**基于SymmetricDS的多主一从数据库同步方案**

下面就记录下测试的一款第三方同步方案SymmetricDS(以下简称S)的使用过程，中文资料较少，而且存在一些版本上的差异，导致一些步骤根本不能通过或报错，自己简化了些操作，并没有按照官方的指导操作，鉴于英文水平有限，很多叙述都是基于自己的理解，也请有不同观点的兄弟留言指正或交流。

之所以选择SymmetricDS，大致三个原因：

       1.  平台独立。不依赖其他组件包，独立配置完后实现功能。

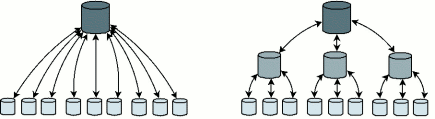
       2.  业务无关。不需要关心数据库中的业务，直接配置同步的表名等参数。

       3.  同步及时。基于触发器的数据同步，保证数据传输的及时性。

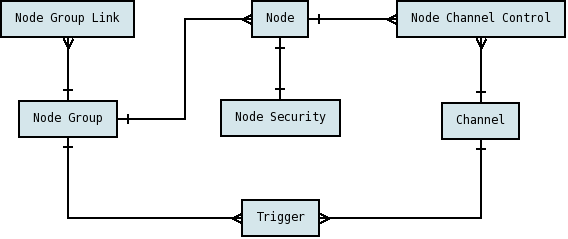
官方地址：http://www.symmetricds.org

版本：[symmetric-3.5.19-server.zip](http://sourceforge.net/projects/symmetricds/files/latest/download)

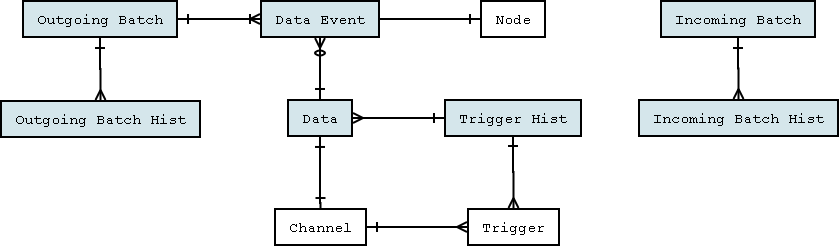
下面是开源中国的一些图表介绍，引用下：



部署示意图，双箭头代表可以双向同步，而不是简单的单向复制。

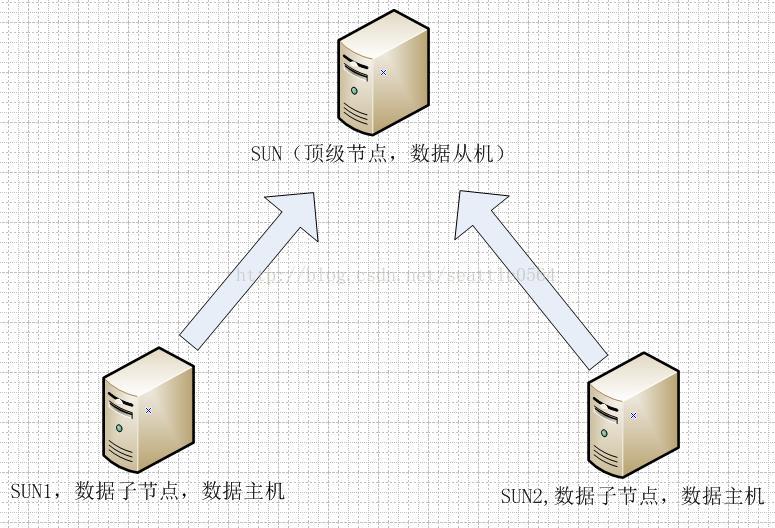


配置数据模型：对于各个服务器，不管是主机还是从机，都需要配置S服务，用来监听或执行动作等，并且是以节点代表服务器来配置，至于触发器更体现它的实时性，一方数据库发生变化，首先被本机S服务监听，同时向关联的节点发起同步请求，关联节点接收请求并做响应动作。



  运行时数据模型：当每个节点监听到本机的数据事件(即数据变动)，会将变动的数据通过触发器与关联节点进行通迅。

测试方案拓扑图如下。sun1、sun2 向 sun进行单向数据同步。



因实际硬件测试环境限制，本地新建三个数据库分别模拟sun，sun1，sun2

http://img.blog.csdn.net/20140328143406968?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQvc2VhdHRsZTA1NjQ=/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/Center

每个节点都需要配置一个s服务，将S服务代码分别解压到了三个目录:

