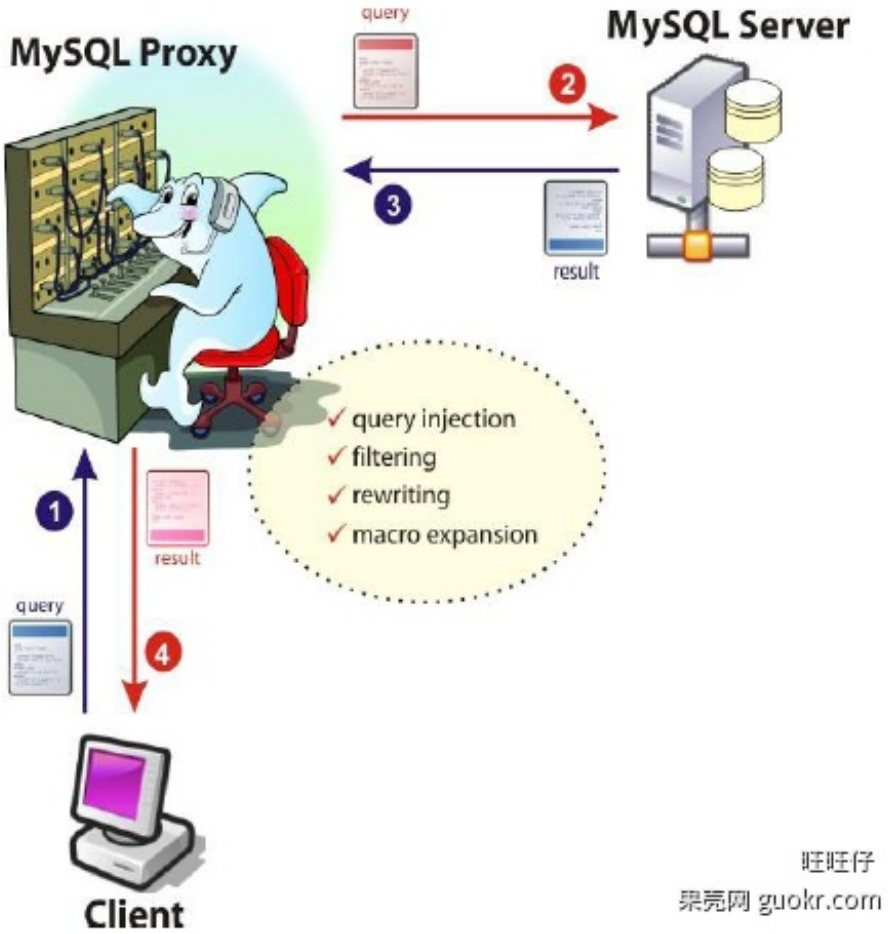
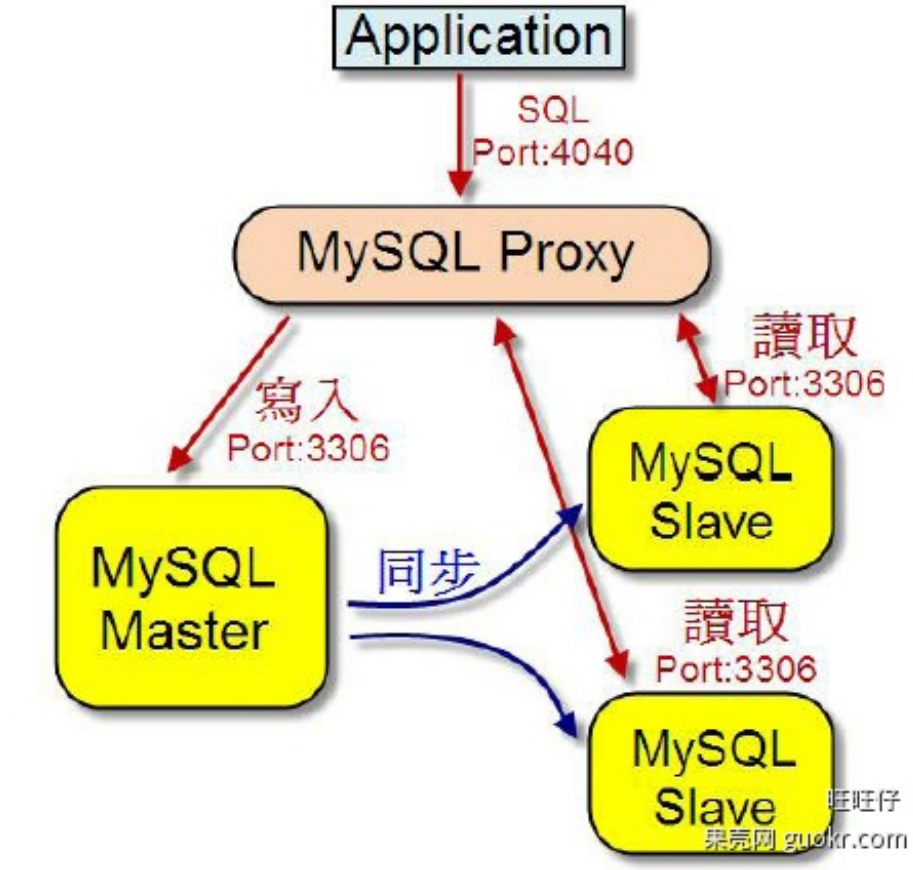
mysql中間件研究（Atlas，cobar，TDDL）

