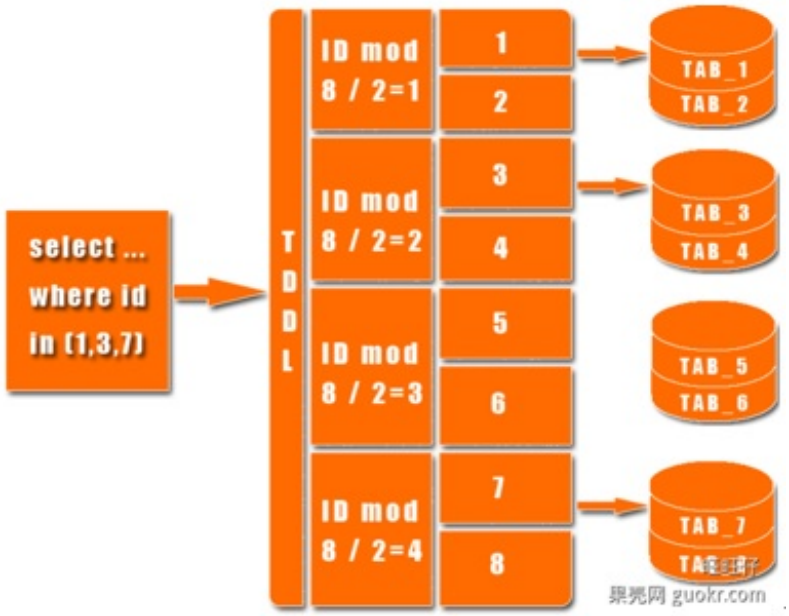
TDDL：來自淘寶的分佈式數據層 | 標點符

淘寶根據自身業務需求研發了TDDL（Taobao Distributed Data Layer）框架，主要用於解決分庫分表場景下的訪問路由（持久層與數據訪問層的配合）以及異構數據庫之間的數據同步，它是一個基於集中式配置的JDBC DataSource實現，具有分庫分表、Master/Salve、動態數據源配置等功能。就目前而言，許多大廠也在出一些更加優秀和社區支持更廣泛的DAL層產品，比如Hibernate Shards、Ibatis-Sharding等。TDDL位於數據庫和持久層之間，它直接與數據庫建立交道，如圖所示：

淘寶很早就對數據進行過分庫的處理，上層系統連接多個數據庫，中間有一個叫做DBRoute的路由來對數據進行統一訪問。DBRoute對數據進行多庫的操作、數據的整合，讓上層系統像操作一個數據庫一樣操作多個庫。但是隨著數據量的增長，對於庫表的分法有了更高的要求，例如，你的商品數據到了百億級別的時候，任何一個庫都無法存放了，於是分成2個、4個、8個、16個、32個……直到1024個、2048個。好，分成這麼多，數據能夠存放了，那怎麼查詢它？這時候，數據查詢的中間件就要能夠承擔這個重任了，它對上層來說，必須像查詢一個數據庫一樣來查詢數據，還要像查詢一個數據庫一樣快（每條查詢在幾毫秒內完成），TDDL就承擔了這樣一個工作。在外面有些系統也用DAL（數據訪問層） 這個概念來命名這個中間件。下圖展示了一個簡單的分庫分表數據查詢策略：

