大数据训练营 — 模块七
数据仓库、ETL 和数据开发体系：DW、ETL、Data Platform

极客时间

金澜涛

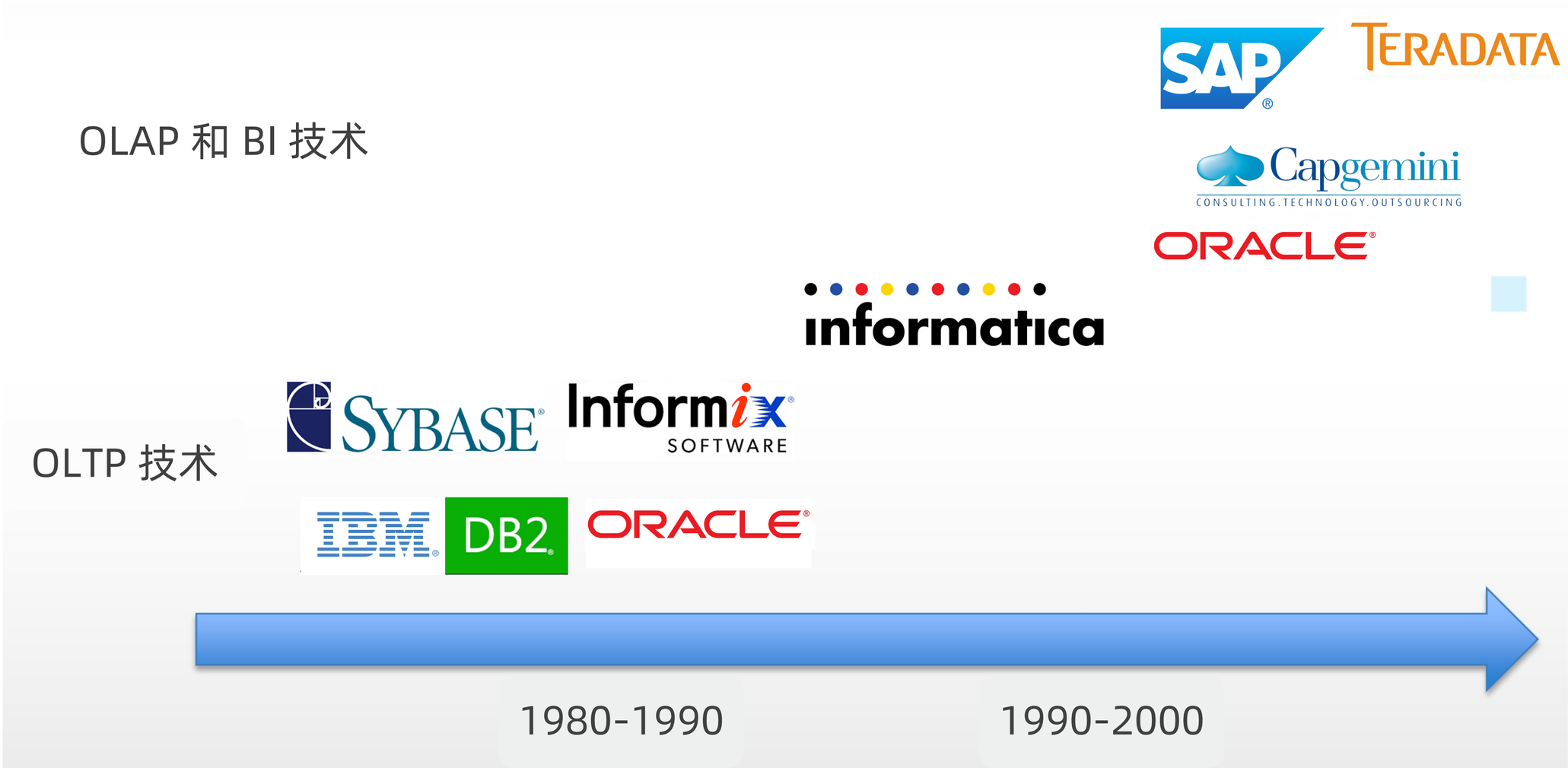
目录

数据仓库体系：

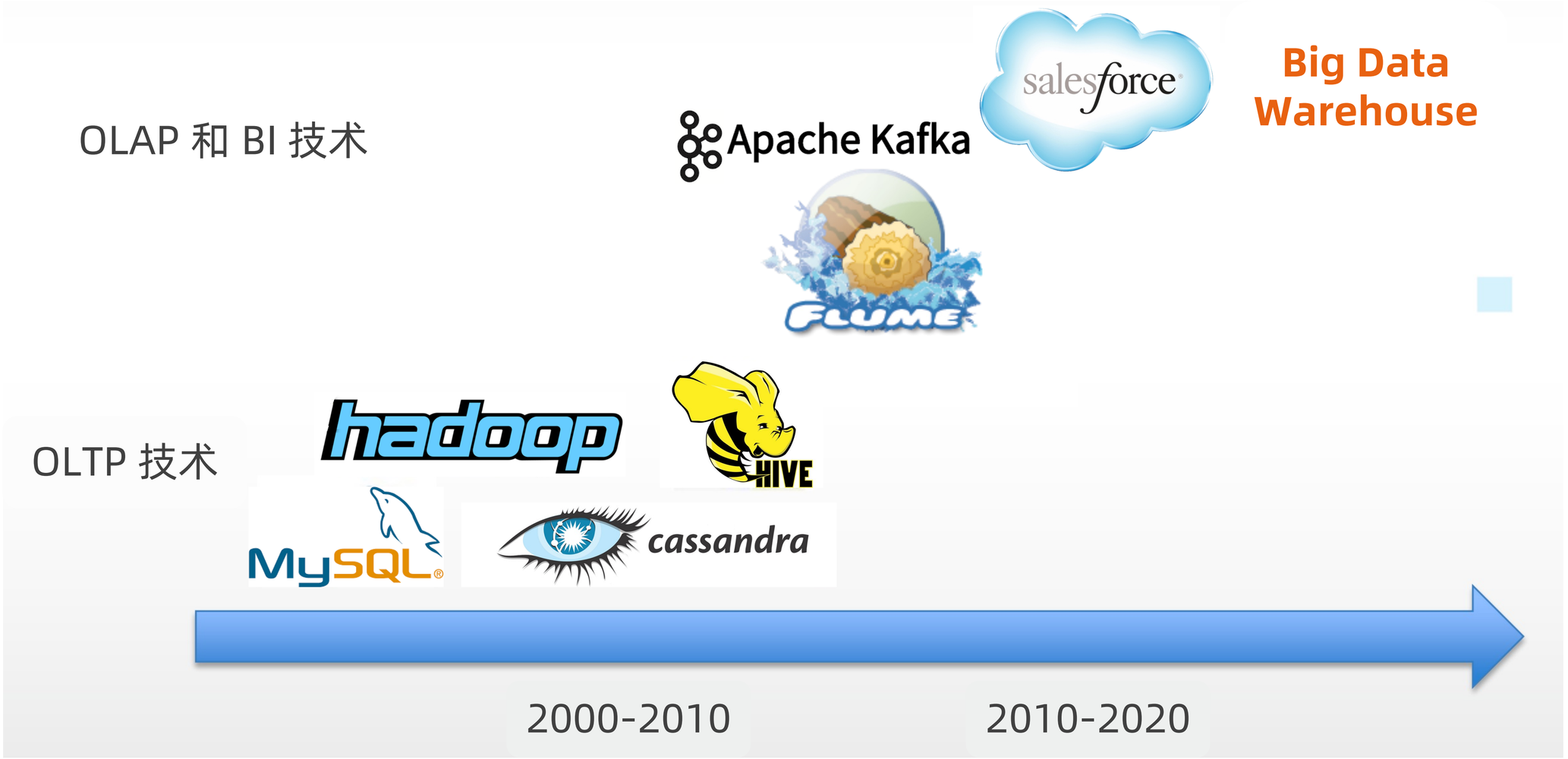
1. DW 的发展历程和现状
2. DW 分层设计
3. Kimball 维度建模技术基础
4. 数仓实践

1. DW 的发展历程和现状

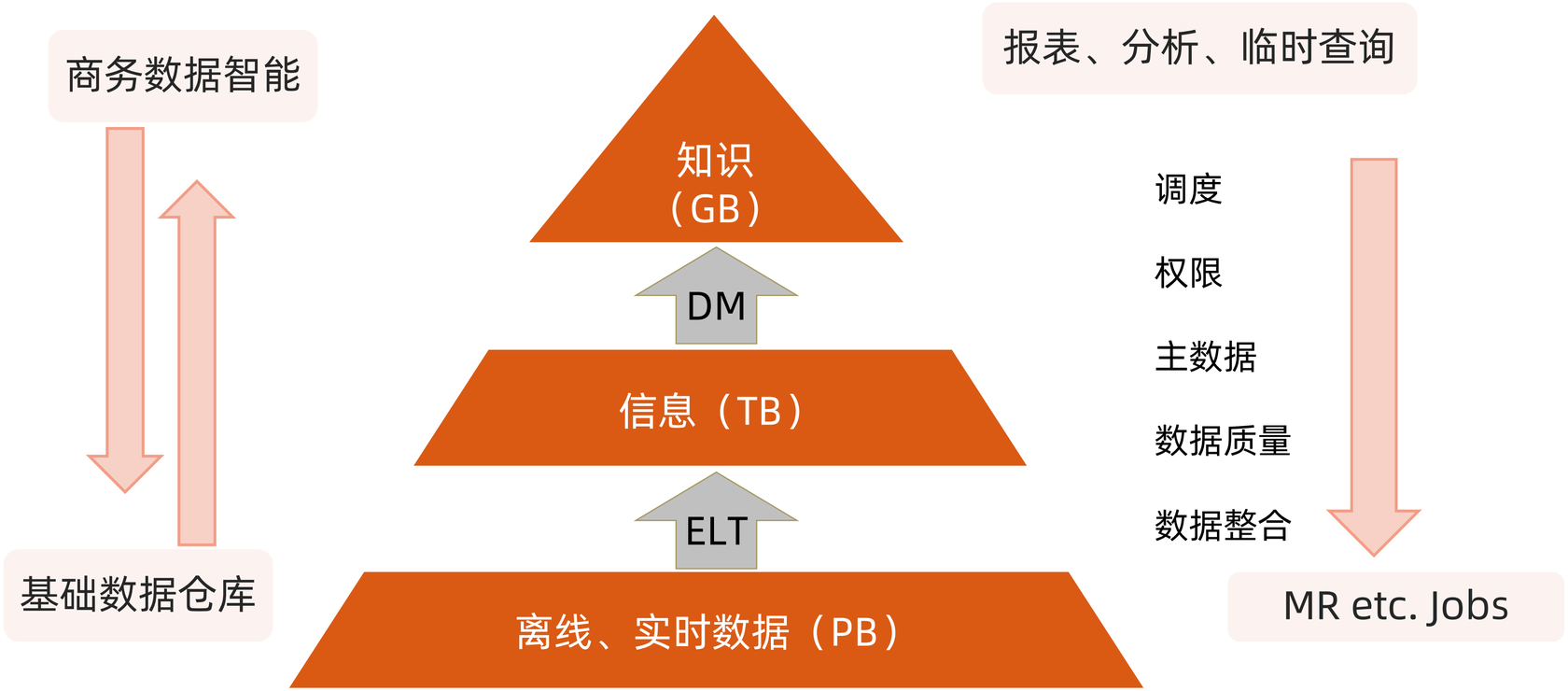
数据技术的历史


OLAP 和 BI 技术
SAP
TERADATA
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING
ORACLE
informatica
OLTP 技术
SYBASE
Informix
SOFTWARE
IBM
DB2
ORACLE
1980-1990
1990-2000