D:\develop\symmetric\sun\symmetric

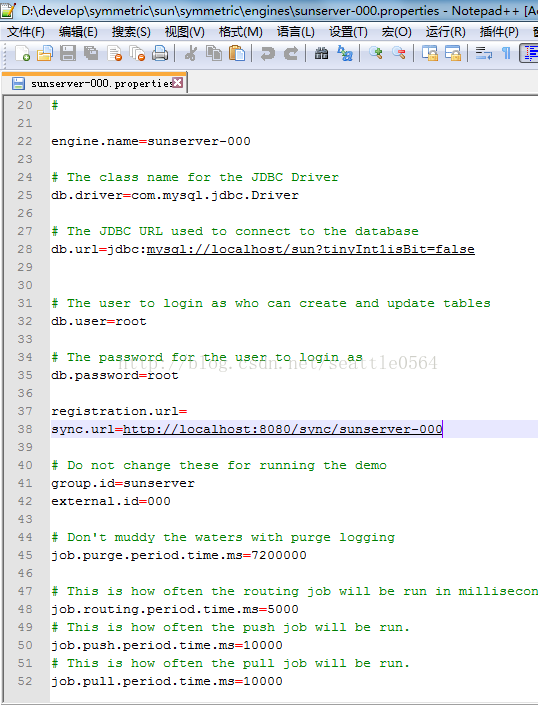
D:\develop\symmetric\sun1\symmetric

D:\develop\symmetric\sun2\symmetric



**一、首先配置从机sun的S服务。**进入到D:\develop\symmetric\sun\symmetric目录，将samples目录下的corp-000.properties文件拷贝到engines目录下，samples目录下是官网提供的一个范例，engines是你配置自己同步服务的目录，corp-000.properties可以说是配置从机节点参数的一个模板，所以拷贝出来做修改，文件名可以自定义修改，修改为sunserver-000.properties。

按照如下模板修改：



engine.name 引擎名称，可以理解成当前节点服务的名称

db.driver 数据库的jdbc驱动名称

db.url 数据库连接字符串

db.user 数据库用户名

db.password 数据库密码

registration.url 上一级节点的注册地址，如果是顶级节点，就设为空

sync.url 本机的注册地址 http://localhost:8080/sync/sunserver-000

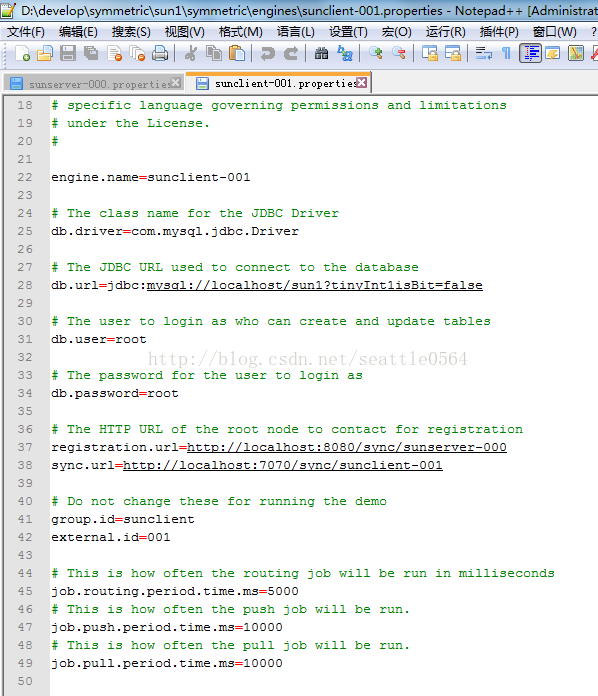
其中8080是当前节点服务启动后需要占用的端口，sunserver-000即第一个参数engine.name的值

group.id 节点组的ID，当前顶级节点组名为sunserver，意思是服务端

external.id 当前节点组下节点的编号，通过上面的group.id和external.id可以定位到唯一的节点

后面的参数是对S服务运行参数的配置，保持默认就好。

**二、配置子节点sun1。**配置好顶级节点从机sun的服务，接下来就该配置子节点数据主机sun1，sun2的了，先配置sun1.同样的，进入到D:\develop\symmetric\sun1\symmetric目录,将samples目录下的store-001.properties文件拷贝到engines目录下，这里的store-001.properties也可以说是配置数据主机节点参数的一个模板，配置内容与sun类似，这里改名为sunclient-001.properties