mysql-proxy是官方提供的mysql中間件產品可以實現負載平衡，讀寫分離，failover等，但其不支持大數據量的分庫分表且性能較差。下面介紹幾款能代替其的mysql開源中間件產品，Atlas，cobar，tddl，讓我們看看它們各自有些什麼優點和新特性吧。

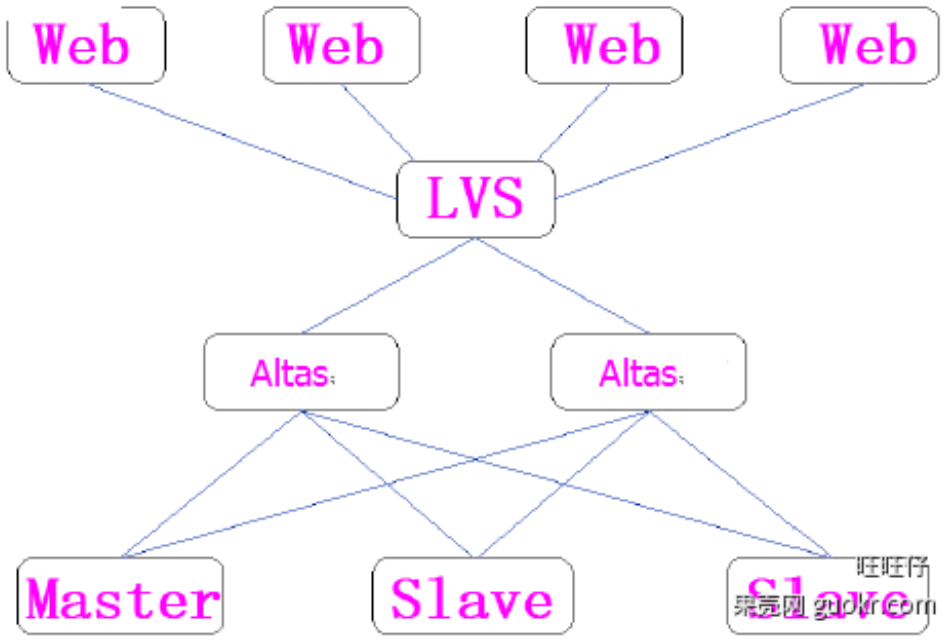
Atlas

Atlas是由 Qihoo 360, Web平台部基礎架構團隊開發維護的一個基於MySQL協議的數據中間層項目。它是在mysql-proxy 0.8.2版本的基礎上，對其進行了優化，增加了一些新的功能特性。360內部使用Atlas運行的mysql業務，每天承載的讀寫請求數達幾十億條。
Altas架構：
Atlas是一個位於應用程序與MySQL之間，它實現了MySQL的客戶端與服務端協議，作為服務端與應用程序通訊，同時作為客戶端與MySQL通訊。它對應用程序屏蔽了DB的細節，同時為了降低MySQL負擔，它還維護了連接池。