大数据技术影响数据仓库

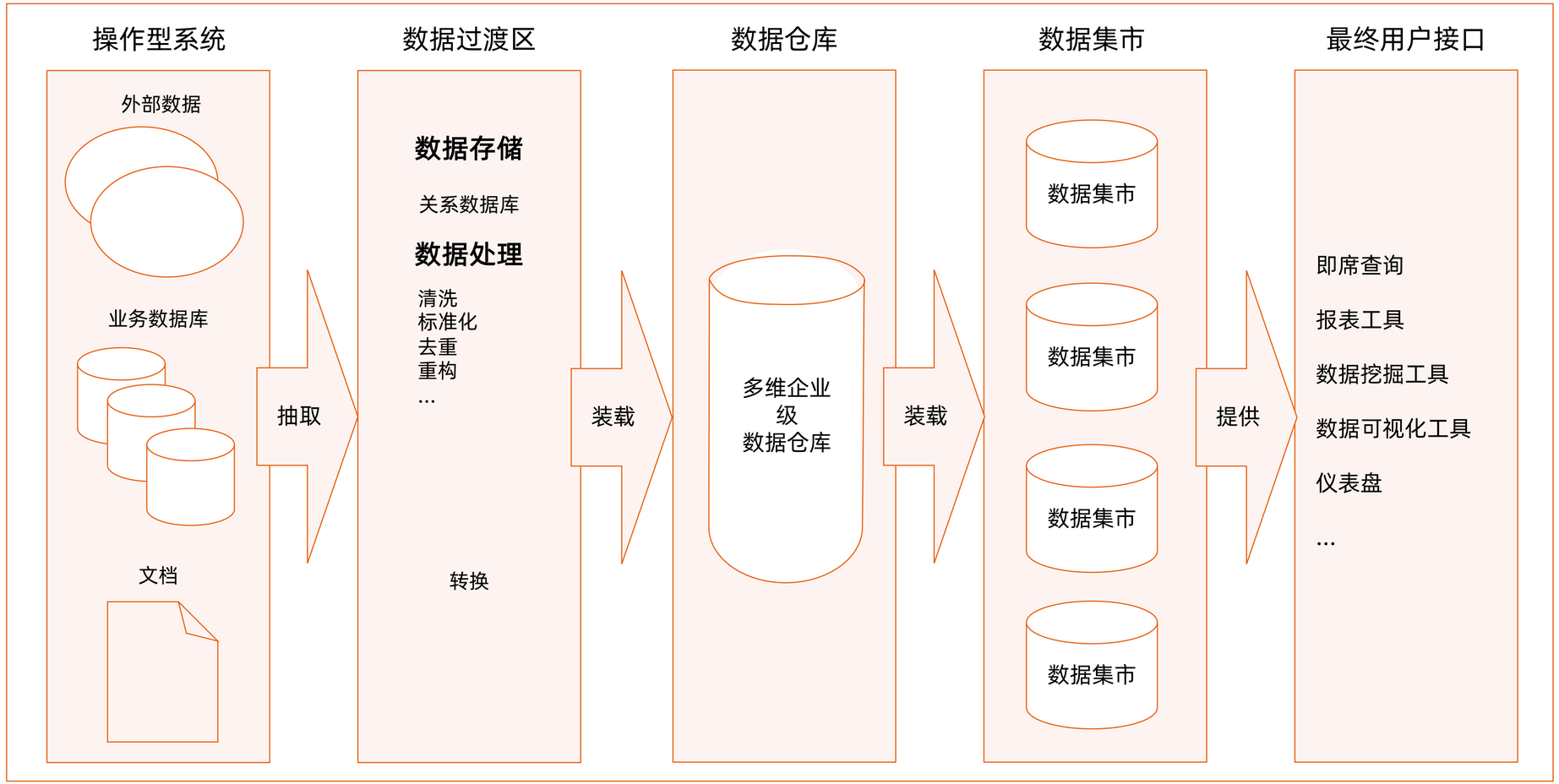


企业级数据仓库


商务数据智能
基础数据仓库
知识
（GB）
DM
信息（TB）
ELT
离线、实时数据（PB）
报表、分析、临时查询
调度
权限
主数据
数据质量
数据整合
MR etc. Jobs

2. DW 分层设计

数据仓库分层架构


操作型系统
数据过渡区
数据仓库
数据集市
最终用户接口
外部数据
业务数据库
文档
抽取
数据存储
关系数据库
数据处理
清洗
标准化
去重
重构
...
转换
装载
多维企业级数据仓库
装载
数据集市
数据集市
数据集市
数据集市
提供
即席查询
报表工具
数据挖掘工具
数据可视化工具
仪表盘
...

为什么要分层

我们对数据进行分层的一个主要原因就是希望在管理数据的时候，能对数据有一个更加清晰的掌控，详细来讲，主要有下面几个原因：

- 清晰数据结构：

  每一个数据分层都有它的作用域，这样我们在使用表的时候能更方便地定位和理解。

- 数据血缘追踪：

  简单来讲一张业务表的来源有很多，如果有一张来源表出问题了，我们希望能够快速准确地定位到问题，并清楚它的危害范围。

- 减少重复开发：

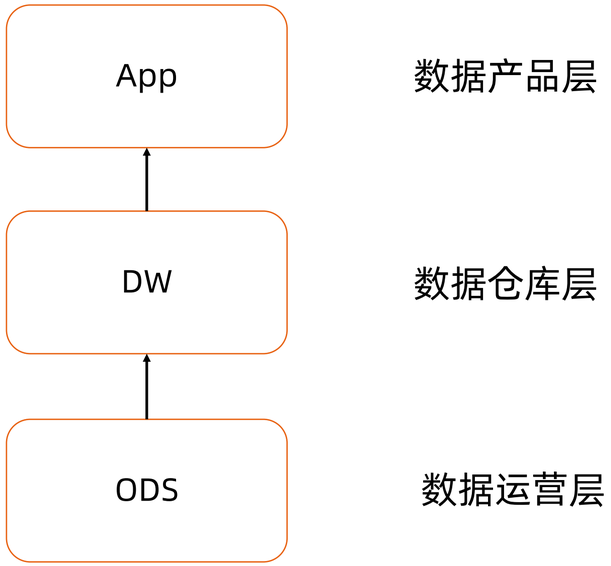
  规范数据分层，开发一些通用的中间层数据，能够减少极大的重复计算。将一个复杂的任务分解成多个步骤来完成，每一层只处理单一的步骤，比较简单和容易理解。而且便于维护数据的准确性，当数据出现问题之后，可以不用修复所有的数据，只需要从有问题的步骤开始修复。

- 屏蔽原始数据的异常：

  屏蔽业务的影响，不必改一次业务就需要重新接入数据。

分层设计

- 我们从理论上来做一个抽象，可以把数据仓库分为下面三个层，即：数据运营层、数据仓库层和数据产品层。



ODS 层

ODS 全称是 Operational Data Store，操作数据存储。

“面向主题的”，数据运营层，也叫 ODS 层，是最接近数据源中数据的一层，数据源中的数据，经过抽取、洗净、传输，也就说传说中的 ETL 之后，装入本层。本层的数据，总体上大多是按照源头业务系统的分类方式而分类的。

ODS 层

例如这一层可能包含的数据表为：

- 人口表（包含每个人的身份证号、姓名、住址等）
- 机场登机记录（包含乘机人身份证号、航班号、乘机日期、起飞城市等）
- 银联的刷卡信息表（包含银行卡号、刷卡地点、刷卡时间、刷卡金额等）
- 银行账户表（包含银行卡号、持卡人身份证号等）等等一系列原始的业务数据

这里我们可以看到，这一层面的数据还具有鲜明的业务数据库的特征，甚至还具有一定的关系数据库中的数据范式的组织形式。

但是，这一层面的数据却不等同于原始数据。在源数据装入这一层时，要进行多项工作，诸如：

- 去噪（例如去掉明显偏离正常水平的银行刷卡信息）
- 去重（例如银行账户信息、公安局人口信息中均含有人的姓名，但是只保留一份即可）
- 提脏（例如有的人的银行卡被盗刷，在十分钟内同时有两笔分别在中国和日本的刷卡信息，这便是脏数据）
- 业务提取
- 单位统一
- 砍字段（例如用于支撑前端系统工作，但是在数据挖掘中不需要的字段）
- 业务判别等

数据仓库层（DW）

- 数据仓库层（DW），是数据仓库的主体。
- 在这里，从 ODS 层中获得的数据按照主题建立各种数据模型。
  - 例如以研究人的旅游消费为主题的数据集中，便可以结合航空公司的登机出行信息，以及银联系统的刷卡记录，进行结合分析，产生数据集。

- 在这里，我们需要了解四个概念：维（dimension）、事实（Fact）、指标（Index）和粒度（Granularity）。

数据仓库层（DW）

极客时间 训练营

- 事实（Fact）数据是一切数据的基础，是针对某一特定事件的度量。
  - 以研究人的旅游消费为主题的数据集为例，这个数据集中的主体的事实数据，便是人在出行到目的地之后，在目的地城市的刷卡消费信息，这一数据由业务发生的事实产生，所以叫做事实数据。
- 维（Dimension）则是事实数据的其中一个侧面。
  - 例如人的旅行的目的地，则便是旅行消费信息的一个维度。
  - 再举一个更加确切的例子，在淘宝网（天猫）中，我们以交易为主题建立一个数据集，那么每一笔交易的信息便构成一个事实表，每一个交易可能有几十个维度，例如商品的类目、卖家、买家、品牌等等。将全部的商品类目提取出来构成一张表，那么这张表便是这个数据集的一个维度表。
  - 在大型数据仓库中，维度表往往非常复杂。例如在淘宝（天猫）中，仅类目一个维度，便包含八十余个一级类目（行业类目）、成千上万个二三四级类目，以及最底层的叶子类目等信息，各个类目之间还存在着树状的关联，有的一级类目便同时是叶子类目（例如手机类目）。
  - 这些维度信息构成了庞大的围绕着事实数据的维度表系统。

