

数据来源：数据库产品上市商用时间



# 第十三届中国数据库技术大会

DATABASE TECHNOLOGY CONFERENCE CHINA 2022

## 数据智能 价值创新



线上直播 | 2022/12/14-16



# 程序员必须掌握的数据库原理

叶正盛

玖章算术CEO

[www.ninedata.cloud](http://www.ninedata.cloud)

让每个人轻松用好数据和云

# 叶正盛

## | 玖章算术CEO 创始人



- 资深数据库与云计算领域专家，精通云计算、数据库、企业级软件开发、互联网等核心技术
- 曾担任阿里云数据库产品管理与解决方案部总经理，阿里云技术架构与产品决策委员会核心成员。
- 带领阿里云数据库成功进入 Gartner DBMS 魔力象限全球领导者位置，是中国基础软件的重大突破
- 阿里巴巴去 IOE、异地多活、云计算多次技术变革核心成员
- 带领团队构建阿里巴巴&蚂蚁集团数据库业务研发支撑体系
- 创立了全球领先的云计算数据传输、数据管理、数据库备份、数据库自动驾驶服务等多款云计算数据库产品

# 目录

## CONTENTS

01 数据库简介

02 数据库内部结构

03 数据库设计与SQL优化

04 数据传输与安全管理

# 01

## 数据库简介

Introduction of database

- 操纵和管理数据库的大型软件，用于建立、使用和维护数据库
- 提供数据采集、存储、查询、分析等功能



手工记账



| 库存出入明细表 |    |    |    |      |    |      |    |      |    |      |    | 年 月 日 |  |
|---------|----|----|----|------|----|------|----|------|----|------|----|-------|--|
| 序号      | 品名 | 规格 | 单位 | 期初库存 |    | 本期收入 |    | 本期发出 |    | 期末库存 |    | 备注    |  |
|         |    |    |    | 数量   | 金额 | 数量   | 金额 | 数量   | 金额 | 数量   | 金额 |       |  |
| 1       |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 2       |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 3       |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 4       |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 5       |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 6       |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 7       |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 8       |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 9       |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 10      |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 11      |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 12      |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 13      |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 14      |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 15      |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 16      |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 17      |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 18      |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 19      |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 20      |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 21      |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 22      |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 23      |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 24      |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 25      |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 26      |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 27      |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 28      |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 29      |    |    |    |      |    |      |    |      |    |      |    |       |  |
| 30      |    |    |    |      |    |      |    |      |    |      |    |       |  |

Excel管理



数据库系统

数据库管理

## 数据库 VS Excel

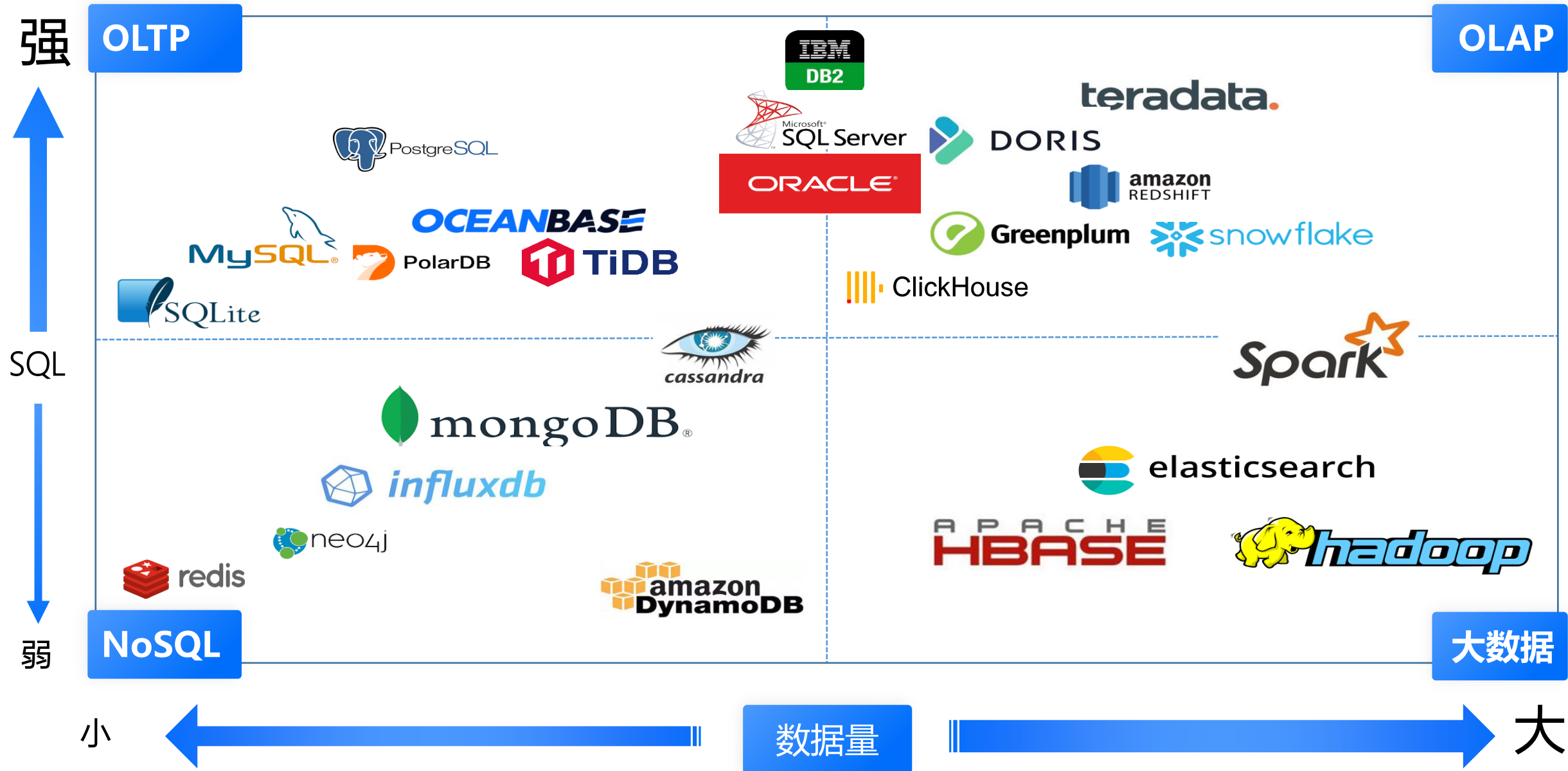
管理  
大数据

多用户  
高性能

数据  
安全管理

灵活的  
开发接口

# 数据库系统分类大图





# 02

## 数据库内部结构

Internal structure of database



## 单机模式

计算节点(RW)  
CPU+内存



存储节点  
本地磁盘

### 基础架构

MySQL  
Oracle  
SQL Server  
PostgreSQL

## 主备读写分离

计算节点(RW)  
CPU+内存

计算节点(R)  
CPU+内存  
Log同步



存储节点  
本地磁盘



存储节点  
本地磁盘

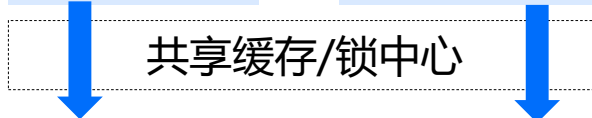
### log实时传输与应用 主备状态管理、读写分离

MySQL(Master-Slave)  
Oracle(Active Data Guard)  
SQL Server(Always on)