以下是一個可以參考的整體架構，LVS前端做負載均衡，兩個Altas做HA,防止單點故障。



Altas的一些新特性：
1.主庫宕機不影響讀
主庫宕機，Atlas自動將宕機的主庫摘除，寫操作會失敗，讀操作不受影響。從庫宕機，Atlas自動將宕機的從庫摘除，對應用沒有影響。在mysql官方的proxy中主庫宕機，從庫亦不可用。
2.通過管理接口，簡化管理工作，DB的上下線對應用完全透明，同時可以手動上下線。
圖1是手動添加一台從庫的示例。

圖1

```
mysql> select * from backends;
+-------------+----------------+-------+------+------+-------------------+
| backend_ndx | address        | state | type | uuid | connected_clients |
+-------------+----------------+-------+------+------+-------------------+
|           1 | 127.0.0.1:3306 | up    | rw   | NULL |                 0 |
+-------------+----------------+-------+------+------+-------------------+
1 row in set (0.00 sec)

mysql> add slave 127.0.0.1:3307;
Empty set (0.01 sec)

mysql> select * from backends;
+-------------+----------------+---------+------+------+-------------------+
| backend_ndx | address        | state   | type | uuid | connected_clients |
+-------------+----------------+---------+------+------+-------------------+
|           1 | 127.0.0.1:3306 | up      | rw   | NULL |                 0 |
|           2 | 127.0.0.1:3307 | unknown | ro   | NULL |                 0 |
+-------------+----------------+---------+------+------+-------------------+
2 rows in set (0.00 sec)

mysql> select * from backends;
+-------------+----------------+-------+------+------+-------------------+
| backend_ndx | address        | state | type | uuid | connected_clients |
+-------------+----------------+-------+------+------+-------------------+
|           1 | 127.0.0.1:3306 | up    | rw   | NULL |                 0 |
|           2 | 127.0.0.1:3307 | up    | ro   | NULL |                 0 |
+-------------+----------------+-------+------+------+-------------------+
2 rows in set (0.00 sec)
```


3.自己實現讀寫分離

（1）為瞭解決讀寫分離存在寫完馬上就想讀而這時可能存在主從同步延遲的情況，Altas中可以在SQL語句前增加/*master*/ 就可以將讀請求強制發往主庫。

（2）如圖2中，主庫可設置多項，用逗號分隔，從庫可設置多項和權重，達到負載均衡。

圖2

```
proxy-backend-addresses = 127.0.0.1:3306

proxy-read-only-backend-addresses = 127.0.0.1:3307@1
```