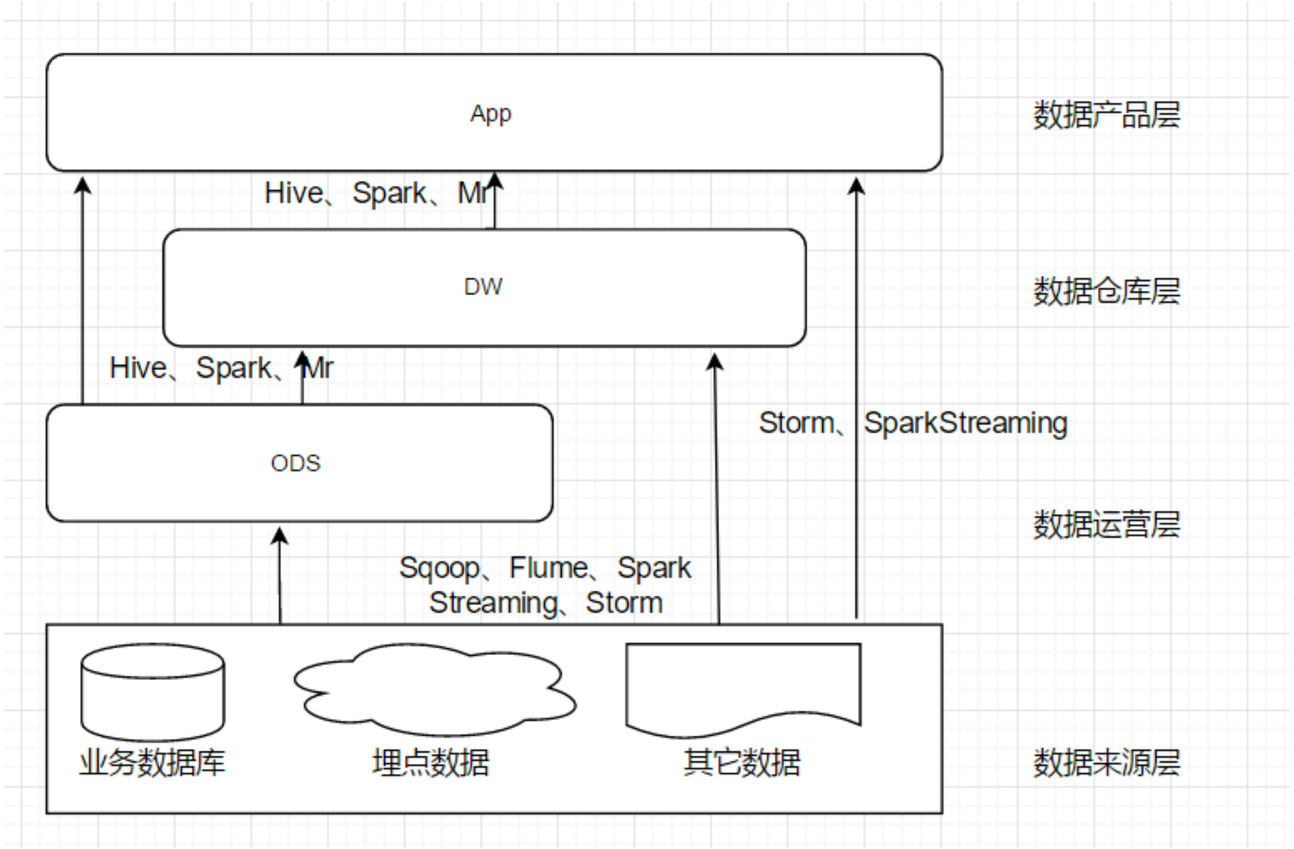
数据仓库层（DW）

- 粒度（Granularity）则表示数据仓库中数据集的精细程度。
  - 粒度越高，则数据的细节越多。例如在高粒度的数据表中，会包含一个人在任何城市中的每一条的刷卡记录，而在低粒度表中，则可能只包含一个人在各个城市中的刷卡总金额。这便是粒度不同的体现。
  - 这么做，是因为大多数的分析挖掘往往仅关注低维度的数据集合，那么使用高维度的数据集进行输入便会大规模的浪费系统资源。
- 指标（Index），则是相对于维度的一个概念，是数据集中不会按照维度进行筛选的数据。
  - 例如刷卡成交金额等。

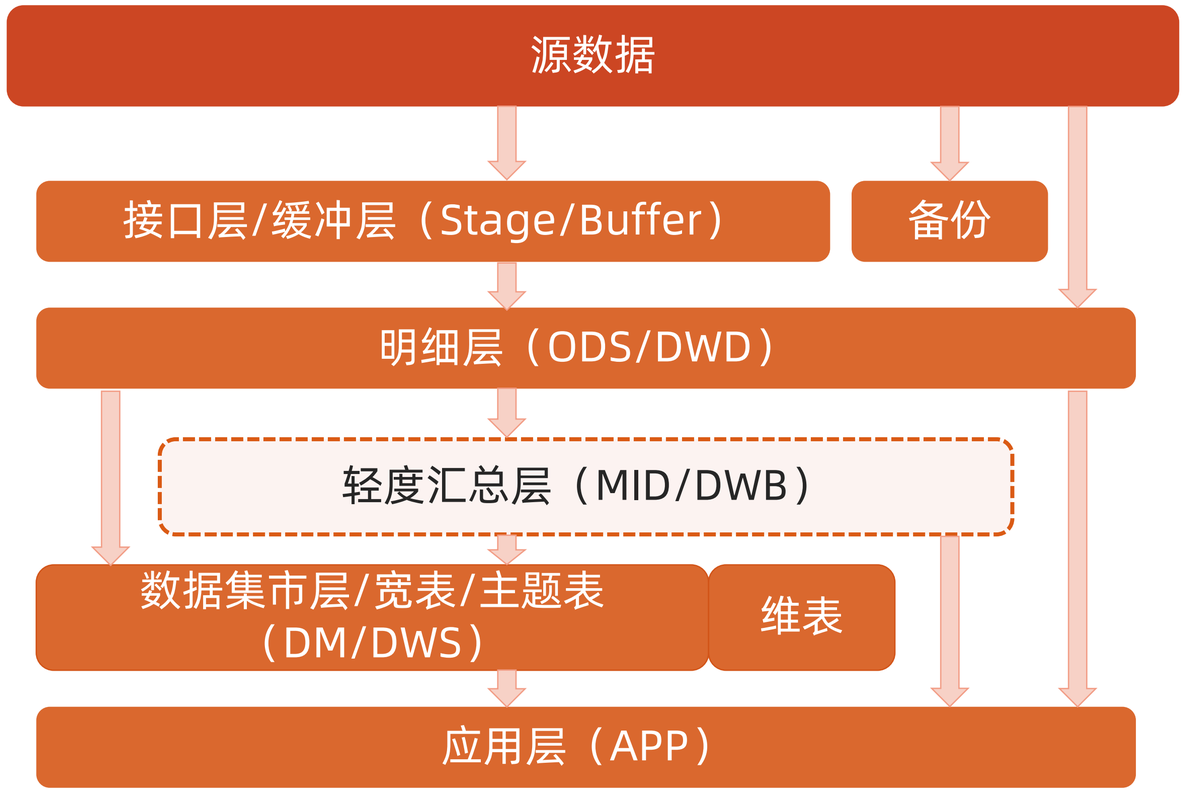
数据产品层（APP）

- 数据产品层（APP），这一层是提供为数据产品使用的结果数据。
  - 在这里，主要是提供给数据产品和数据分析使用的数据，一般会存放在 Elasticsearch、MySQL 等系统中供线上系统使用，也可能会存在 Hive 或者 Druid 中供数据分析和数据挖掘使用。
- 数据生成方式：由明细层、轻度汇总层，数据集市层生成，一般要求数据主要来源于集市层。
- 日志存储方式：使用 Impala/Hive 内表，parquet 文件格式。
- 日志删除方式：长久存储。
- 表 schema：一般按天创建分区，没有时间概念的按具体业务选择分区字段。

数据仓库分层架构和计算引擎



数据分层设计


源数据
接口层/缓冲层（Stage/Buffer）
备份
明细层（ODS/DWD）
轻度汇总层（MID/DWB）
数据集市层/宽表/主题表
（DM/DWS）
维表
应用层（APP）

缓冲层

极客时间训练营

- 缓冲层（buffer）
  - 概念：又称为接口层（stage），用于存储每天的增量数据和变更数据，如 Canal 接收的业务变更日志。
  - 数据生成方式：直接从 Kafka 接收源数据，需要业务表每天生成 update、delete、inseret 数据，只生成 insert 数据的业务表，数据直接入明细层。
  - 表 schema：一般按天创建分区，没有时间概念的按具体业务选择分区字段。

轻度汇总层

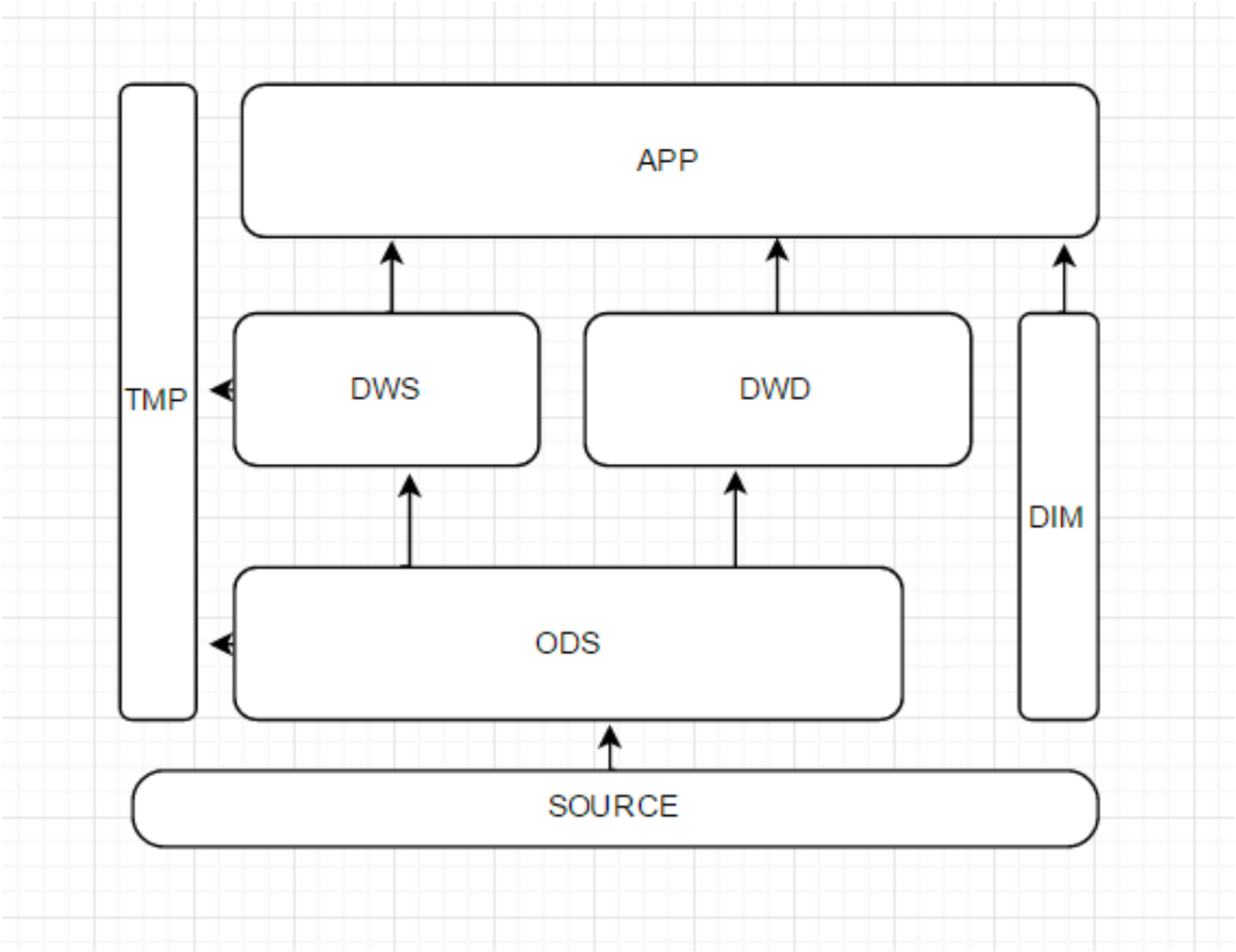
极客时间训练营

- 轻度汇总层（MID 或 DWB，data warehouse basis）
  - 轻度汇总层是数据仓库中 DWD 层和 DM 层之间的一个过渡层次，是对 DWD 层的生产数据进行轻度综合和汇总统计（可以把复杂的清洗、处理包含，如根据 PV 日志生成的会话数据）。轻度综合层与 DWD 的主要区别在于二者的应用领域不同，DWD 的数据来源于生产型系统，轻度综合层则面向分析型应用进行细粒度的统计和沉淀。
  - 数据生成方式：由明细层按照一定的业务需求生成轻度汇总表。明细层需要复杂清洗的数据和需要 MR 处理的数据也经过处理后接入到轻度汇总层。
  - 日志删除方式：长久存储。