## 分布式(share everything)

计算节点(RW)  
CPU+内存

计算节点(R/RW)  
CPU+内存



共享缓存/锁中心

存储节点  
共享存储设备/云存储

### 共享存储、RDMA 缓存同步、MPP

Oracle RAC  
阿里云PolarDB  
AWS Aurora  
DB2 Parallel Sysplex(大型机)  
DB2 pureScale  
Snowflake(云存储)  
AWS Redshift(RA3)

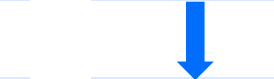
## 分布式(Share nothing)

计算节点  
(RW)  
CPU+内存

计算节点  
(RW)  
CPU+内存



存储节点  
本地磁盘



存储节点  
本地磁盘

### 数据分片 MPP、分布式事务

MongoDB  
OceanBase  
Greenplum  
ClickHouse  
TiDB  
AWS Redshift

## 外部接口（通讯协议）

JDBC、ODBC、OLEDB...

## 会话管理

连接、状态、配置

## 计算引擎（查询引擎）

SQL、存储过程、执行计划、执行算子

## 缓存管理

数据、META、SQL、排序、JOIN

## 存储引擎

数据、索引、分区、日志

## 事务处理

ACID  
锁

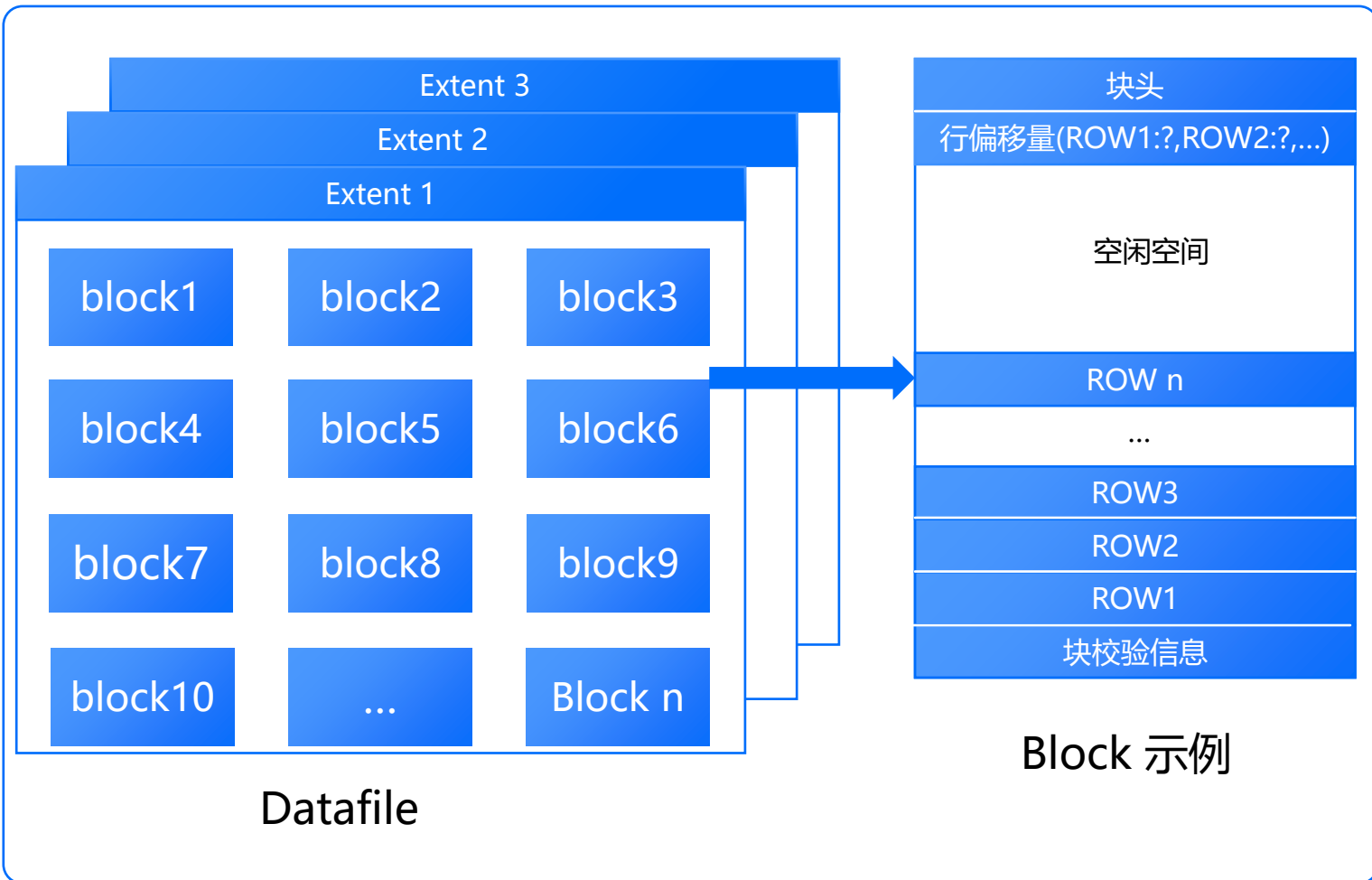
## 分布式管理

通讯 事务  
同步 计算  
存储

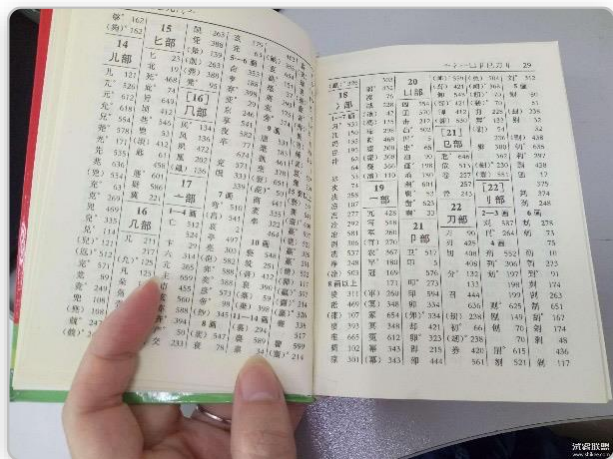
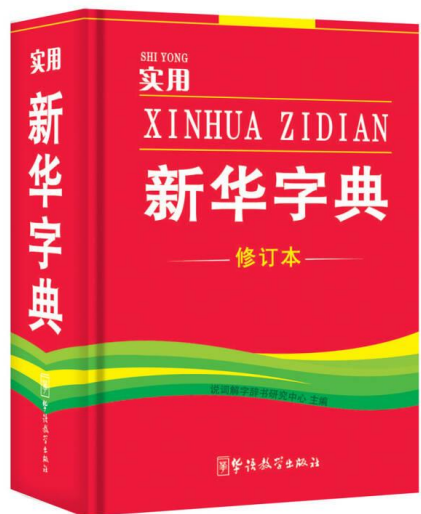
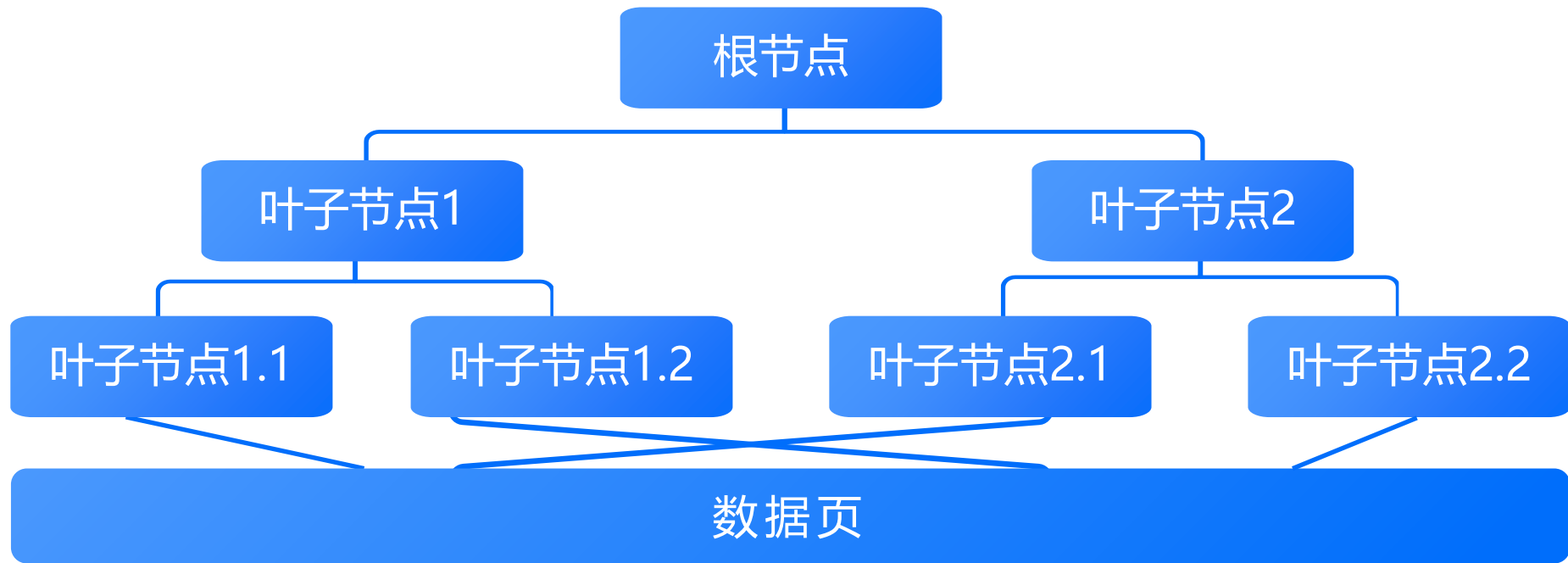
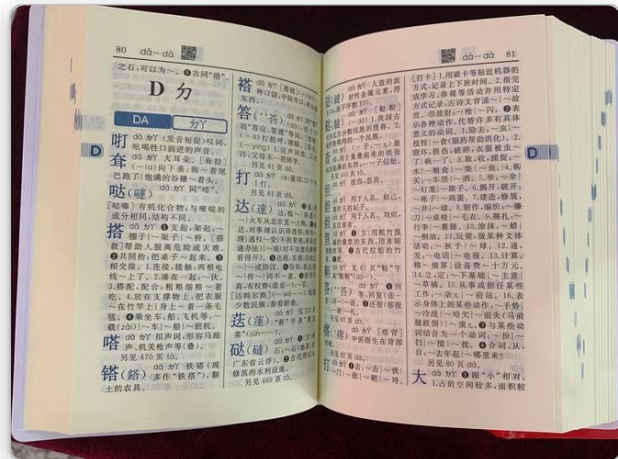
## 安全管理

用户 权限  
角色 认证  
加密 脱敏

- Oracle(堆表)
- SQL Server(堆表)
- MySQL(MyISAM)
- PostgreSQL

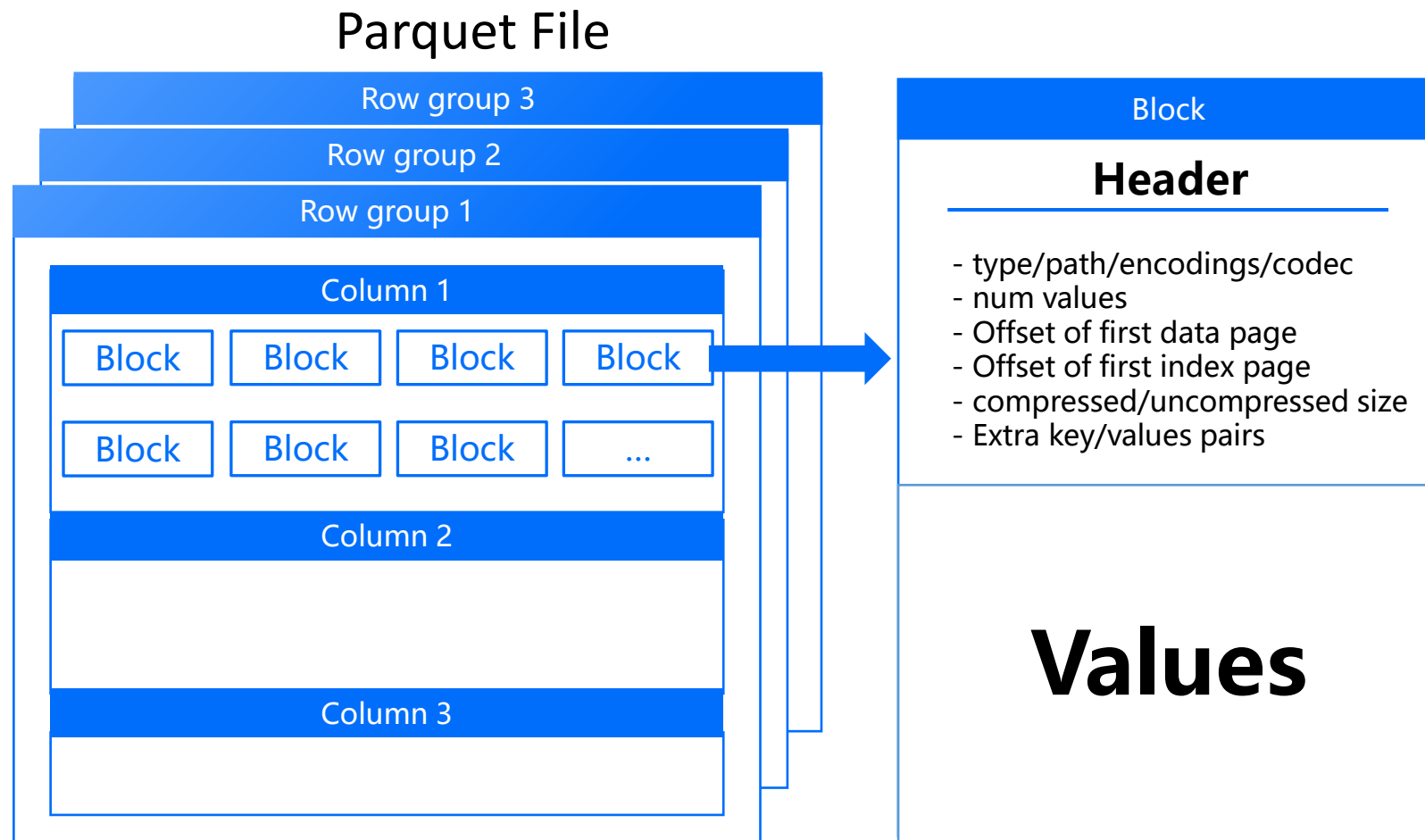


- Oracle
- SQL Server
- PostgreSQL
- MySQL
- MongoDB

[illegible]