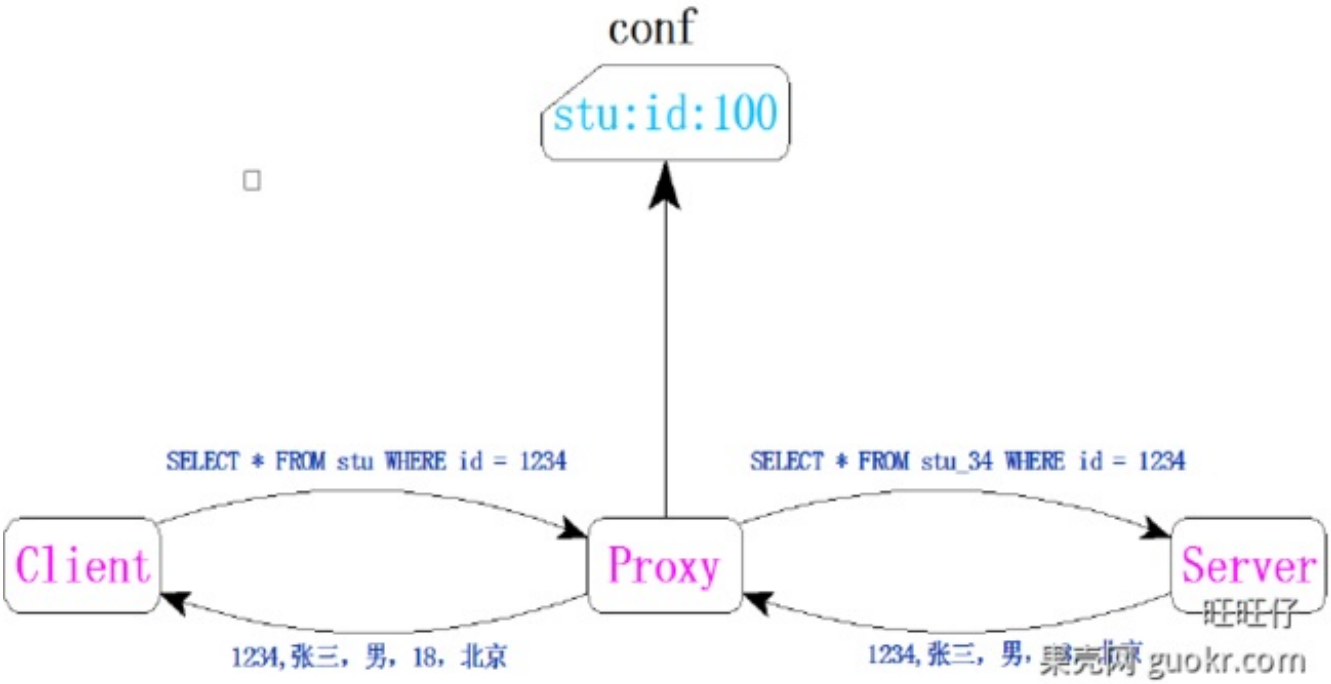

4.自己實現分表（圖3）

（1）需帶有分表字段。

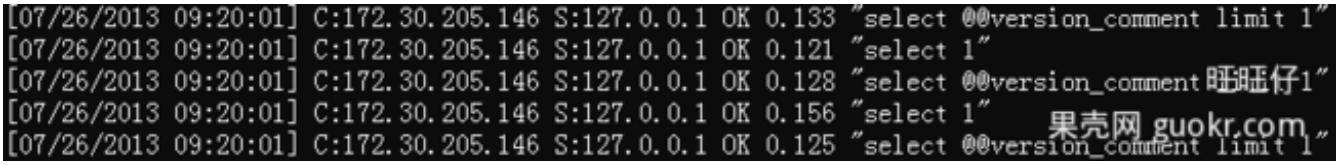
（2）支持SELECT、INSERT、UPDATE、DELETE、REPLACE語句。

（3）支持多個子表查詢結果的合併和排序。

圖3



這裡不得不吐槽Atlas的分表功能，不能實現分佈式分表，所有的子表必須在同一台DB的同一個database裡且所有的子表必須事先建好，Atlas沒有自動建表的功能。
5.之前官方主要功能邏輯由使用lua腳本編寫，效率低，Atlas用C改寫，QPS提高，latency降低。
6.安全方面的提升
（1）通過配置文件中的pwds參數進行連接Atlas的用戶的權限控制。
（2）通過client-ips參數對有權限連接Atlas的ip進行過濾。
（3）日誌中記錄所有通過Altas處理的SQL語句，包括客戶端IP、實際執行該語句的DB、執行成功與否、執行所耗費的時間 ，如下面例子（圖4）。
圖4