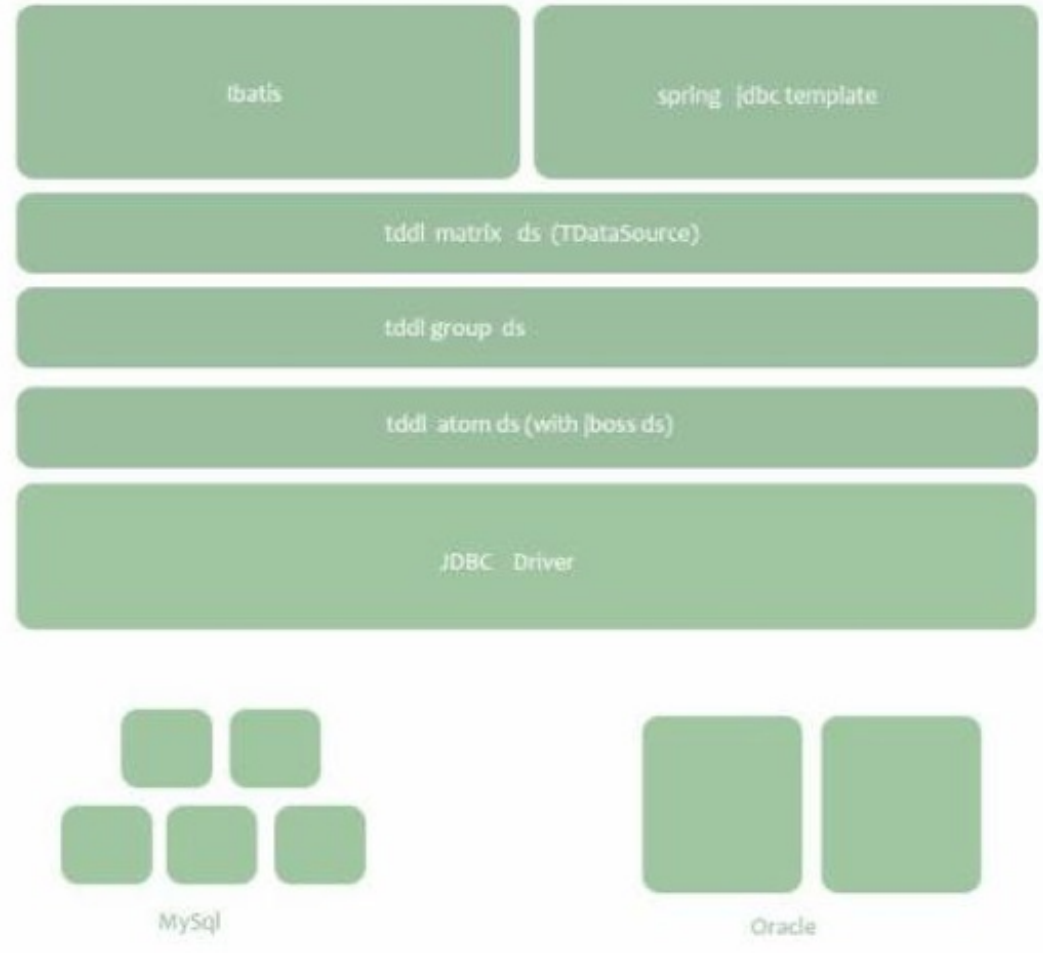


TDDL的主要優點：

- 數據庫主備和動態切換
- 帶權重的讀寫分離
- 單線程讀重試
- 集中式數據源信息管理和動態變更
- 剝離的穩定jboss數據源
- 支持mysql和oracle數據庫
- 基於jdbc規範，很容易擴展支持實現jdbc規範的數據源
- 無server,client-jar形式存在，應用直連數據庫
- 讀寫次數,並發度流程控制，動態變更
- 可分析的日誌打印,日誌流控，動態變更

TDDL的體系架構

TDDL其實主要可以劃分為3層架構，分別是Matrix層、Group層和Atom層。Matrix層用於實現分庫分表邏輯，底層持有多個Group實例。而Group層和Atom共同組成了動態數據源， Group層實現了數據庫的Master/Salve模式的寫分離邏輯，底層持有多個Atom實例。最後Atom層 (TAtomDataSource)實現數據庫ip,port,password,connectionProperties等信息的動態推送,以及持有原子的數據源分離的JBOSS數據源）。



持久層只關心對數據源的CRUD操作，而多數據源的訪問並不應該由它來關心。也就是說TDDL透明給持久層的數據

源接口應該是統一且「單一」的，至於數據庫到底如何分庫分表持久層無需知道也無需編寫對應的SQL去實行應對策略。這個時候對TDDL一些疑問就出現了，TDDL需要對SQL進行二次解析和拼裝嗎？答案是不解析僅拼裝。TDDL只需要從持久層拿到發出的SQL再按照一些分庫分表條件，進行特定的SQL擴充以此滿足訪問路路由操作。

1. TDDL除了拿到分庫分表條件外，還需要拿到order by、group by、limit、join等信息，SUM、MAX、MIN等聚合函數信息，DISTINCT信息。具有這些關鍵字的SQL將會在單庫和多庫情況下進行,語義是不同的。TDDL必須對使用這些關鍵字的SQL返回的結果做出合適的處理；
2. TDDL行複製需要重新拼寫SQL,帶上sync_version字段；
3. 不通過sql解析,因為TDDL遵守JDBC規範,它不可能去擴充JDBC規範裡面的接口,所以只能通過SQL中加額外的字符條件(也就是HINT方式)或者ThreadLocal方式進行傳遞,前者使SQL過長,後者難以維護,開發debug時不容易跟蹤,而且需要判定是在一條SQL執行後失效還是1個連接關閉後才失效；
4. TDDL現在也同時支持Hint方式和ThreadLocal方式傳遞這些信息；

參考鏈接：https://github.com/alibaba/tb_tddl

碼字很辛苦，轉載請註明來自**標點符**的《TDDL：來自淘寶的分佈式數據層》