主题层

- 主题层（DM，date market 或 DWS，data warehouse service）
  - 又称数据集市或宽表。按照业务划分，如流量、订单、用户等，生成字段比较多的宽表，用于提供后续的业务查询、OLAP 分析、数据分发等。
  - 数据生成方式：由轻度汇总层和明细层数据计算生成。
  - 日志存储方式：使用 Impala 内表，parquet 文件格式。
  - 日志删除方式：长久存储。
  - 表 schema：一般按天创建分区，没有时间概念的按具体业务选择分区字段。

分层总结



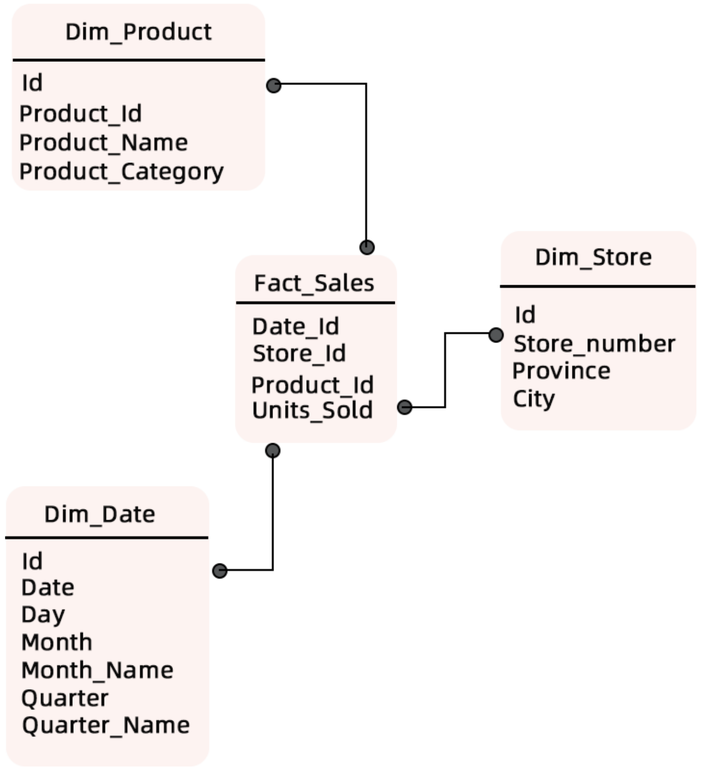
3. Kimball 维度建模技术基础

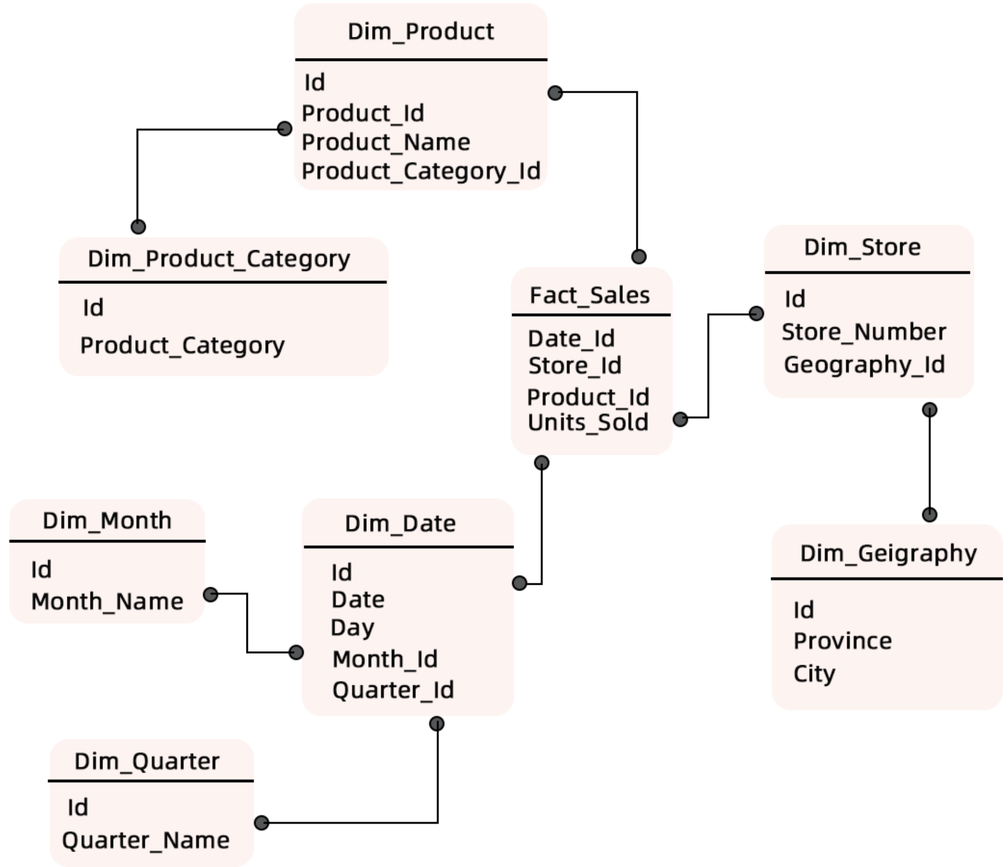
维度建模

极客时间 训练营

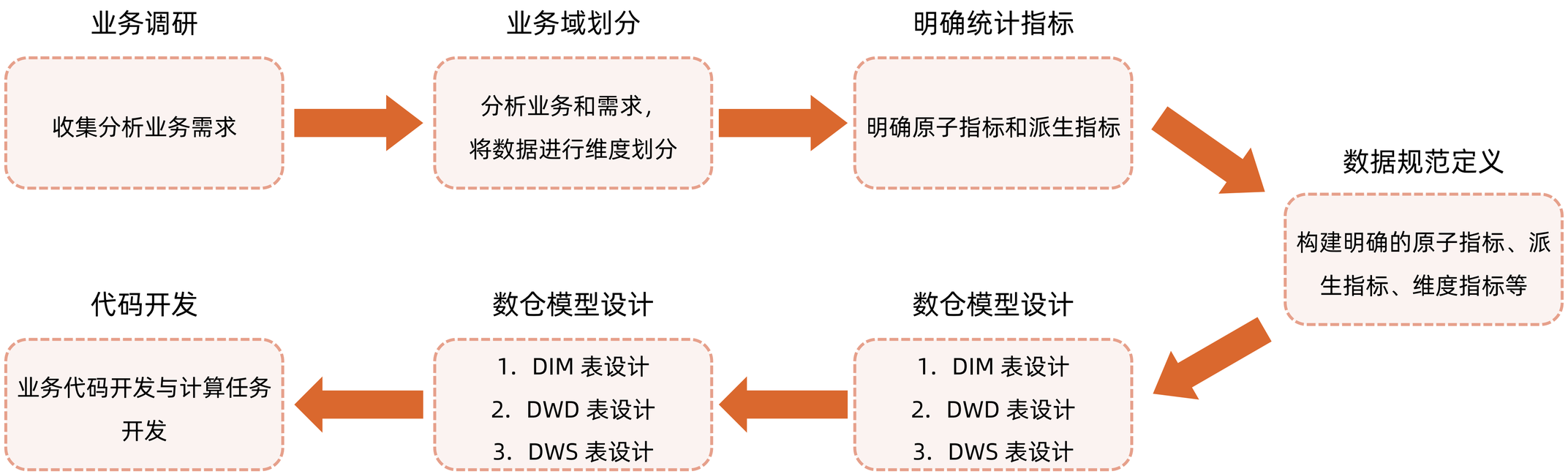
- 维度建模（dimensional modeling）是数据仓库建设中的一种数据建模方法，专门用于分析型数据库、数据仓库、数据集市建模。
- 维度表
  - 维度是对数据进行分析时所用的一个量，比如分析产品销售情况，可以选择按类别来进行分析，或按区域来分析。这样的按什么分析就构成一个维度。
- 事实表
  - 事实表包含业务具体数据，通常包含大量的行。事实数据表的主要特点是包含数字数据（事实），并且这些数字信息可以汇总，以提供有关单位作为历史的数据。事实表包含了与各维度表相关联的外键，并通过 JOIN 方式与维度表关联。

维度建模方式


Dim_Product
Id
Product_Id
Product_Name
Product_Category
Fact_Sales
Date_Id
Store_Id
Product_Id
Units_Sold
Dim_Store
Id
Store_number
Province
City
Dim_Date
Id
Date
Day
Month
Month_Name
Quarter
Quarter_Name

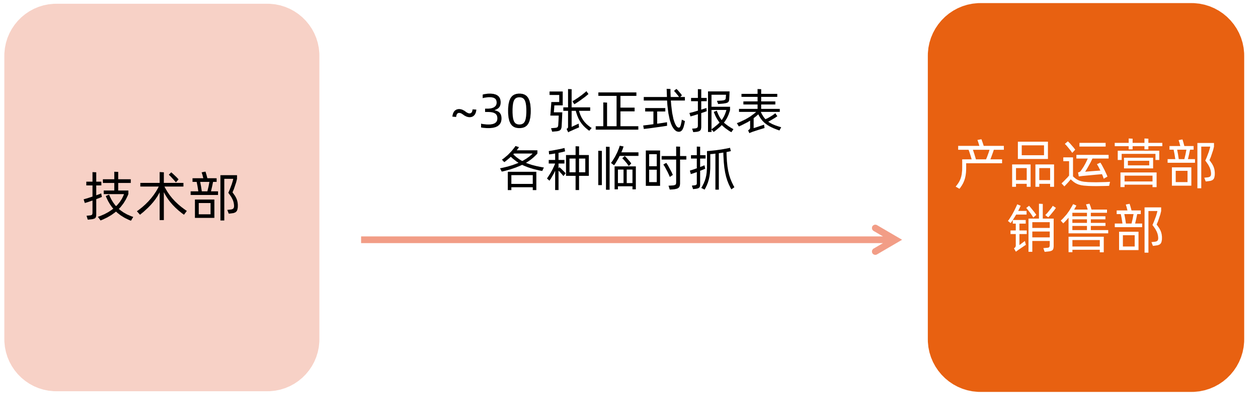

Dim_Product
Id
Product_Id
Product_Name
Product_Category_Id
Dim_Product_Category
Id
Product_Category
Fact_Sales
Date_Id
Store_Id
Product_Id
Units_Sold
Dim_Store
Id
Store_Number
Geography_Id
Dim_Month
Id
Month_Name
Dim_Date
Id
Date
Day
Month_Id
Quarter_Id
Dim_Geigraphy
Id
Province
City
Dim_Quarter
Id
Quarter_Name

数据建模流程


业务调研
收集分析业务需求
业务域划分
分析业务和需求，
将数据进行维度划分
明确统计指标
明确原子指标和派生指标
数据规范定义
构建明确的原子指标、派生指标、维度指标等
数仓模型设计
1．DIM 表设计
2．DWD 表设计
3．DWS 表设计
数仓模型设计
1．DIM 表设计
2．DWD 表设计
3．DWS 表设计
代码开发
业务代码开发与计算任务开发