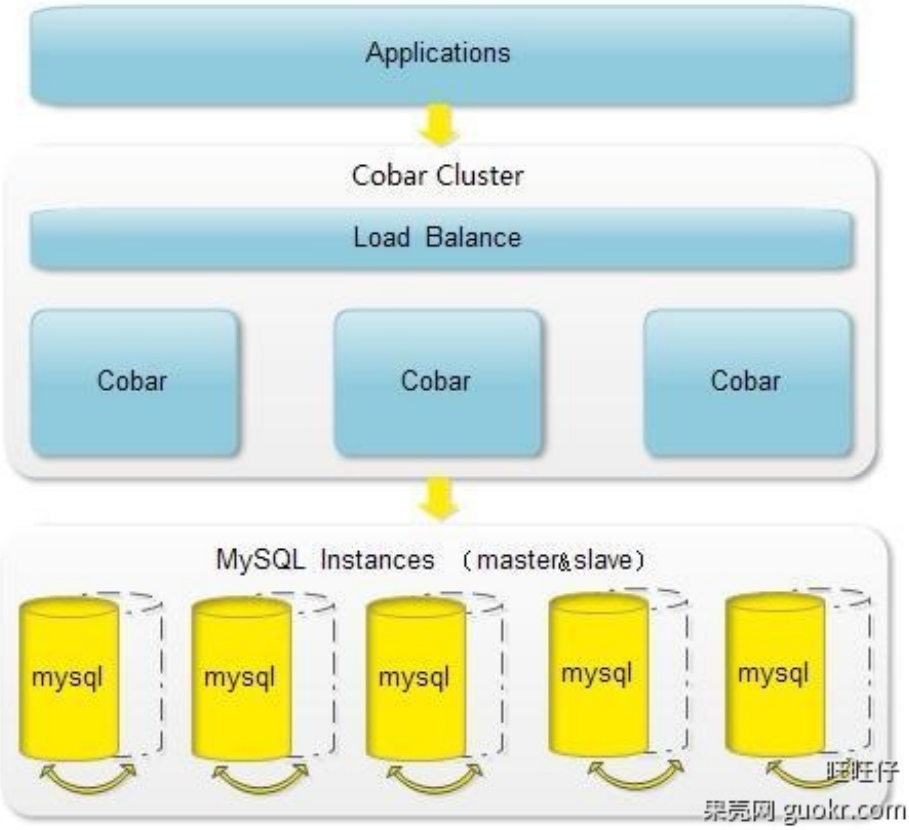


7.平滑重啟
通過配置文件中設置lvs-ips參數實現平滑重啟功能，否則重啟Altas的瞬間那些SQL請求都會失敗。該參數前面掛接的lvs的物理網卡的ip，注意不是虛ip。平滑重啟的條件是至少有兩台配置相同的Atlas且掛在lvs之後。
source：https://github.com/Qihoo360/Atlas