- ORC
- PARQUET

- Greenplum
- ClickHouse
- Snowflake
- MySQL(Infobright)
- MySQL(HeatWave)



示例：超市货架管理

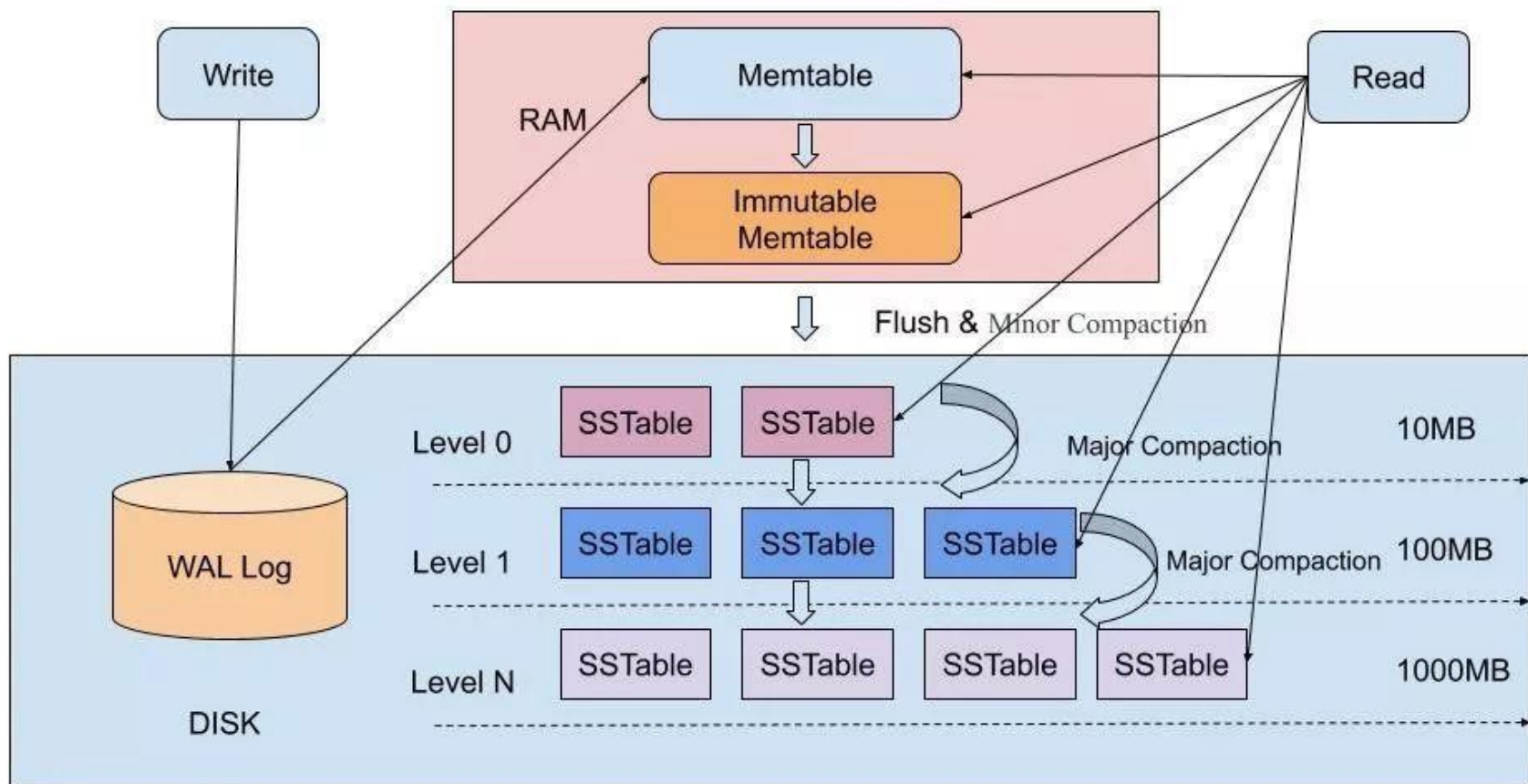


# 存储引擎 (LSM-TREE)

顺序写、日志存储、大数据量管理

- BigTable
- HBase
- OceanBase
- LevelDB
- RocksDB
- TiDB

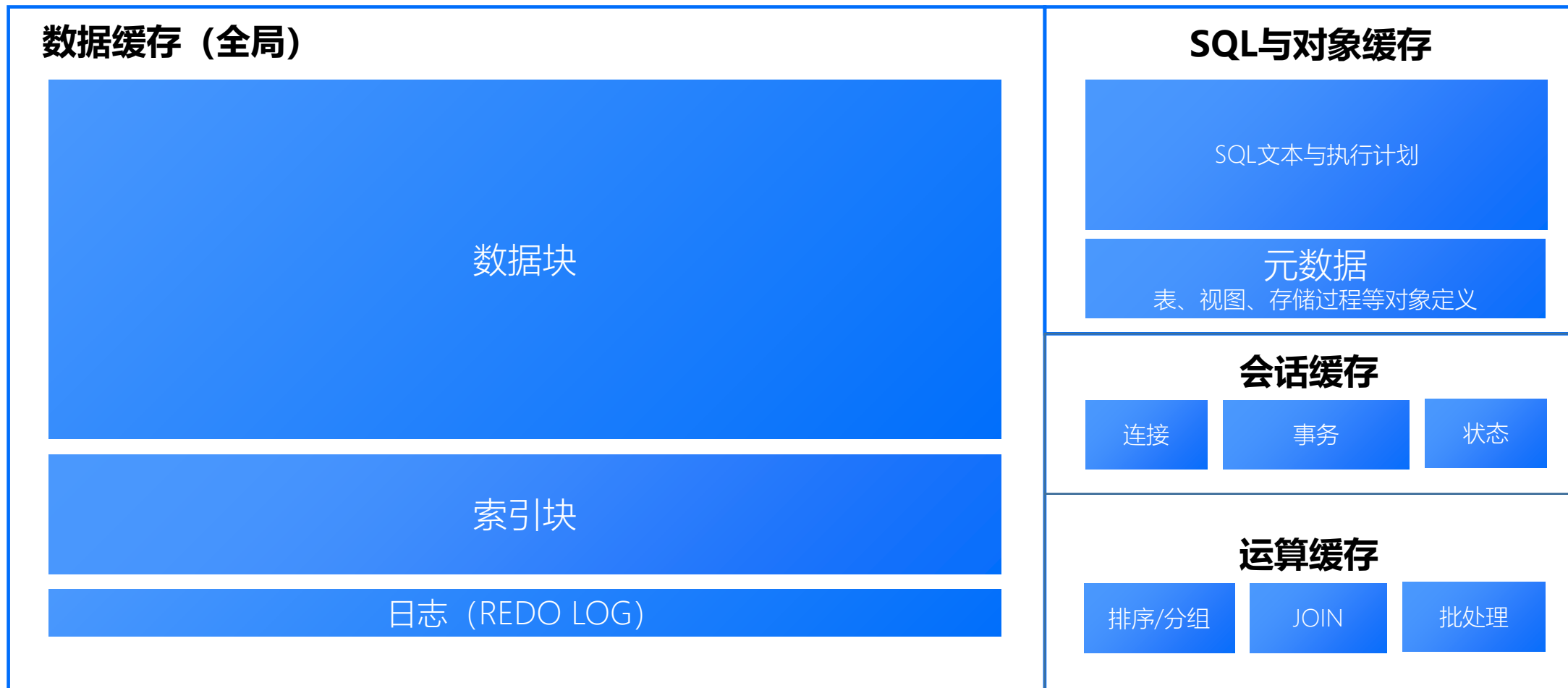
## Log Structured Merge Trees



图片来自X-DB LSM存储架构

| 类型           | 优势                      | 劣势                    | 应用场景              | 数据库   |
|--------------|-------------------------|-----------------------|-------------------|---|
| HEAP         | 基本数据结构<br>结构简单<br>写入方便  | 没有查询优化                | OLTP表数据           | Oracle(堆表)<br>MySQL(ISAM)<br>SQL Server(堆表)<br>PostgreSQL         |
| B+TREE       | 小数据量查询性能高<br>响应时间稳定     | 全表查询性能差<br>维护复杂       | OLTP索引<br>OLTP索引表 | Oracle(索引组织表)<br>MySQL(InnoDB)<br>SQL Server(Cluster表)<br>MongoDB |
| COLUMN-STORE | 压缩存储<br>效率高<br>按字段查询性能好 | 单行查询差<br>实时更新难        | OLAP              | ClickHouse<br>Snowflake<br>Greenplum                              |
| LSM-TREE     | 写入友好<br>可压缩存储           | 查询效率差 (二级索引,<br>范围查询) | 日志写入<br>大数据量管理    | HBase<br>TiDB<br>OceanBase  |





OLTP: 数据缓存非常重要

OLAP: 运算缓存非常重要

## 4种标准事务隔离级别

|           | Read Uncommitted                   | Read Committed  | Repeatable Read   | Serializable Read           |
|-----------|------------------------------------|---|-------------------|-----------------------------|
| 读取到未提交的数据 | 会                                  | 不会  | 不会                | 不会                          |