4. 数仓实践（大众点评）

初期



2012.07（1.0）

数据：

1. 以支持用户报表需求为主。

2. 初步沉淀出了一些底层模型。

3. 模型计算程序以 Python 为主。

数据应用：

1. 报表数据以邮件的形式发送给用户。

2. 用户可以使用自定义 SQL 的 web 查询工具主动查询数据。

架构：

1. 存储和计算都在 GreenPlum。

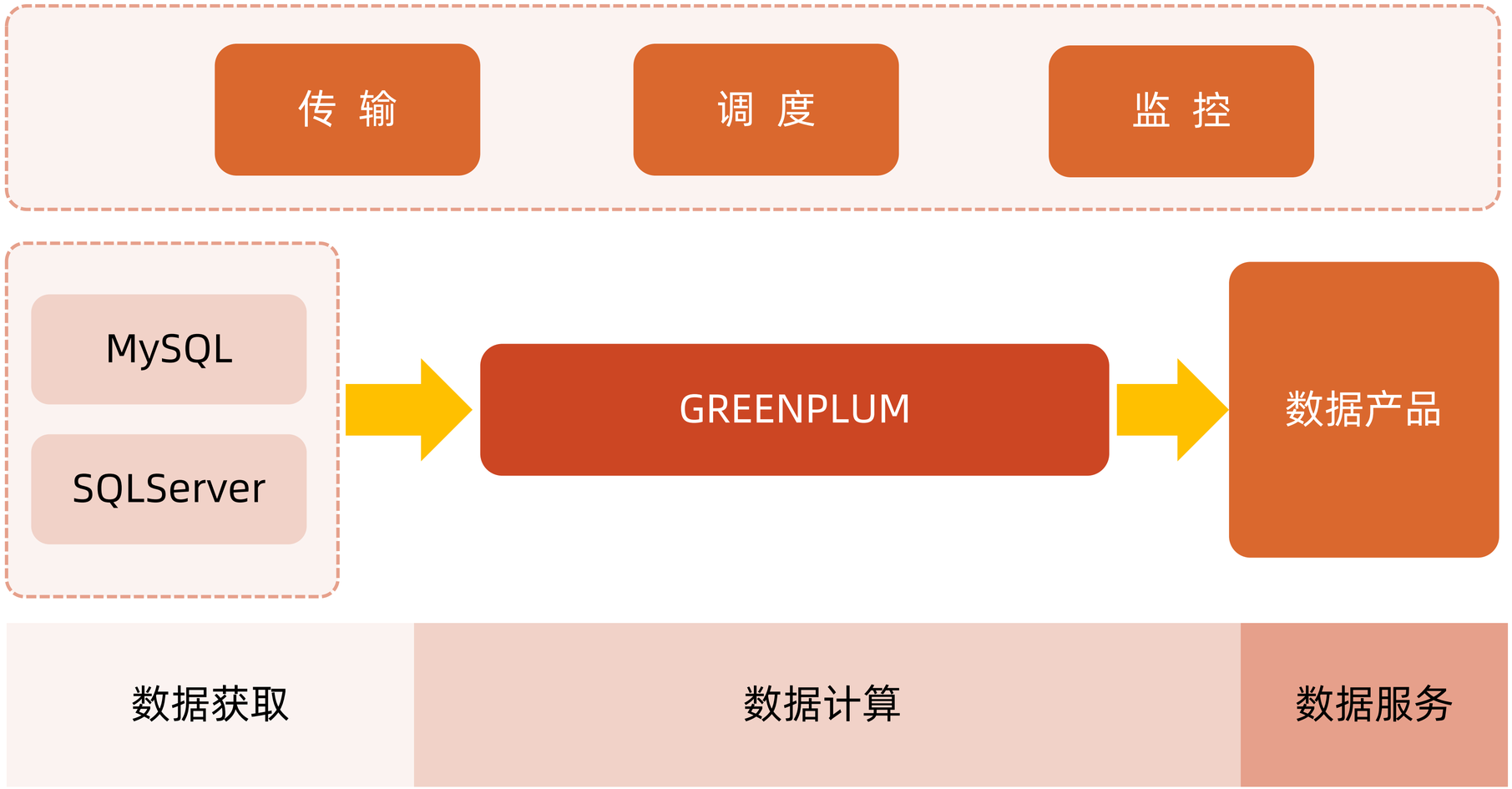
2. GreenPlum 采用双集群热备，部分关键报表数据同时在两个集群存储、计算。

3. 传输：公司的 DBA 同学将数据从 MySQL、SQL Server 拉出来，落地成文件。传输程序每天凌晨解析落地的文件，然后将数据 load 到 Greenplum。

4. 调度：使用 Quartz 框架，依赖关系存放到表中，将依赖检查做成一个脚本，下游 job 调用方法 check 上游任务是否完成。

5. 监控：用户程序自主判断异常，邮件、手机报警。

2012.07（1.0）



2013.04（2.0）

数据：

1. 有了明确的模型分层：

   - ODS：存放从原系统采集来的原始数据。
   - DW：保存经过清洗、转换和重新组织的历史数据，数据将保留较长时间,满足系统最细粒度的查询需要。
   - DM：数据集市，基于部门或某一特定分析主题需要。
   - RPT：直接面向用户的报表。

2. 形成了流量、团购、信息三大基础模型及构建于三大基础模型之上的数据集市。

3. 基于 velocity 开发了计算框架。

4. 开发了一些自定义的 UDF。

2013.04（2.0）

极客时间 训练营

架构：

1. 存储和计算都基于 Hive。

2. Greenplum 作为 Hive 的“cache”存在，供用户做一些小数据的快查询、报表存储。

3. 调度：和 Canaan 框架进行整合，支持用户快速新增任务，并自动导入任务依赖。

4. 主数据：保存了数据仓库元数据信息，供用户查询和系统内部各个模块交互。

5. ACL：构建了数据仓库数据访问权限控制，包括用户权限申请、审批者审批、数据赋权等。

6. 传输：

- 参考阿里 DataX 的设计，实现了点评的异构数据离线传输工具 wormhole。
- 可视化界面，用户通过界面操作，方便地将数据导入导出。
- 和调度、主数据等系统打通。

7. 监控：由于任务数量增长较快（2000+），运维已经是个问题，因此我们花了较大精力做了可视化的工作。

2013.04（2.0）

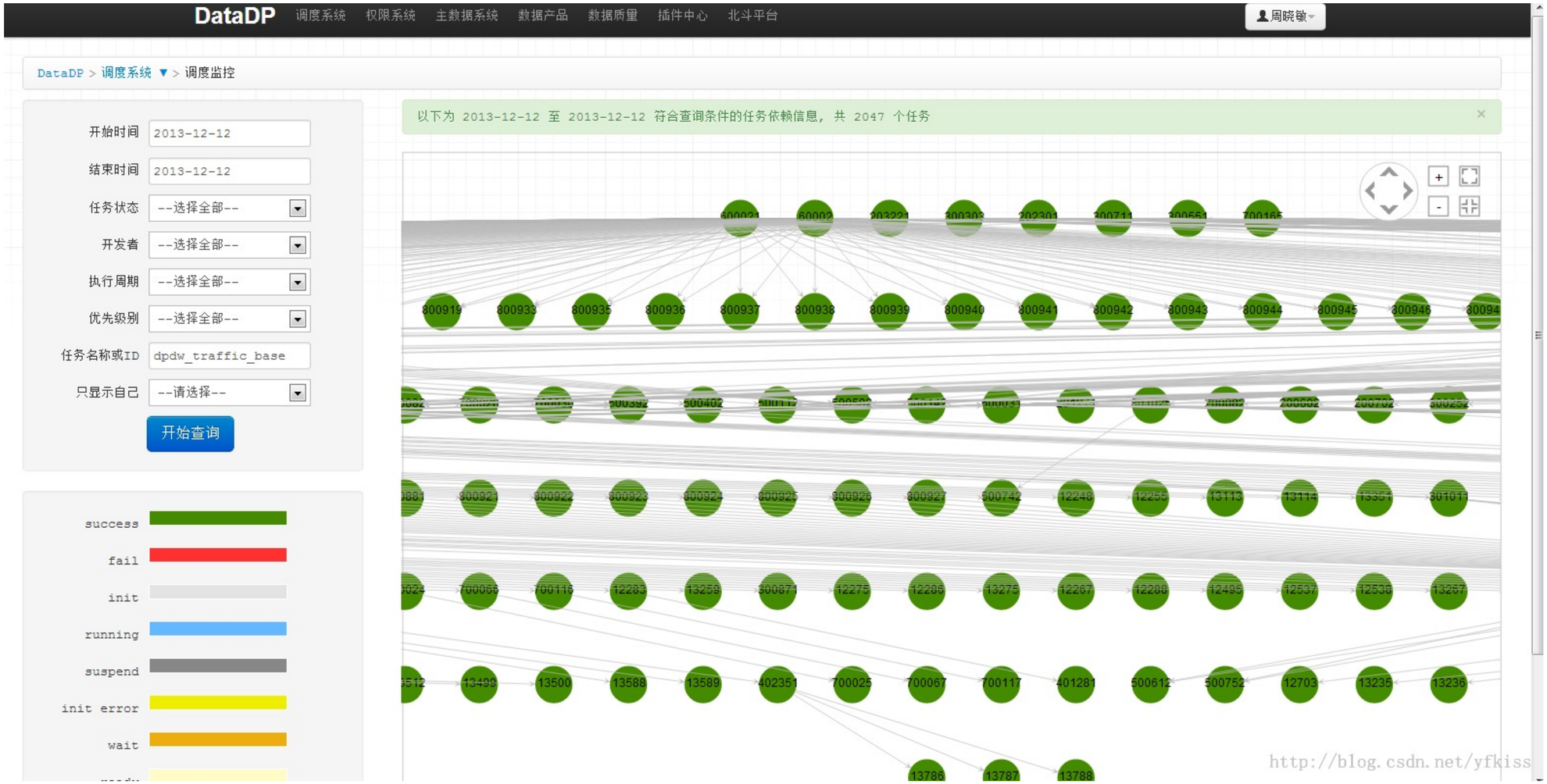
数据应用：

1. 运营工具：用户自定义 SQL，存储基于 Hive。

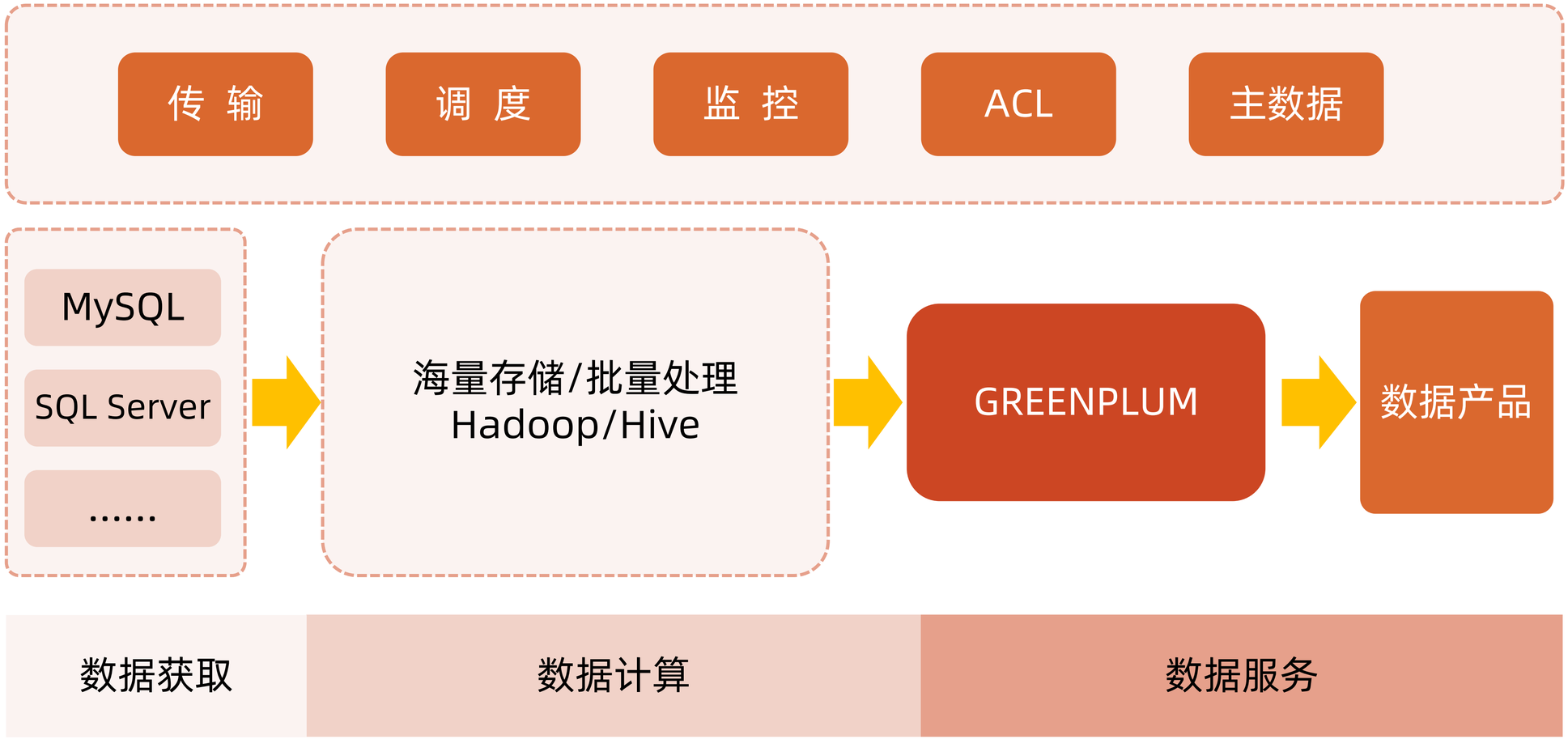
2. 指标（KPI）：用户自定义 SQL，计算基于 Hive，结果放到 Greenplum 中，用户可以根据指标通过时间拼接成报表。