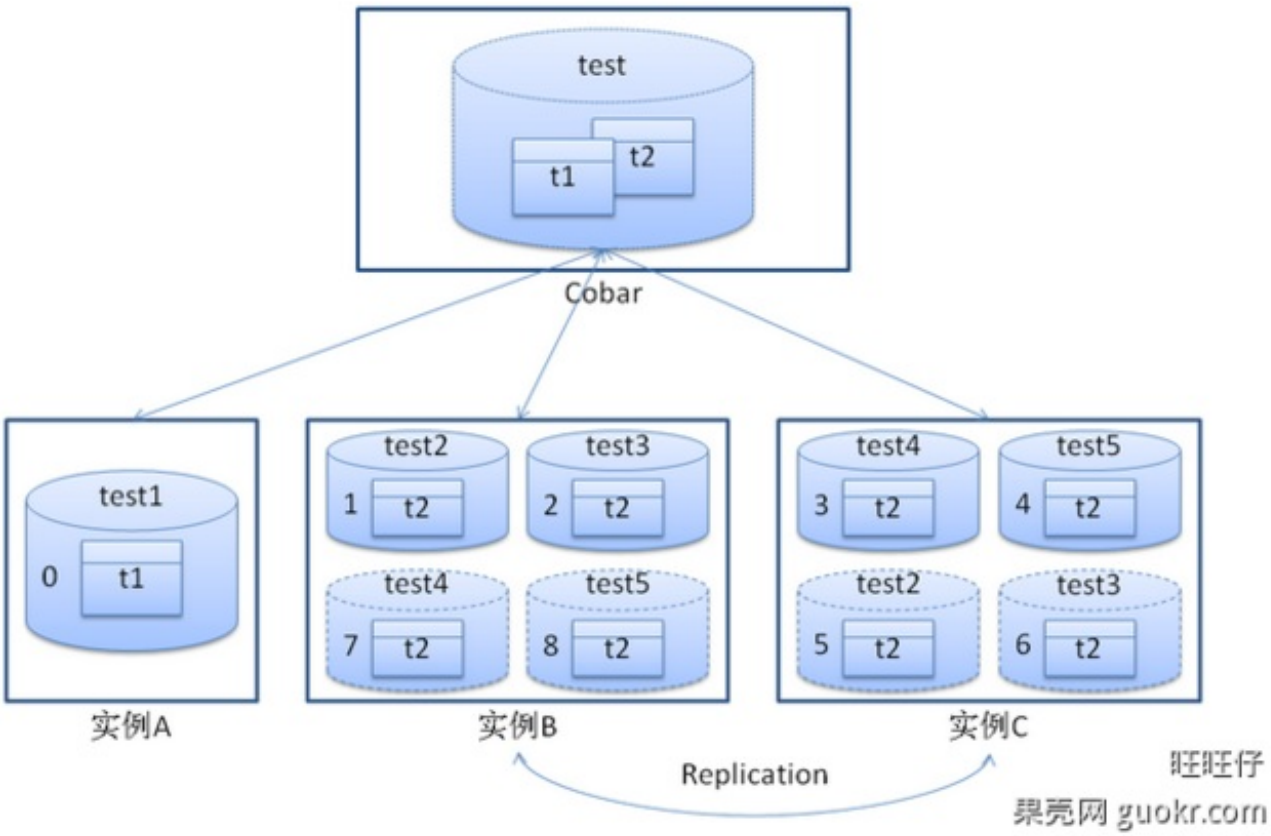
alibaba.cobar

Cobar是阿里巴巴（B2B）部門開發的一種關係型數據的分佈式處理系統，它可以在分佈式的環境下看上去像傳統數據庫一樣為您提供海量數據服務。那麼具體說說我們為什麼要用它，或說cobar--能幹什麼？以下是我們業務運行中會存在的一些問題：
1.隨著業務的進行數據庫的數據量和訪問量的劇增，需要對數據進行水平拆分來降低單庫的壓力，而且需要高效且相對透明的來屏蔽掉水平拆分的細節。
2.為提高訪問的可用性，數據源需要備份。

3.數據源可用性的檢測和failover。
4.前台的高並發造成後台數據庫連接數過多，降低了性能，怎麼解決。
針對以上問題就有了cobar施展自己的空間了，cobar中間件以proxy的形式位於前台應用和實際數據庫之間，對前台的開放的接口是mysql通信協議。將前台SQL語句變更並按照數據分佈規則轉發到合適的後台數據分庫，再合併返回結果，模擬單庫下的數據庫行為。



