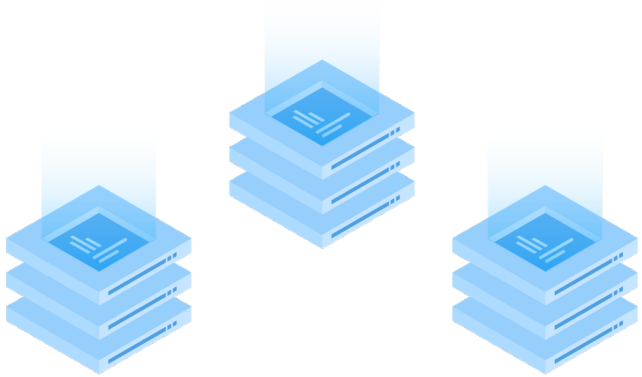
TiCDC在海尔智家的生产实践

姚翔，海尔优家智能科技，技术经理，西安RocketMQ社区联合发起人

PingCAP TiDB



PingCAP

Part I – 新业务下的数据库选型

业务背景

- 2020年4月接到需求需要在5月20日左右在智家APP一级入口上线智慧家板块。
- 智慧家板块围绕智慧生活的信息流产品包括文字，图文，图片集，视频等。涉及录入，展示，搜索，推荐等场景。
- 需要考虑到系统的扩展性包括数据存储。

数据库选型

- 水平线性扩展---存储计算分离的架构；
- 兼容性---mysql 5.7 92%兼容性，事务隔离级别同mysql;
- 摒弃复杂的分库分表方案;
- 分布式事务，乐观和悲观事务的支持，强一致性;
- 原生支持CDC功能；
- 最好能解决现有部分业务数据存储的问题。

作为一个全新的业务，没有历史包袱。考虑到后续的数据量希望能摒弃到传统的分库分表方案，研发能更专注于业务。同时考虑的业务场景需要CDC的场景，mongodb和tidb进入了我们的视野，综合考虑公司现有其他业务系统，其实离不开关系型数据库。我们考虑能不能收敛整个业务系统的数据存储方案，同时具备nosql数据库的扩展性（比如现有的推送系统），且改动不大，支持mysql协议的分布式强一致性关系数据库tidb 4.0最终敲定。

TiDB 4.0不得不说

- TiUP和Dashboard：两个工具极大的简化了部署，运维及监控，由于公司的基础设施构建于阿里云之上，在说服领导TiDB相比于阿里云DRDS，PolarDB及OceanBase的优势外，运维侧同意和落地起到了关键性的作用

- TiFlash：配合 TiDB 体系的列存引擎，可以实时与行存保持同步。这个是其他关系型数据库所不具备，我们APP有大量的数据分析的场景，而TiFlash给了业务团队新的选择，能轻量级实现相关业务，无需通过大数据团队类似lambda等架构去实现。

- TiCDC: 低延迟，高可用的数据变更捕获组件。

PingCAP

Part II – 数据同步TiCDC

TiCDC和Binlog

- 原理
  - TiDB Binlog 是一个用于收集各个 TiDB 实例产生的 binlog，并按照事务提交的时间排序后同步到下游，提供准实时备份和同步功能的商业工具，分为 Pump 和 Drainer 两个组件，以及 binlogctl 工具。
  - TiCDC 是通过 TiKV 的 KV Change Logs 来实现的增量数据同步工具，具有将数据还原到与上游任意 TSO 一致状态的能力，同时提供开放数据协议 (TiCDC Open Protocol)，支持其他系统订阅数据变更，无状态节点。

- 性能
  - 由于TiDB-Binlog 中 Drainer 是单节点归并排序，所以针对大规模的集群不如TiCDC。
  - TiDB-Binlog 在极端情况下可能会丢失 Commit Binlog，需要反查 TiKV 事务状态，同步延迟可达到 10 分钟，正常情况下延迟秒级，而 TiCDC 的同步延迟通常在毫秒级别。

TiCDC和Binlog

- 可用性
  - TiCDC 不依赖 TiDB 事务模型保证数据同步的一致性。
  - TiDB Binlog存在单点瓶颈，TiCDC 各节点无状态，通过 PD 的 etcd 保存元数据信息，系统可水平扩展且天然支持高可用。