| 不可重复读     | 会                                  | 会   | 不会                | 不会                          |
| 幻读        | 会                                  | 会   | 不会?               | 不会                          |
| 默认数据库     | SQL Server<br>Synapse<br>Analytics | Oracle<br>DB2<br>SQL Server<br>PostgreSQL<br>Greenplum<br>Snowflake | MySQL<br>(InnoDB) | AWS<br>REDSHIFT<br>Teradata |

OLTP数据库：建议默认采用Read committed

### Atomicity

WAL(redo log)  
硬件

### Consistency

主外键  
约束

### Isolation

快照MVCC  
锁

### Durability

WAL(redo log)  
Undo log

# 查询引擎（查询语言）

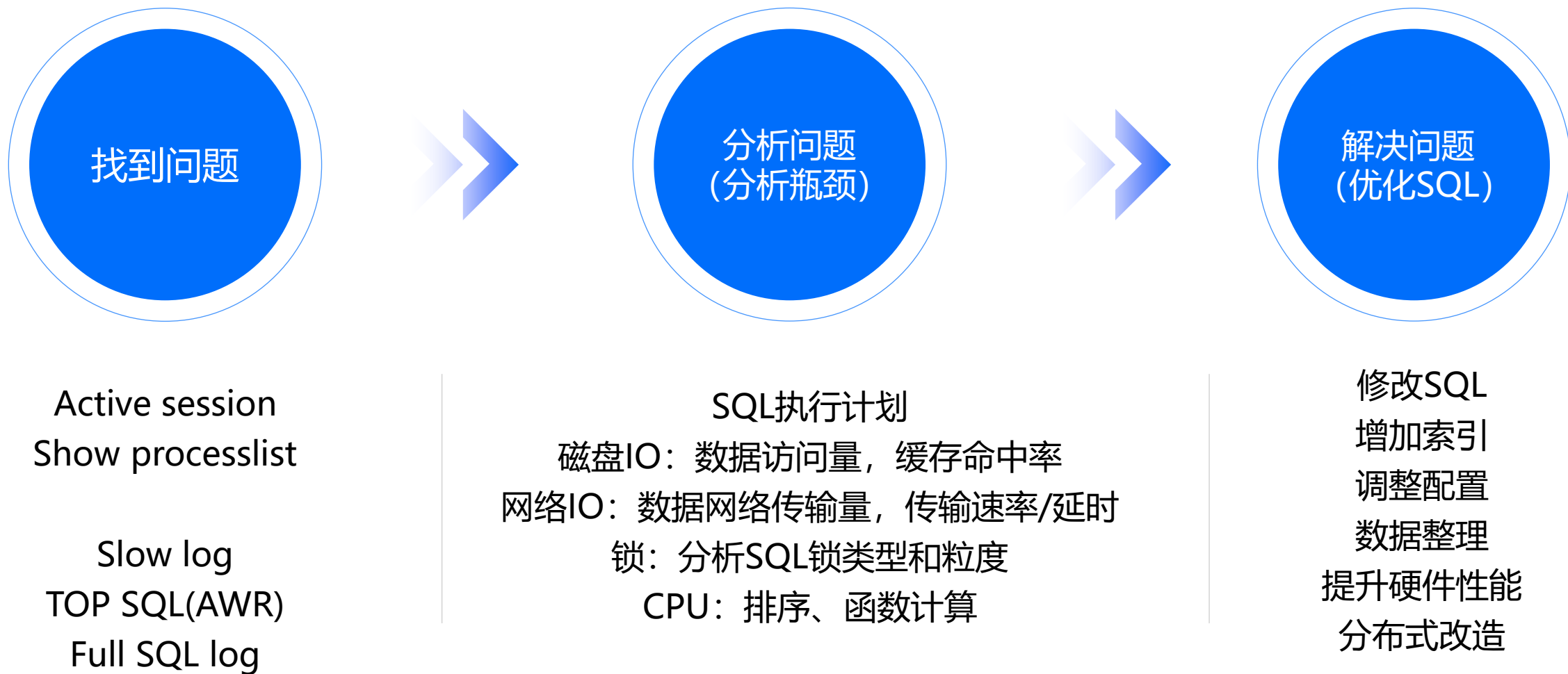
|       | 查询语言       | 支持的数据库  | 描述         |
|-------|------------|---|------------|
| 关系型   | SQL        | MySQL、Oracle、SQL Server、PG、Snowflake、Teradata、Hive    | 结构化查询语言    |
| KV    | get/set    | Redis, AWS DynamoDB                                   |            |
| 文档    | JSON       | MongoDB, Google Firestore                             |            |
|       | 类SQL       | Couchbase, MongoDB                                    |            |
| 图     | OpenCypher | Neo4j、AWS Neptune<br>TigerGraph (gSQL)<br>Nebula(nQL) | 与SQL类似     |
|       | Gremlin    | JanusGraph、AWS Neptune                                |            |
|       | SPARQL     | AWS Neptune   | 与SQL类似     |
| 时序数据库 | 类SQL       | InfluxDB、TDengine                                     | SQL+时序计算功能 |