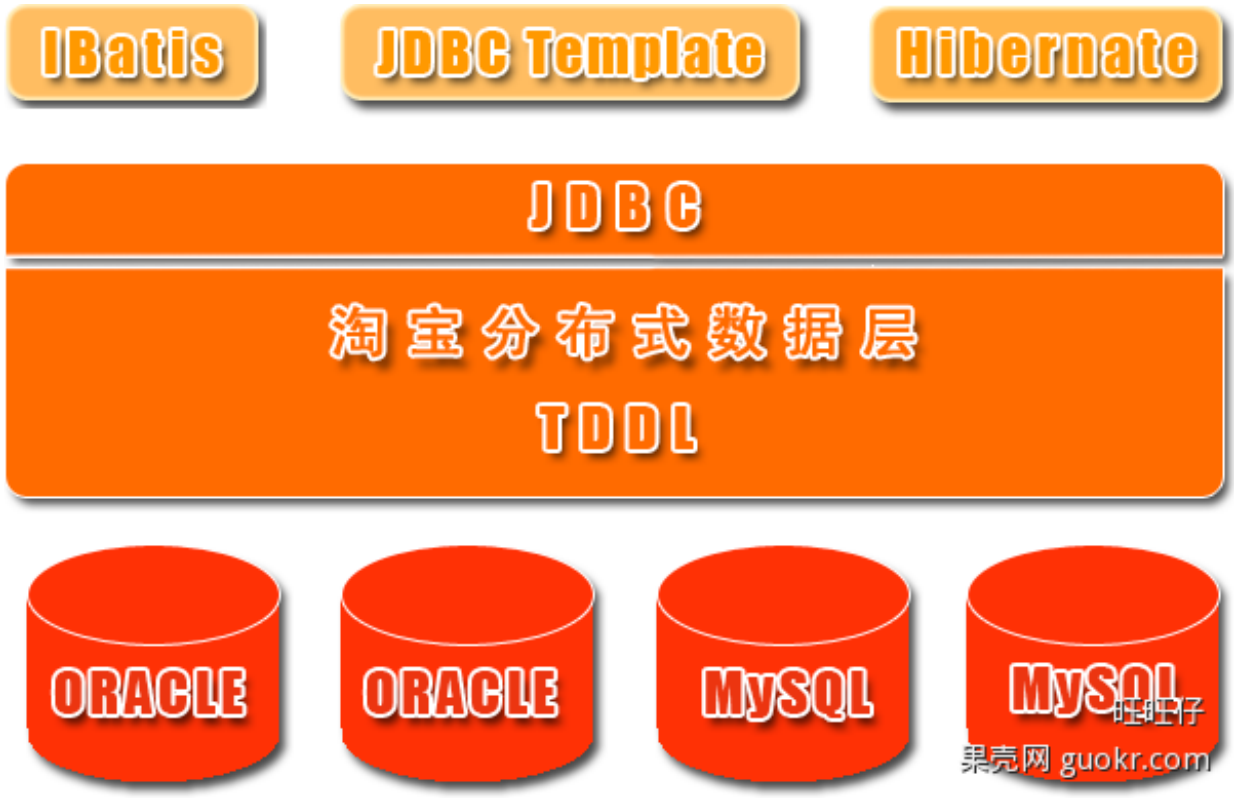
Cobar應用舉例
應用架構：



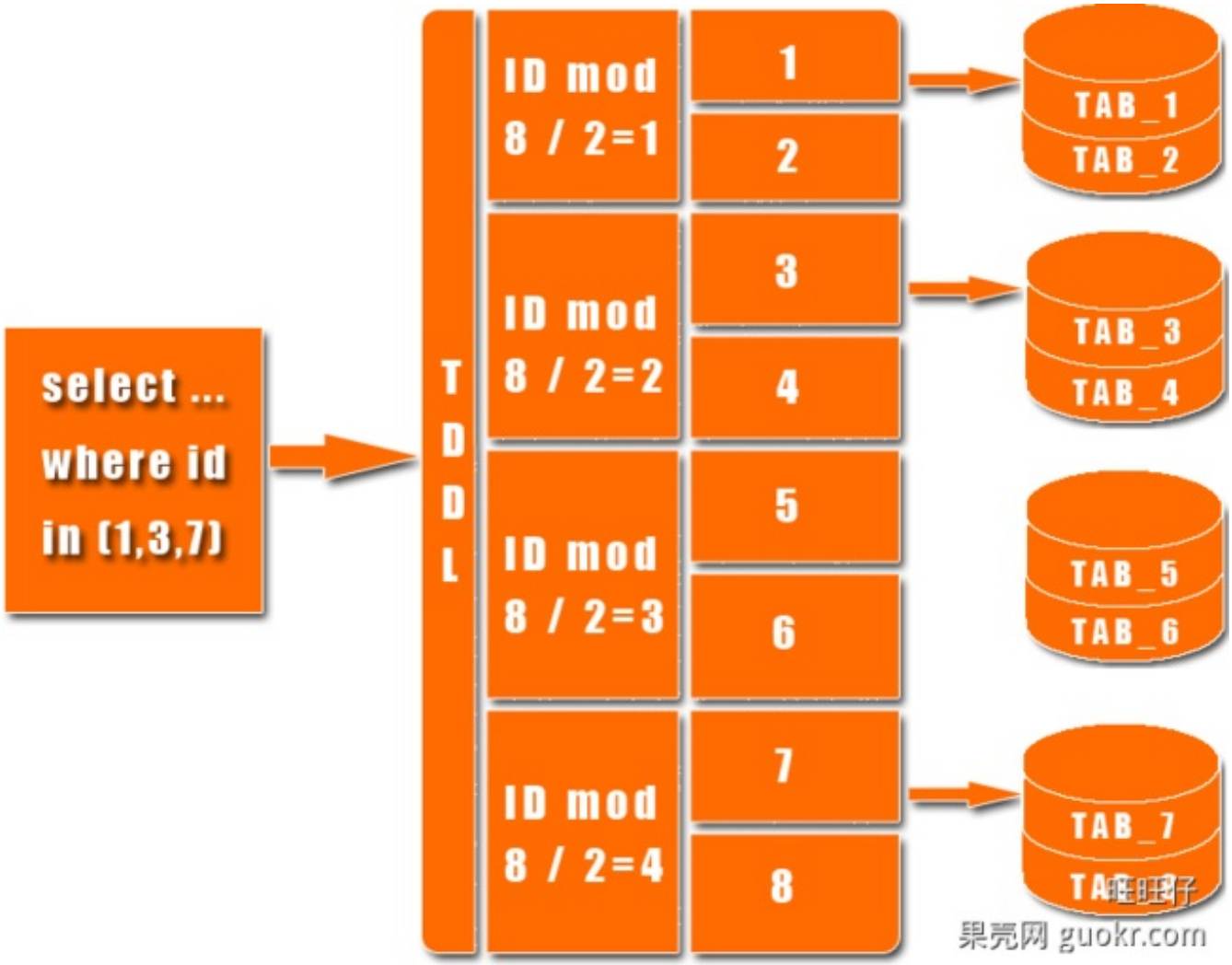
應用介紹：
1.通過Cobar提供一個名為test的數據庫，其中包含t1,t2兩張表。後台有3個MySQL實例(ip:port)為其提供服務，分別為：A,B,C。
2.期望t1表的數據放置在實例A中，t2表的數據水平拆成四份並在實例B和C中各自放兩份。t2表的數據要具備HA功能，即B或者C實例其中一個出現故障，不影響使用且可提供完整的數據服務。
cabar優點總結：
1.數據和訪問從集中式改變為分佈：
（1）Cobar支持將一張表水平拆分成多份分別放入不同的庫來實現表的水平拆分
（2）Cobar也支持將不同的表放入不同的庫
（3） 多數情況下，用戶會將以上兩種方式混合使用
注意！：Cobar不支持將一張表，例如test表拆分成test_1,test_2, test_3.....放在同一個庫中，必須將拆分後的表分別放入不同的庫來實現分佈式。
2.解決連接數過大的問題。
3.對業務代碼侵入性少。
4.提供數據節點的failover,HA：
(1)Cobar的主備切換有兩種觸發方式，一種是用戶手動觸發，一種是Cobar的心跳語句檢測到異常後自動觸發。那麼，當心跳檢測到主機異常，切換到備機，如果主機恢復了，需要用戶手動切回主機工作，Cobar不會在主機恢復時自動切換回主機，除非備機的心跳也返回異常。
(2)Cobar只檢查MySQL主備異常，不關心主備之間的數據同步，因此用戶需要在使用Cobar之前在MySQL主備上配置雙向同步。
cobar缺點：
開源版本中數據庫只支持mysql，並且不支持讀寫分離。
source：http://code.alibabatech.com/wiki/display/cobar/Home