3. Hive Web：非常便捷的 Hive Web 工具，可用性可以甩 Hive 原生的 Web 界面 HWI 几条街了。

2013.04（2.0）


DataDP
调度系统
权限系统
主数据系统
数据产品
数据质量
插件中心
北斗平台
周晓敏
DataDP > 调度系统 ▼ > 调度监控
开始时间
2013-12-12
结束时间
2013-12-12
任务状态
--选择全部--
开发者
--选择全部--
执行周期
--选择全部--
优先级别
--选择全部--
任务名称或ID
dpdw_traffic_base
只显示自己
--请选择--
开始查询
以下为 2013-12-12 至 2013-12-12 符合查询条件的任务依赖信息，共 2047 个任务
success
fail
init
running
suspend
init error
wait
http://blog.csdn.net/yfkiss

2013.04（2.0）


传 输
调 度
监 控
ACL
主数据
MySQL
SQL Server
.....
海量存储/批量处理
Hadoop/Hive
GREENPLUM
数据产品
数据获取
数据计算
数据服务

2013.12（3.0）

极客时间 训练营

数据：

1. 有了明确的上层数据集市，各层数据集市打通，例如团购数据和流量数据打通。
2. 形成了用户集市、商户集市两大主题。
3. 和算法团队合作建设推荐系统。
4. 提供框架和工具支持，引入外部数据开发者。

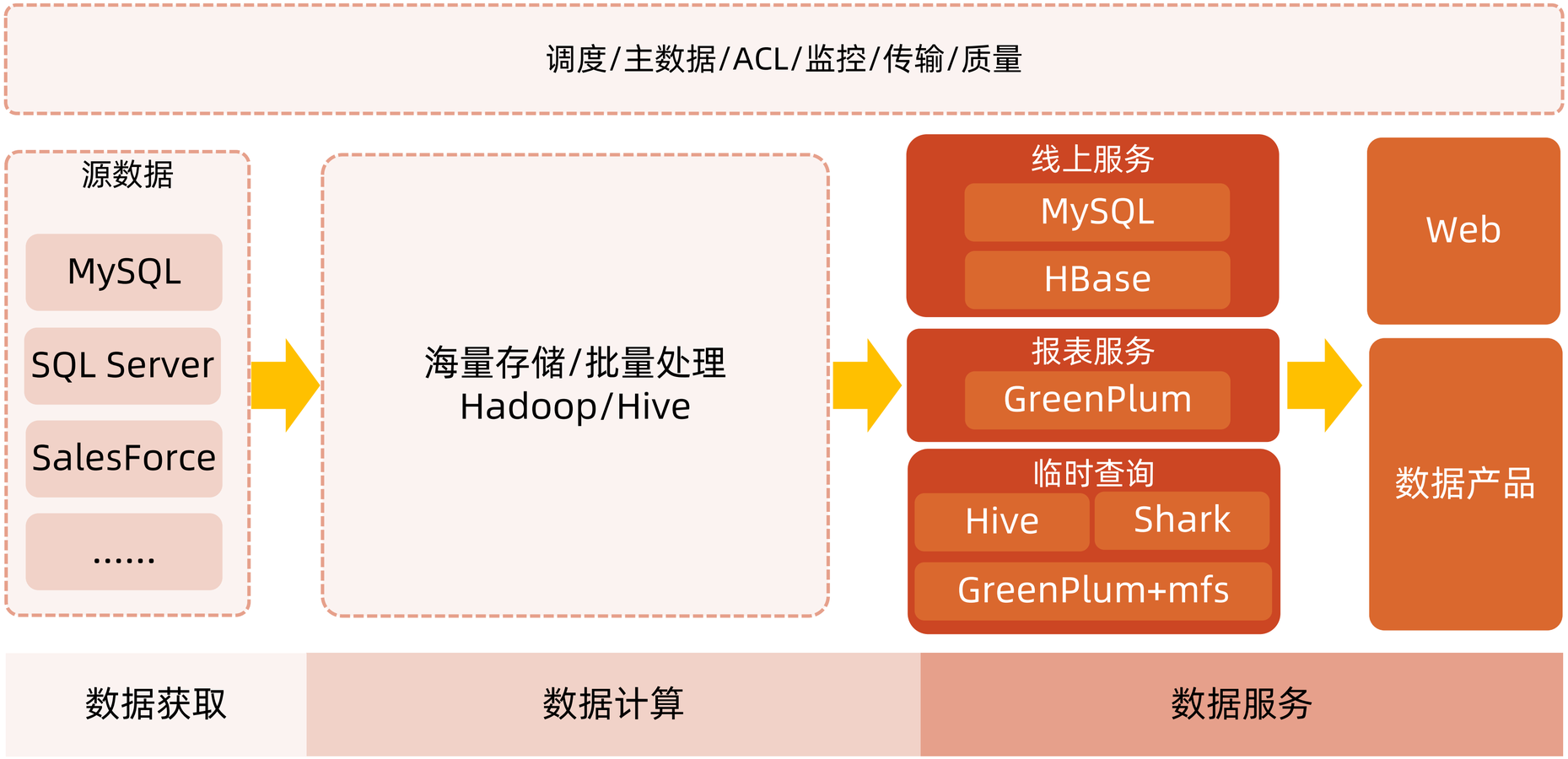
架构：

1. 引入 MySQL、HBase，支持线上服务。
2. 数据访问接口支持：API、Query Engine、RPC Service。
3. 引入 Shark 支持临时查询，出于稳定性考虑，牺牲性能，Shark/Spark 集群和 Hadoop/Hive 集群物理隔离。
4. 数据质量：用户指定以条件，对计算结果做检查。

数据产品：

支持 DashBoard。

2013.12（3.0）


调度/主数据/ACL/监控/传输/质量
源数据
MySQL
SQL Server
SalesForce
.....
海量存储/批量处理
Hadoop/Hive
线上服务
MySQL
HBase
报表服务
GreenPlum
临时查询
Hive
Shark
GreenPlum+mfs
Web
数据产品
数据获取
数据计算
数据服务

2014.12（4.0）

数据：

1. 持续扩充/完善数据模型。
2. 数据规范化，主要包括：APP 日志、渠道。
3. 完善数据开发平台，其他部门数据开发者 100+。

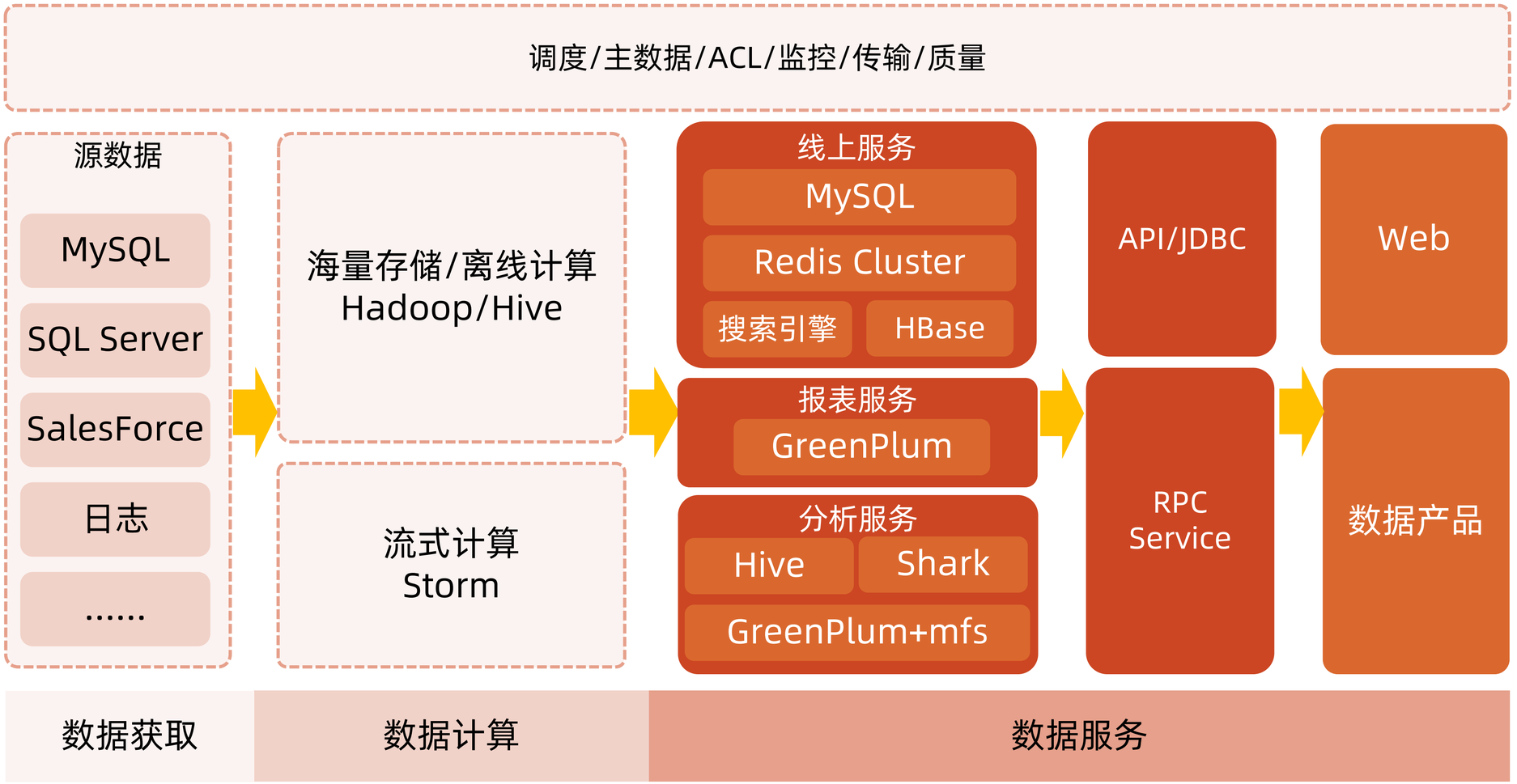
架构：

1. 建设 Redis Cluster，支持实时推荐、用户画像等服务。
2. Hadoop 升级到 YARN。
3. 引入 Storm 支持实时计算。
4. 推出类 Kafka 的分布式消息系统，结合日志框架，支持日志数据的快速 / 低成本接入。
5. 建设元数据中心。