ODBC (通用)、JDBC (java)、ADO.NET/OLE DB (.net)

## 03

## 数据库设计与SQL优化

Database Design and SQL Optimization



```
SELECT t1.id, t2.name
FROM t1
INNER JOIN t2 ON t1.id=t2.id
WHERE t1.name = 'NineData'
      AND t2.email = 'abc@ninedata.cloud'
      AND t1.status = 'deleted'
ORDER BY t2.create_time
```

---

已知:

t1.id、t2.id分别是t1和t2表的主键

t1.name、t1.status、t2.email、t2.create\_time 字段上都分别有单个字段的索引

这条SQL的执行计划? 先访问那张表, 使用哪些索引?

描述SQL的详细执行路径和算法

## MySQL 通过EXPLAIN语法 查看SQL执行计划

```
mysql> explain select * from t_user a inner join t_user1 b on a.id=b.id and a.name='abc' ;
```

| id | select_type | table | type   | possible_keys                                 | key     | key_len | ref      | rows | Extra       |
|----|-------------|-------|--------|---|---------|---------|----------|------|-------------|
| 1  | SIMPLE      | a     | ALL    | PRIMARY                                       | NULL    | NULL    | NULL     | 8    | Using where |
| 1  | SIMPLE      | b     | eq_ref | PRIMARY, t_user1_pk, t_user_id_name(20)_index | PRIMARY | 4       | yzs.a.id | 1    | NULL        |

2 rows in set (0.00 sec)

### 表访问方式

| 访问方式     | 优先级 |
|----------|-----|
| 主键查询     | 1   |
| 唯一键索引查询  | 2   |
| 普通索引等值查询 | 3   |
| 索引范围查询   | 4   |
| 全表扫描     | 5   |

### 常见表JOIN算法

| JOIN算法                          | 成本计算公式   | 适合场景                      |
|---------------------------------|--|---------------------------|
| Nested Loop Join                | $M + rows(M) * N$  | 有比较好的过滤索引定位               |
| Hash Join<br>(*Grace Hash Join) | $M + hash(M) + N + rows(N) * hashMatch(M)$                             | M和N比较大并且是等值匹配<br>(需要临时内存) |
| Broadcast Hash Join             | $M * x + hash(M) * x + N + rows(N) * hashMatch(M)$                     | 分布式查询, M比较小,N比较大          |
| Shuffle Hash Join               | $M + hash(M) + N + hash(N) + M + Hash(M) + N + rows(N) * hashMatch(M)$ | 分布式查询, M,N都比较大            |
| Merge Join                      | $M + sort(M) + N + sort(N)$  | M和N比较大(需要排序内存)            |

M:表1访问成本; N: 表2访问成本;x: 集群节点数据



# SQL执行计划（优化器）

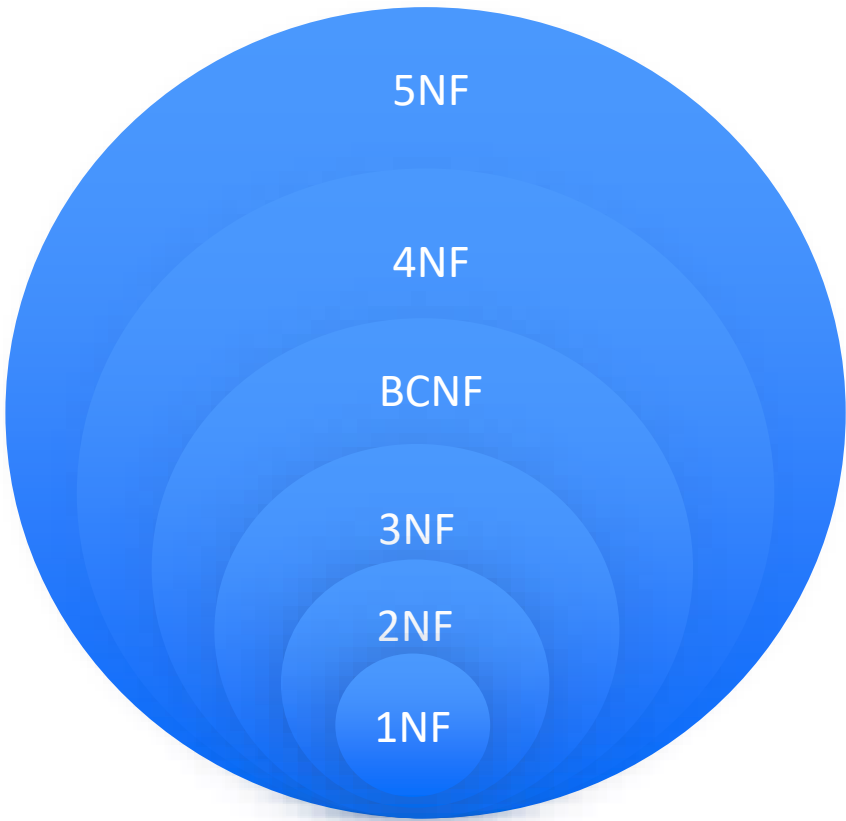
优化器：数据库大脑

| 优化器类型    | 描述                                 | 常见数据库类型                         | 备注                     |
|----------|------------------------------------|---------------------------------|------------------------|
| RBO      | 基于规则定义                             | 数据库早期实现                         | 主键、索引等值、索引范围、全表扫描      |
| CBO      | 基于成本计算（磁盘IO,CPU）<br>基于历史统计数据?(HBO) | 主流数据库默认规则                       | 最快返回首条数据?<br>最快返回全部数据? |
| Adaptive | 自适应，运行中动态选择                        | SQL Server 2017+<br>Oracle 12c+ | Join、Parallel Process  |

示例：北京->杭州，选择交通方案？


| 交通工具 | 速度排名 | 速度       | 当前交通状态      |
|------|------|----------|-------------|
| 飞机   | 1    | 800公里/小时 | 交通管制，大量航班延误 |
| 火车   | 2    | 200公里/小时 | 正常          |
| 汽车   | 3    | 80公里/小时  | 大雾天，高速限行    |
| 轮船   | 4    | 50公里/小时  | 正常          |
| 走路   | 5    | 5公里/小时   | 正常          |