海尔智家TiCDC场景

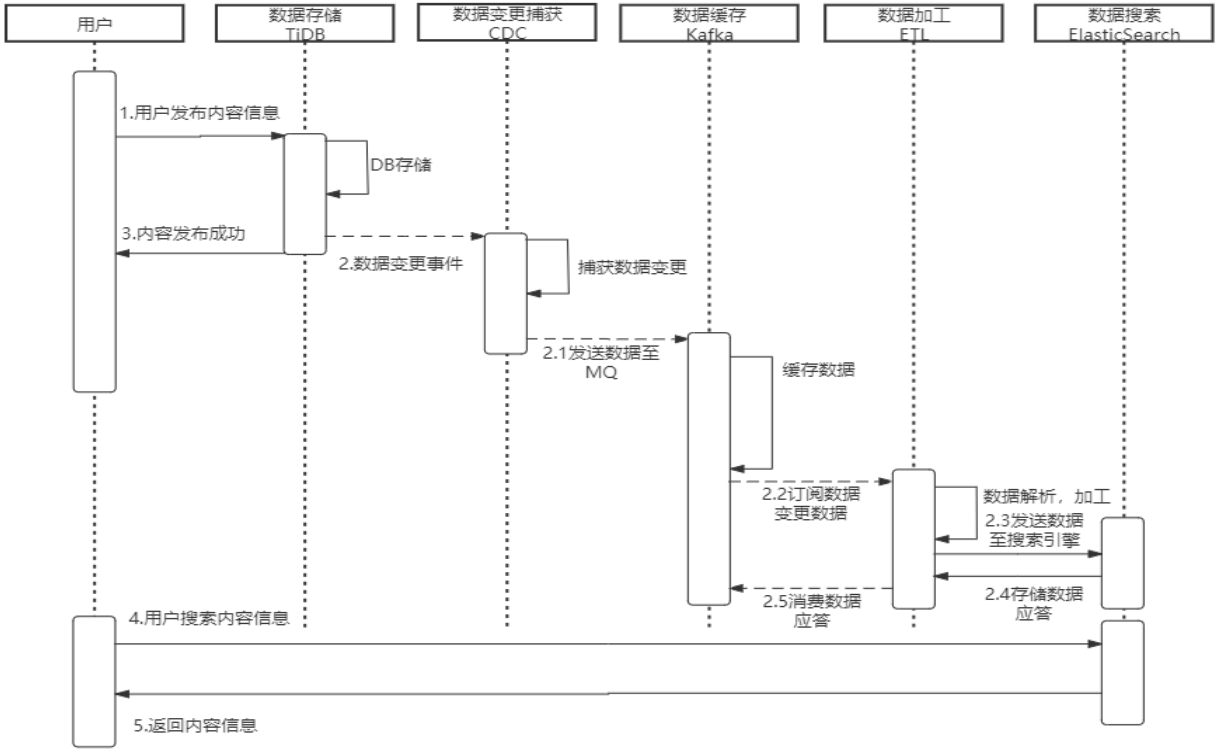
- 场景1：搜索

  同步用户信息和生活家信息到es，提供近实时的搜索功能。目前用户表数据千万，2G+数据量，kafka日消费消息量在300万。(用户搜索APP测入口未开放)。

- 场景2：推荐

  同步生活家信息到大数据，用于智能推荐（在开发）。

海尔智家TiCDC场景



以生活家内容
同步ES为例

TiCDC分发策略

- sink
  - MySQL 协议兼容的数据库，提供最终一致性支持。
  - 以 TiCDC Open Protocol 输出到 Kafka，可实现行级别有序、最终一致性或严格事务一致性三种一致性保证。

    对于 MQ 类的 Sink，支持 default、ts、rowid、table 四种分发器。

    用户表: rowid(主键+唯一索引)

    生活家内容表: default（主键）

TiCDC分发策略

Row Changed Event 划分 Partition 算法				
分发方式	行内有序性	表内有序性	表内事务一致性	partition 分配平衡度
tidb_row_id 分发	[a]			平衡
基于 table 分发				不平衡
基于 ts 分发				平衡
default	有多个唯一索引（包括主键）时按照 table 模式分发； 只有一个唯一索引（或主键）按照 rowid 模式分发； 如果开启了 old value 特性，按照 table 分发			

a:仅当上游表内只有一个主键且主键为 int 类型时，_tidb_row_id 分发满足行内有序性

PingCAP

Part III –问题及实践

问题及生产实践

- 1.kafka:目前 TiCDC 控制了向 Kafka 发送的消息批量的大小最大为 512 MB，其中单个消息的大小最大为 4 MB。需要调整message.max.bytes，replica.fetch.max.bytes，fetch.message.max.bytes到1073741824 (1 GB)，默认值比较小。

- 2.下游MQ持续异常，TiCDC 多次重试后仍然失败,需要手动恢复同步任务。

- 3.TiCDC(4.0.0)挂掉后，出现无法启动TICDC，不管是通过cdc server命令还是整个集群重启，启动后又马上挂掉， TiCDC4.0.5已修复。

- 4.如果同步任务长时间中断（ TiCDC4.0.5之前版本），停止时间超过GC时间，并且还有其他运行中的任务，停止的任务恢复后大概率不能正常访问，因为历史数据可能已经被GC。

- 5.升级TiCDC4.0.0到4.0.5，原有的任务会变为failed状态，线上运行一段时间后CDC节点内存飙升几乎耗尽，多个CDC节点轮流从持续上涨到耗尽到恢复正常，

PingCAP

Thank You !