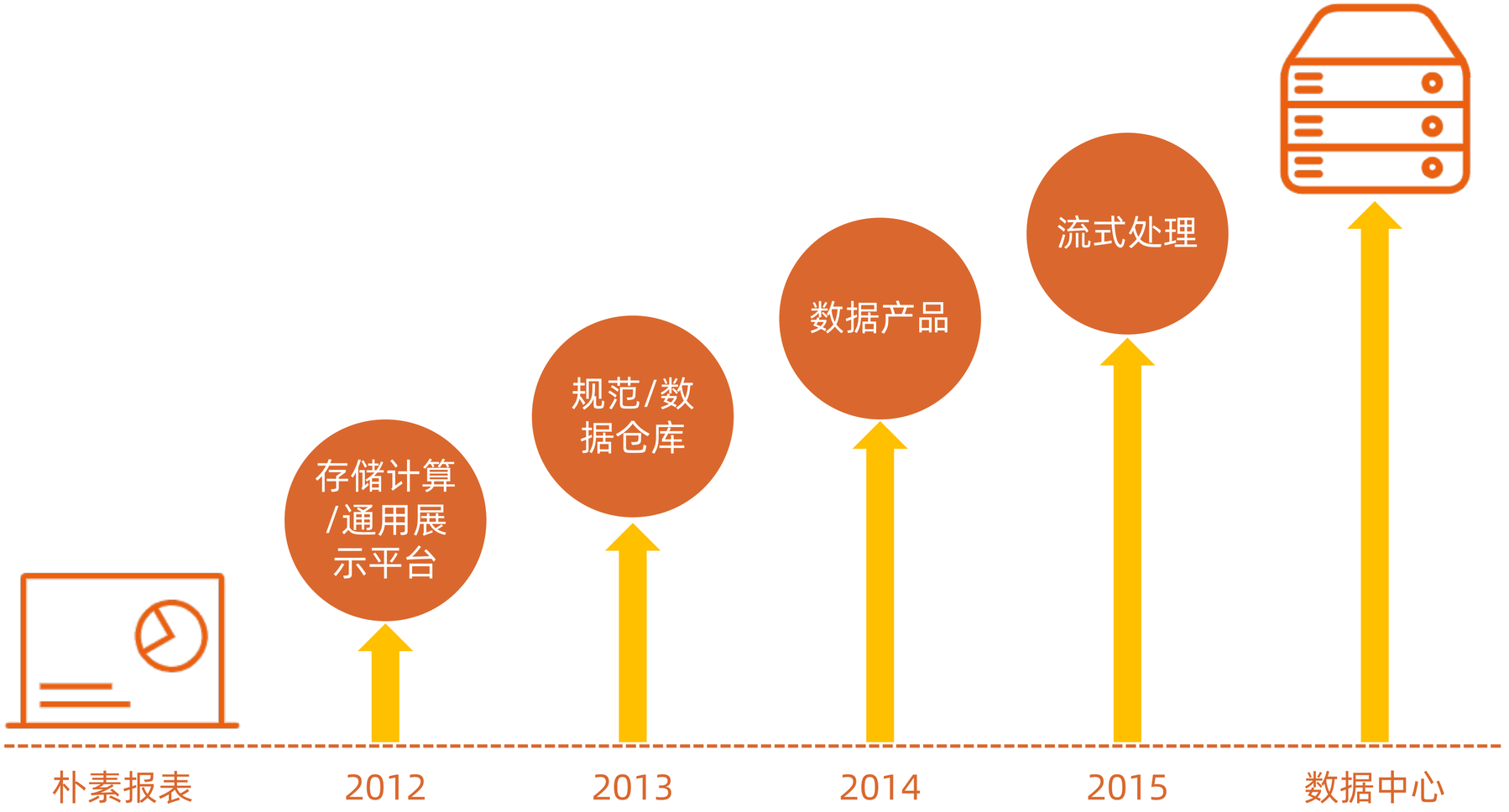
数据产品：

推出专有数据产品，包括：运营效果评估、流量分析产品等。

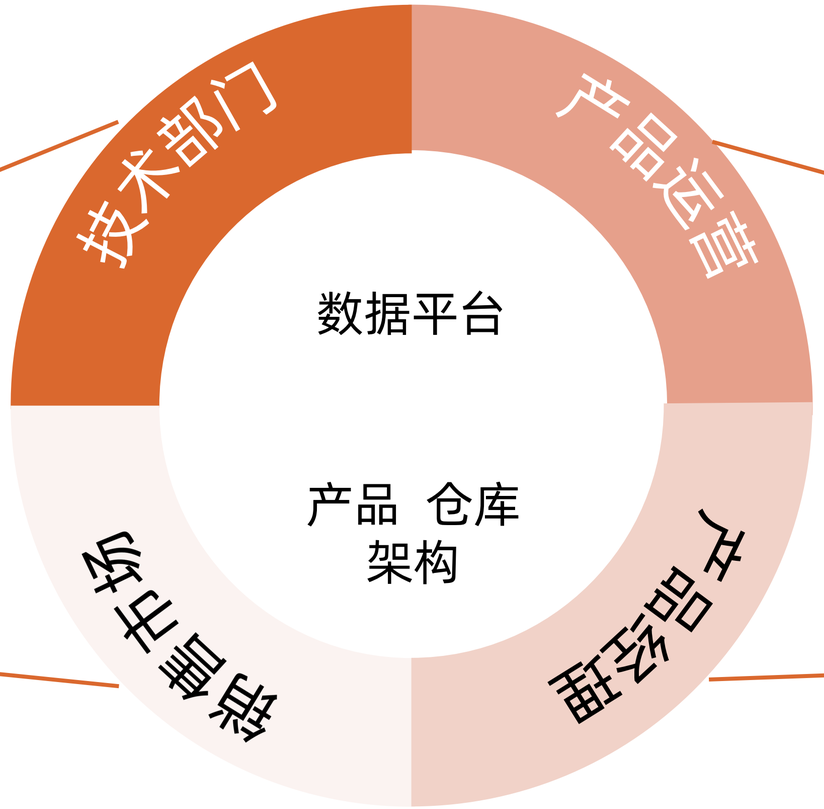
2014.12（4.0）


调度/主数据/ACL/监控/传输/质量
源数据
MySQL
SQL Server
SalesForce
日志
.....
海量存储/离线计算
Hadoop/Hive
流式计算
Storm
线上服务
MySQL
Redis Cluster
搜索引擎
HBase
报表服务
GreenPlum
分析服务
Hive
Shark
GreenPlum+mfs
API/JDBC
RPC
Service
Web
数据产品
数据获取
数据计算
数据服务