## 范式



## 常见反范式设计场景

| 场景   | 描述与示例                                      | 违反范式说明                  |
|------|--|-------------------------|
| 信息组合 | 多个状态标记组合，Bit类型，101011101                   | 违反1NF，提升存储和查询效率         |
| 日常习惯 | 身份证号包括了大量信息，但通常采用一个字段保存：360111202212150034 | 违反1NF，提升可读性             |
| 计算列  | 单价*数量=金额，订单总金额                             | 违反3NF，提升性能和可读性          |
| 冗余字段 | 数据仓库中事实表大量冗余维表名称数据                         | 违反3NF，提升查询性能            |
| 历史快照 | 历史数据保留快照时间点详细信息                            | 违反2NF、3NF、BCNF<br>提升可读性 |
| 字段分表 | 根据访问场景把字段拆分的不同的表                           | 增加管理复杂度，提升查询性能          |

| 主键类型  | 优点                       | 缺点                                      | 适用场景   |
|---|--------------------------|---|--------|
| 自增ID<br>SEQUENCE  | 数据库内置功能<br>简单易用<br>存储高效  | 连续ID，容易被攻击<br>不适合分布式                    | 适合内部系统 |
| 全局SEQUENCE  | 简单易用<br>分布式专用            | 需要部署SEQUENCE服务<br>受网络性能影响<br>缓存后不一定时间有序 | 分布式系统  |
| UUID<br>示例：<br>d23b0fcf-8205-4d12-a173-edcbde904809   | 数据库内置功能<br>简单易用<br>分布式场景 | 占用存储空间大<br>容易产生随机IO<br>V1有MAC地址泄露风险     | 分布式系统  |
| 雪花算法（时间+机器+序列号）<br> | 性能良好，安全                  | 复杂，需要由应用程序产生                            | 通用     |
| 组合字段  |                          | SQL编写复杂<br>索引表的二级索引效率下降                 | 遗留设计   |

