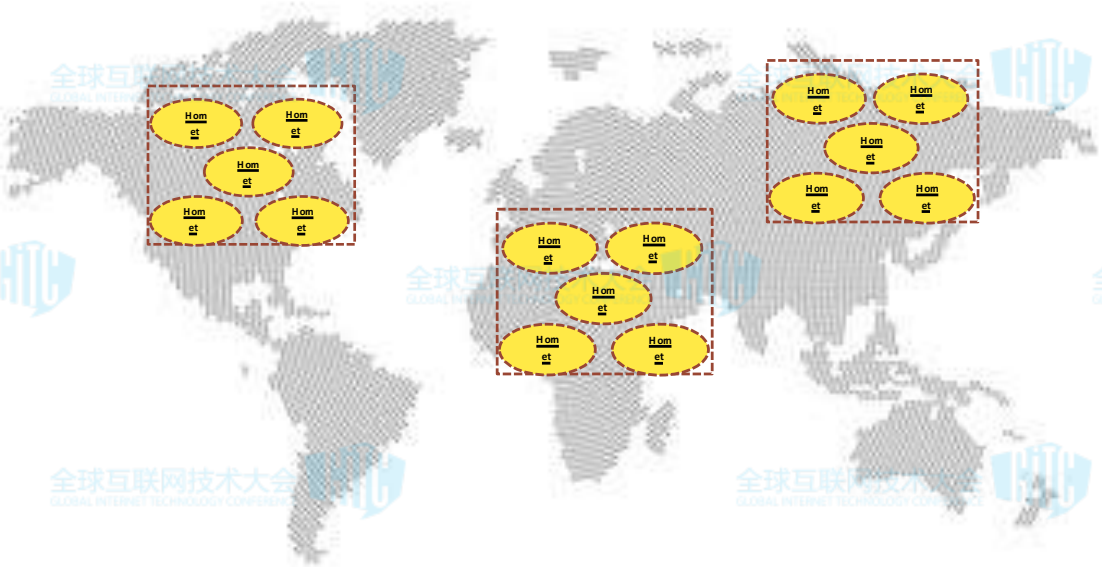
TPCH Query

单位（毫秒ms）	Oushu	Spark	Ratio
TPCHQ1	1175.99	18626	15.84
TPCHQ1	1140.01	18060	15.84
TPCHQ1	1161.93	18096	15.57
AVERAGE	1159.31	18260.67	15.75

TPCH_Q1

```
select
    l_returnflag,
    l_linestatus,
    sum(l_quantity) as sum_qty,
    sum(l_extendedprice) as sum_base_price,
    sum(l_extendedprice * (1 - l_discount)) as sum_disc_price,
    sum(l_extendedprice * (1 - l_discount) * (1 + l_tax)) as sum_charge,
    avg(l_quantity) as avg_qty,
    avg(l_extendedprice) as avg_price,
    avg(l_discount) as avg_disc,
    count(*) as count_order
from
    lineitem_1gorc_none
where
    l_shipdate <= '1998-08-20'
group by
    l_returnflag,
    l_linestatus;
```

Oushu Database 4.0: Global Scale (2017 H1)



Global Scale: No master, P2P, Geo-replication, mixed workload

HAWQ全球用户(部分)

某大型制造企业案例

背景

- 大量传感器数据无法及时处理
- 故障无法及时检测带来很大损失
- 传统解决方案过于昂贵

实现目标

- 搭建大数据平台，提高其处理处理能力
- 200+节点分析平台集群
- PB级数据存储
- 实现实时故障预测等应用

某大型证券交易所

- 挑战
  - 为了应对每天增长的交易量，替换现有Oracle EDW平台
  - 为了合规需要保存最细力度的交易数据
  - 经济有效的方式保证每天处理TB级别增量数据
- 解决方案
  - 把所有交易数据放入Hadoop和HAWQ
  - 把12亿条记录放到HAWQ里面进行查询分析，获得更好的性能

偶数科技简介

- EMC/Pivotal HAWQ创始人及HAWQ核心团队成员创立
- 偶数两大数据仓库/AI产品
  - Oushu Database (HAWQ++)
  - Apache HAWQ
- 成员大多为Apache Committer & PMC成员，来自各大云计算和大数据公司: EMC/Pivotal, Oracle, IBM, Teradata等
- 毕业于国内外顶级学府，多个ACM程序设计大赛奖牌得主
- 团队研究成果发布在国际顶级数据管理会议上（比如SIGMOD等），并拥有多项国际专利
- 获得国际顶级VC投资：红点和红杉

谢谢！