发展



数据用户群



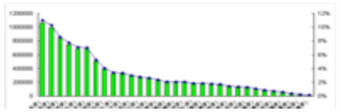
线上服务 HA
编程接口
开发平台

线上反馈应用

大众点评 dianping.com

数据仓库模型
数据质量
性能/稳定性

报告 / OLAP 类



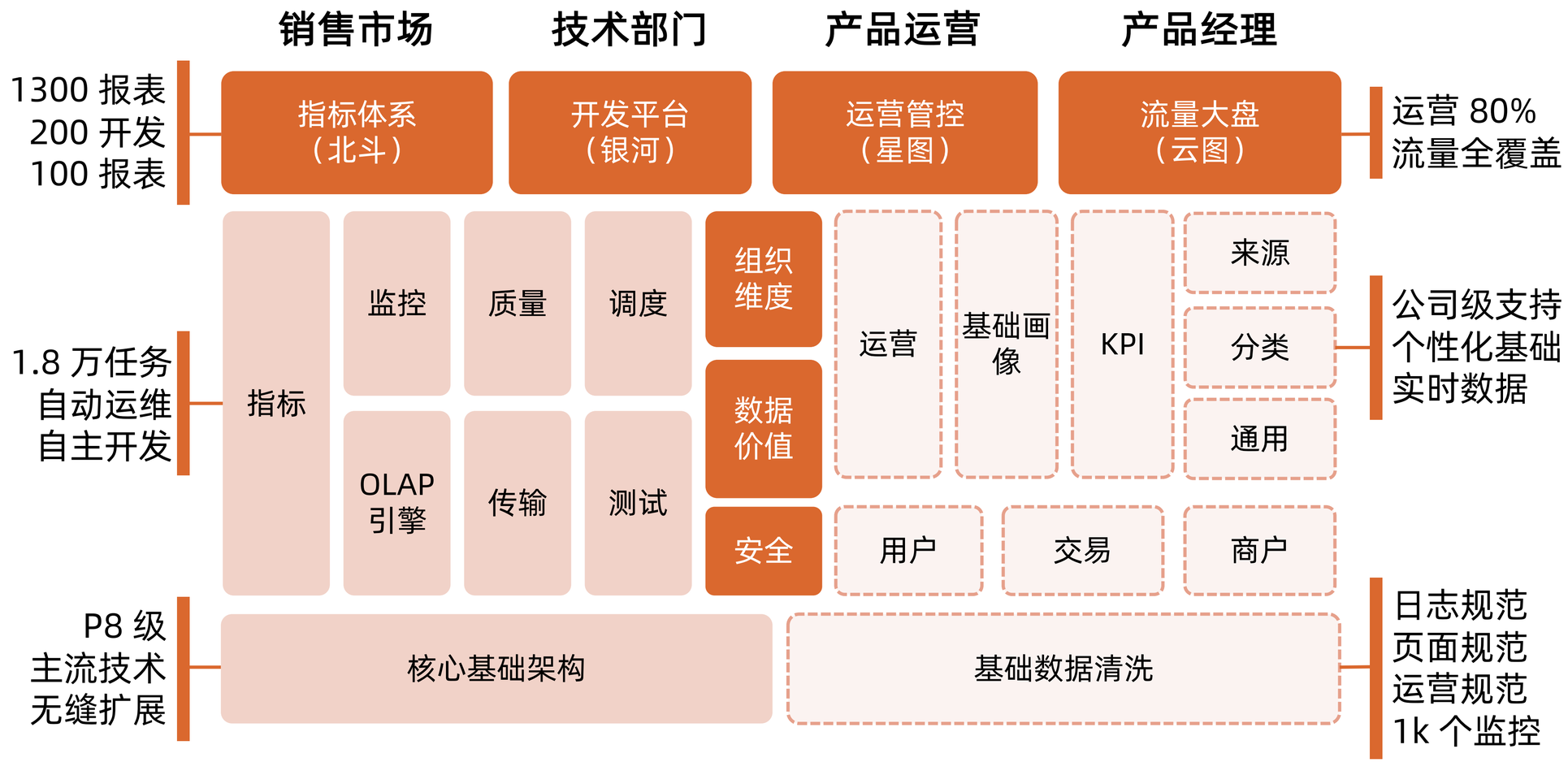
Ad hoc 工具
响应性能优化
指标监控体系

Ad-hoc查询

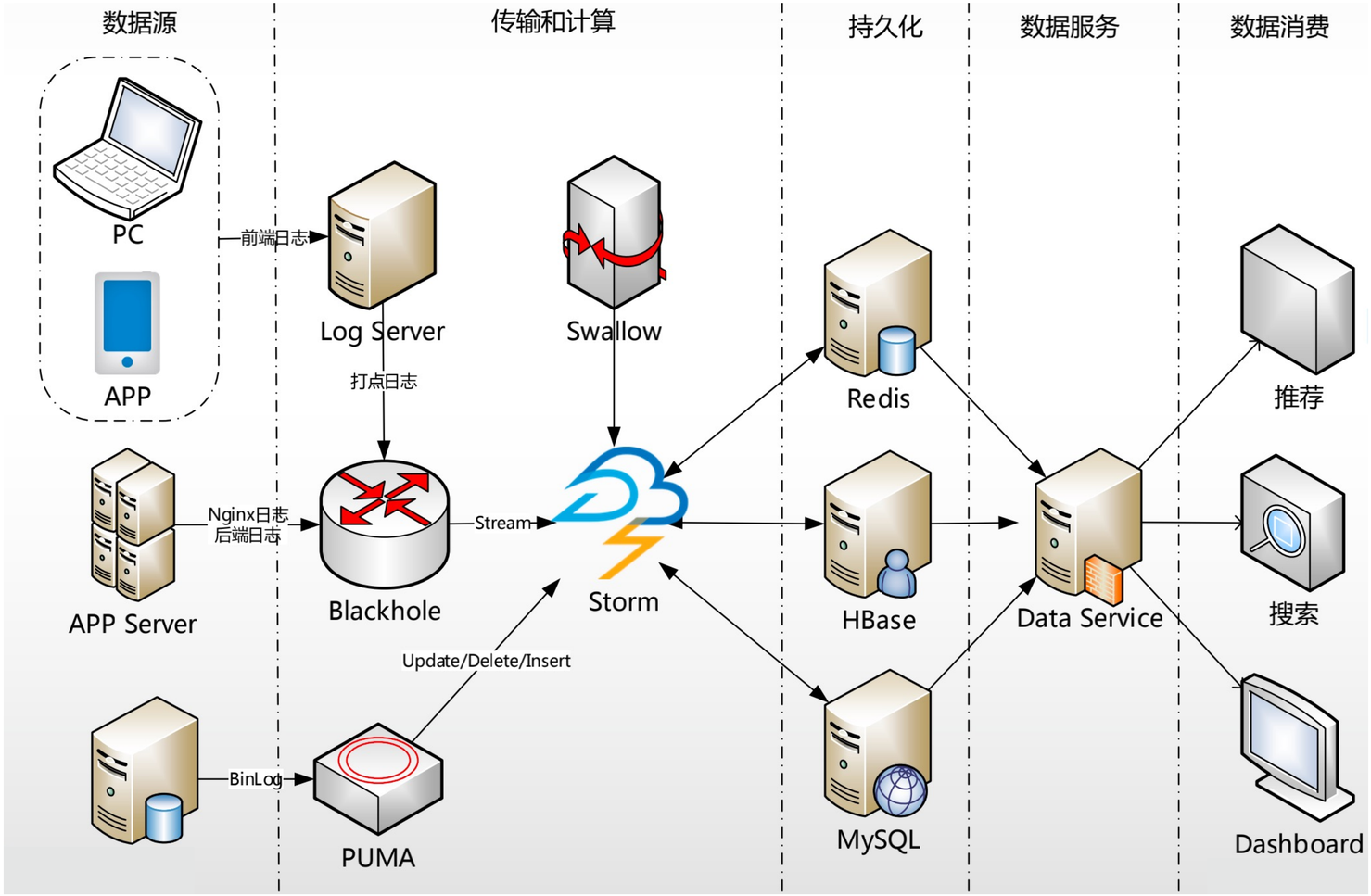
指标数据
自主指标引擎
流式计算

Dashboard 类

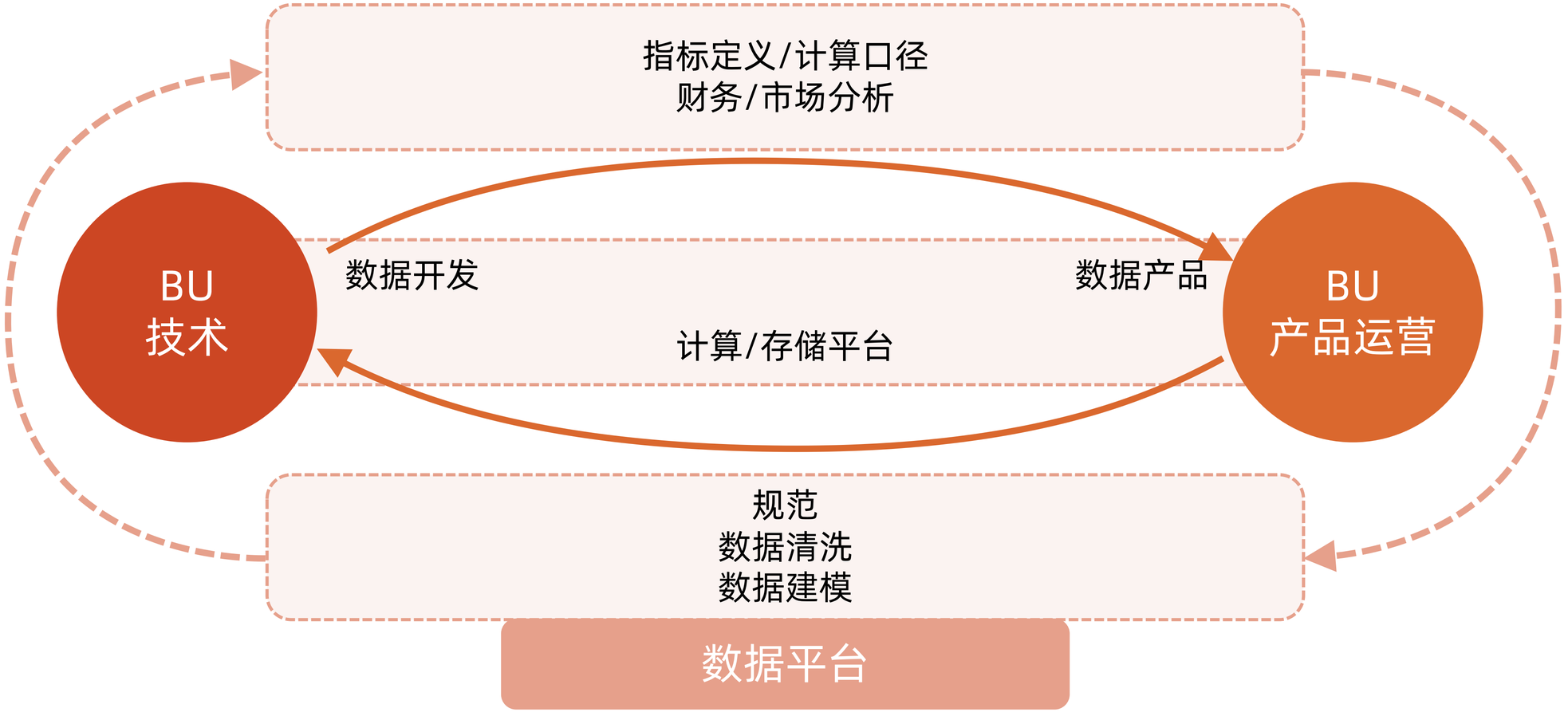
体系架构


销售市场
技术部门
产品运营
产品经理
1300 报表
200 开发
100 报表
指标体系（北斗）
开发平台（银河）
运营管控（星图）
流量大盘（云图）
运营 80%
流量全覆盖
1.8 万任务
自动运维
自主开发
指标
监控
质量
调度
OLAP 引擎
传输
测试
组织维度
数据价值
安全
运营
基础画像
KPI
来源
分类
通用
用户
交易
商户
公司级支持
个性化基础
实时数据
P8 级
主流技术
无缝扩展
核心基础架构
基础数据清洗
日志规范
页面规范
运营规范
1k 个监控

实时计算架构


数据源
传输和计算
持久化
数据服务
数据消费
PC
APP
前端日志
Log Server
打点日志
Swallow
Redis
推荐
Nginx日志
后端日志
APP Server
Blackhole
Stream
Storm
HBase
Data Service
搜索
Update/Delete/Insert
BinLog
PUMA
MySQL
Dashboard

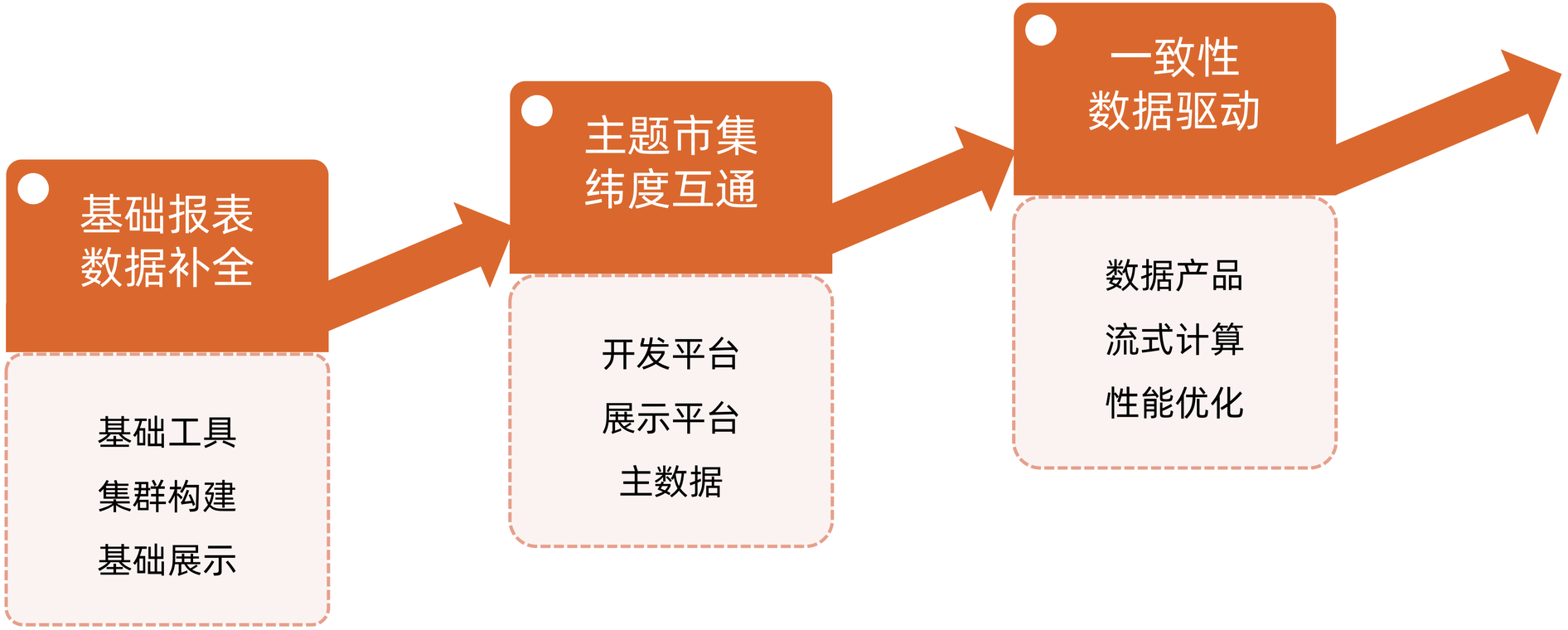
数据产品化之路：业务闭环 VS 数据闭环



数据产品化之路：内部系统 VS 用户体验

- 运营活动平台
  - 团单、资源位、电影、红包
  - 分析和报表（Push 分析）
- 数据部门的产品
  - Frame 方式嵌入（报表平台）
  - Data Service 复用（用户画像）
  - 数据模型的建立（留存分析）
  - 数据源标准的定义（运营效果拆分）

数据中心的未来


基础报表
数据补全
基础工具
集群构建
基础展示
主题市集
纬度互通
开发平台
展示平台
主数据
一致性
数据驱动
数据产品
流式计算
性能优化

THANKS

极客时间 | 训练营