淘宝、微信公众号、微博、滴滴、抖音、美团、BOSS直聘

## 会员表

会员、买家、粉丝、乘客、求职人

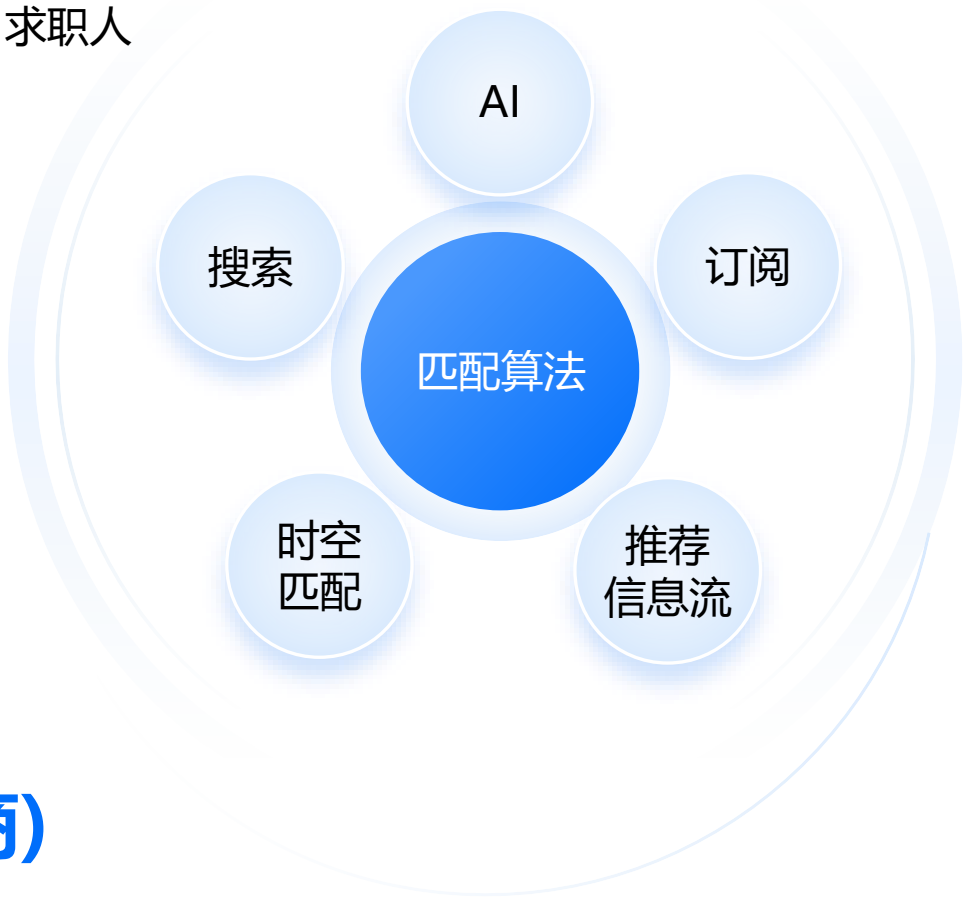
## 服务提供者表

卖家、作家、博主、司机、  
播主、公司

## 服务内容表

商品、文章、微博、用车、视频、  
岗位

## 库存记录表(电商)



## 订单表

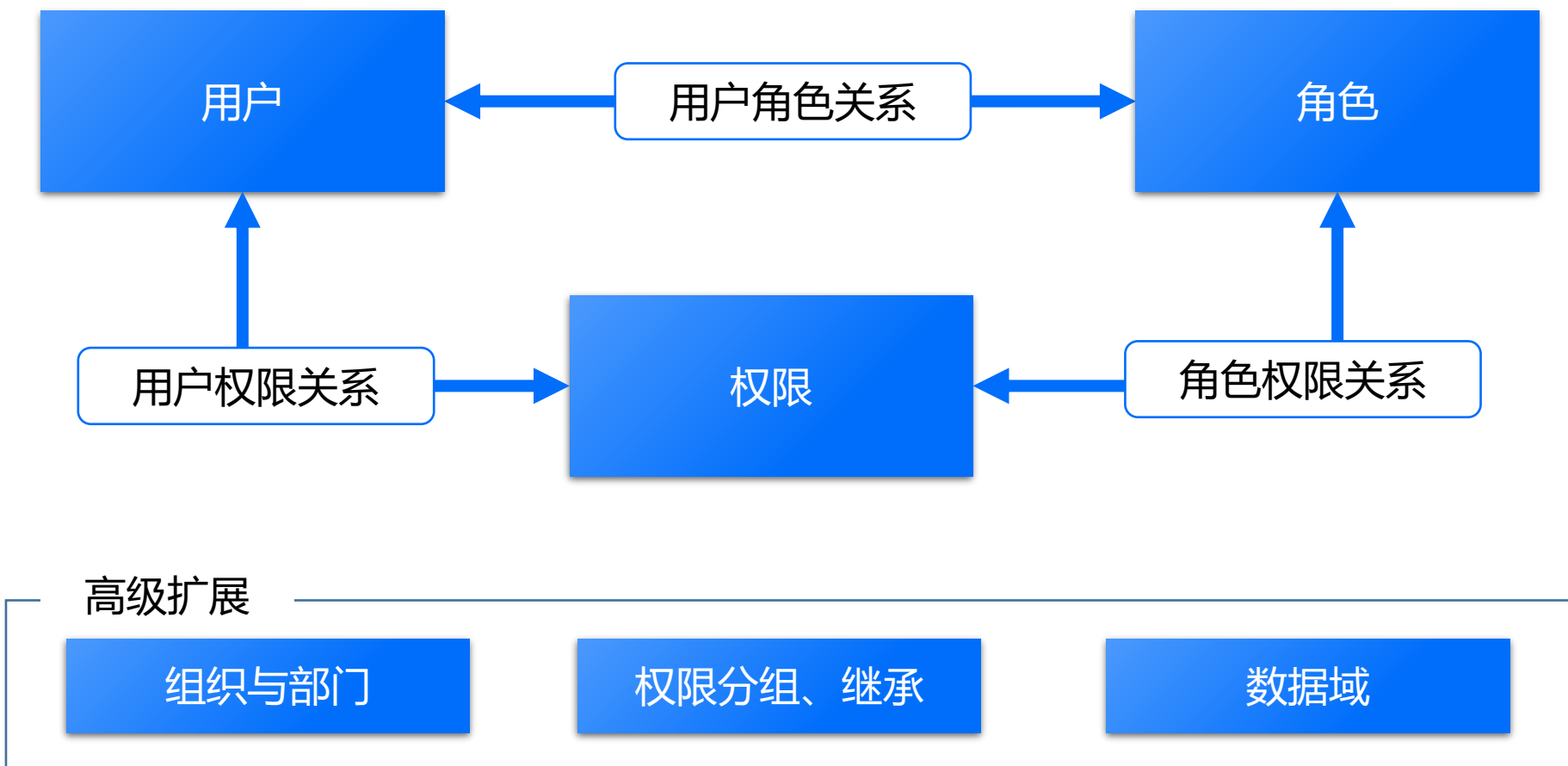
## 付款表

## 物流 / 配送表

## 服务反馈表

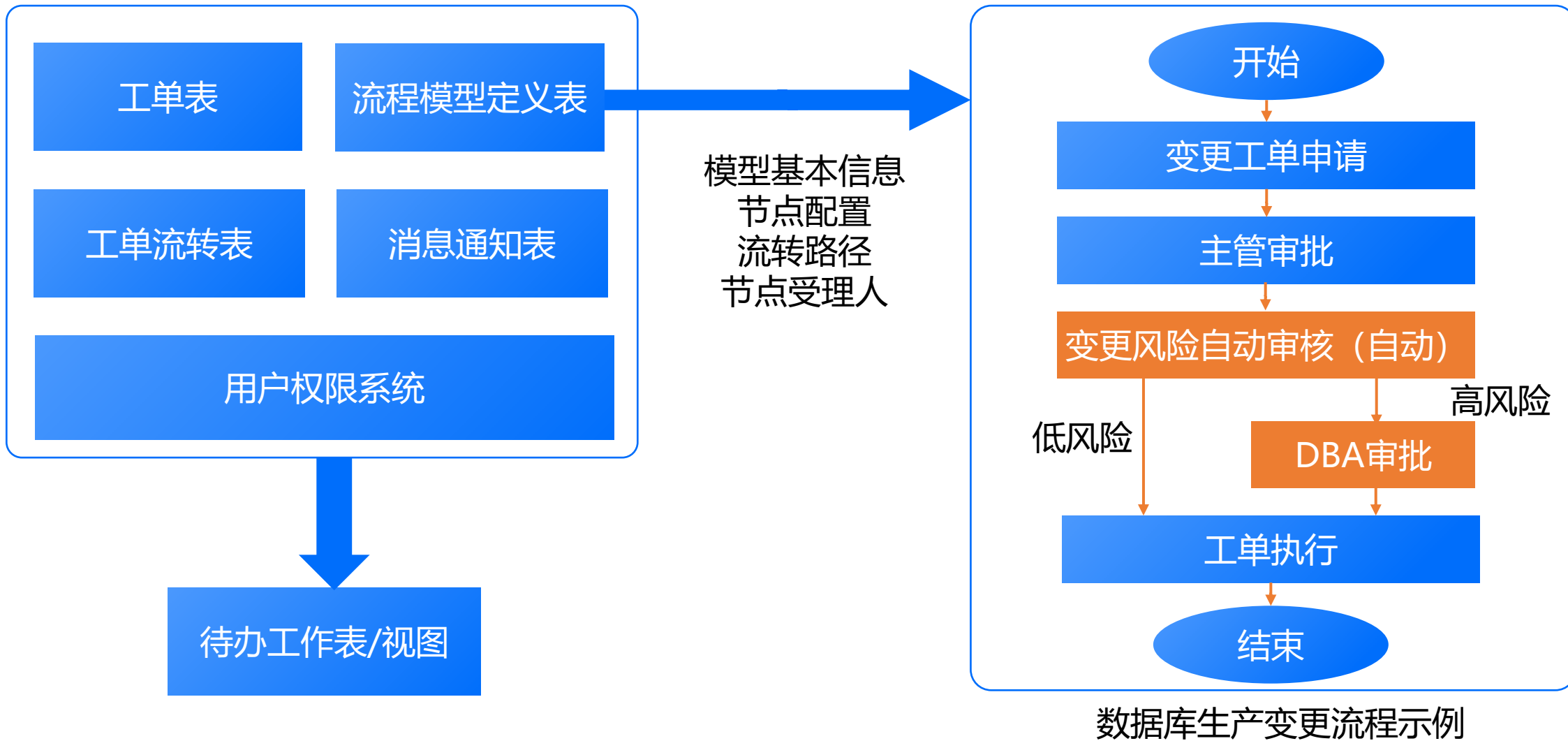
评价、点赞、留言

RBAC(Role Base Access Control)



# 企业软件-工作流程模型

请假、报销、权限申请、软件发布、数据库生产变更



# 04

## 数据传输与安全管理

Data transmission and security management



源端 (OLTP)




目标端 (异地同步)



目标端 (数仓/搜索)



| 典型产品     | 特点                            |
|----------|-------------------------------|
| DTS(阿里云) | 云服务, 与阿里云集成紧密                 |
| NineData | 混合SaaS服务, 无需安装, 功能强大, 支持多个云平台 |
| Canal    | 开源软件                          |

做好备份、管好密码、加密传输、补丁升级（非云数据库）

## 软硬件故障

### 硬件故障

服务器、硬盘

### 机房灾难

火灾、地震

### 软件bug

## 安全策略

数据备份

异地容灾

数据加密

数据脱敏

流程与规范

操作审计

## 人为故障

黑客入侵（数据泄露、勒索）  
SQL注入、密码泄露、系统漏洞

内部数据泄露、删库跑路

数据误操作



# 谢谢!

[www.ninedata.cloud](http://www.ninedata.cloud)

让每个人轻松用好数据和云



SQL开发



数据备份



数据复制



数据对比

# THANKS

SQL Server  
vertica  
D B 2  
G B a s e  
O r a c l e  
达梦数据库  
神舟通用  
KingbaseES

2010

2014

2018

openGauss  
OceanBase  
ArkDB  
RASESQL  
HotDB  
StellarDB  
QianBase xTP  
GoldenDB  
云树Shard  
MatrixDB  
DynamoDB  
SinoDB  
DolphinDB  
FastData  
Galaxybase  
KunDB  
GDB  
GaussDB  
PolarDB  
KunDB  
Spacture  
Sequoiadb  
OushuDB  
ArgoDB  
开务数据库  
GreatDB  
MongoDB  
TDSQL  
TiDB  
Tapdata  
StarRocks  
UbiSQL