TDDL

淘寶根據自己的業務特點開發了TDDL（Taobao Distributed Data Layer 外號:頭都大了 ©_Ob）框架，主要解決了分庫分表對應用的透明化以及異構數據庫之間的數據複製，它是一個基於集中式配置的 jdbc datasource實現，具有主備，讀寫分離，動態數據庫配置等功能。
TDDL所處的位置（tddl通用數據訪問層，部署在客戶端的jar包，用於將用戶的SQL路由到指定的數據庫中）：



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：



主要優點：
1.數據庫主備和動態切換
2.帶權重的讀寫分離
3.單線程讀重試
4.集中式數據源信息管理和動態變更
5.剝離的穩定jboss數據源
6.支持mysql和oracle數據庫
7.基於jdbc規範，很容易擴展支持實現jdbc規範的數據源
8.無server,client-jar形式存在，應用直連數據庫
9.讀寫次數,並發度流程控制，動態變更
10.可分析的日誌打印,日誌流控，動態變更
TDDL必須要依賴diamond配置中心（diamond是淘寶內部使用的一個管理持久配置的系統，目前淘寶內部絕大多數系統的配置，由diamond來進行統一管理，同時diamond也已開源）。
TDDL動態數據源使用示例說明：http://rdc.taobao.com/team/jm/archives/1645
diamond簡介和快速使用：http://jm.taobao.org/tag/diamond%E4%B8%93%E9%A2%98/
TDDL源碼：https://github.com/alibaba/tb_tddl
TDDL複雜度相對較高。當前公佈的文檔較少，只開源動態數據源，分表分庫部分還未開源，還需要依賴diamond，不推薦使用。
終其所有，我們研究中間件的目的是使數據庫實現性能的提高。具體使用哪種還要經過深入的研究，嚴謹的測試才可決定。