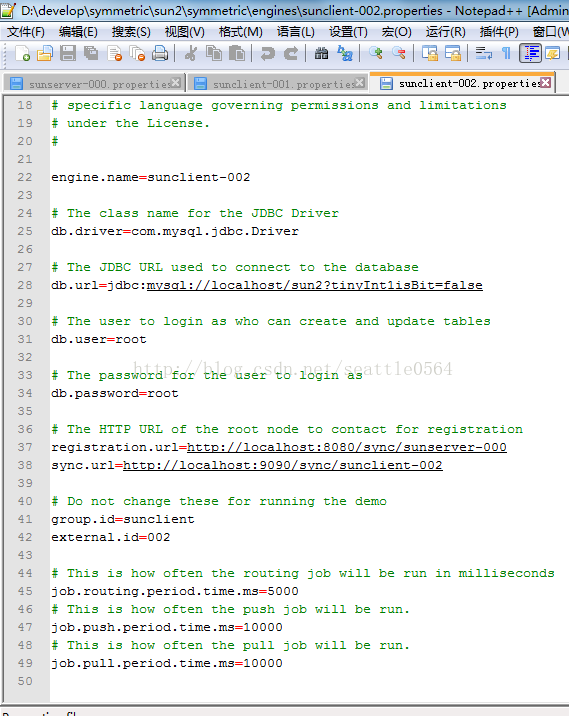
配置基本上与sun节点类似，其中有些需要关注的是：

registration.url 这里填写的是上级节点的注册地址，可以看到就是sun配置参数中的sync.url

这里的sync.url其实不用配置，因为节点到此为止。如果sun1以下还有节点分支，那需要配置这个参数，并且在下一个节点的registration.url配置这个值。需要注意的是，我当前是在单机上模拟三个节点服务的运行，所以每个节点服务的启动端口不同，避免冲突。sun1的启动端口是7070

group.id 这里的组ID为sunclient，意思是客户端。

**三、配置sun2节点。**进入到D:\develop\symmetric\sun2\symmetric目录,后续步骤与上一步相同。



**四、创建同步表数据。**接下来，需要向同步的数据库进行数据导入，官网的方式是通过命令行进入到Symmetric的目录，通过命令执行sql脚本创建表和数据，但我尝试了几次，都没有成功，有些资料说是因为官网相关XML文件格式的问题，不管咋样，索性自己手动创建表和数据。

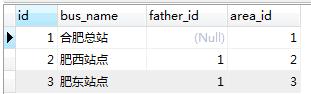
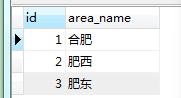
    所以要做的，就是在数据库中创建你要同步的表，如果数据库中已经有表结构了，可不做操作了。

    创建后的表为

http://img.blog.csdn.net/20140328143844406?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQvc2VhdHRsZTA1NjQ=/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/Center

区域表，电站表。

具体表结构也一目了然:



所有的节点数据库sun、sun1、sun2都需手动创建和添加数据，并且理论上数据库的结构应该是保持一致的。

**五、初始化顶级节点sun的系统表结构。**这里的系统表，指的就是Symmetric自己的同步服务表，并且所有的节点数据库中都需要有这些系统表(这也是我觉得悲催的地方，光这些系统表就二三十个，忒不美观了)。稍微庆幸的是，我们只要在顶级节点中创建这些表，所有子节点在服务启动后，会根据顶级节点自动创建。

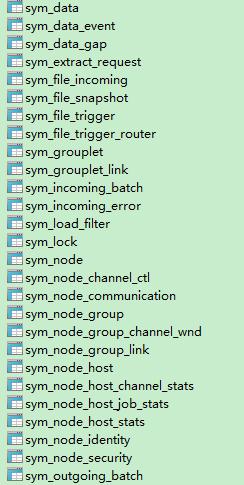
通过命令行进入到D:\develop\symmetric\sun\symmetric\engines目录

执行

**[plain]** [view plain](http://blog.csdn.net/seattle0564/article/details/22096901) [copy](http://blog.csdn.net/seattle0564/article/details/22096901)

1. ..\bin\symadmin --engine sunserver-000 create-sym-tables

运行成功，完成初始化系统表。此时可以查看下数据库，发现多了好多表啊。



sym前缀的表 都是，表数量不止于此，还不是完整截图。:-(

**六、初始化顶级节点sun的系统表数据。**上一步骤是初始化表结构，下面就要初始化数据了。其实说到现在，我们都还不知道怎么控制SymmetricDS同步的逻辑，比如需要同步哪些表，哪个节点向哪个节点发送同步数据，这些都体现在系统表数据中了。

        至于如何初始化系统表数据，我参照了samples目录中insert\_sample.sql脚本文件，官方范例中系统表的初始化脚本,另外还参照了百度文库中一篇说明。

**（1）配置节点组**

**[sql]** [view plain](http://blog.csdn.net/seattle0564/article/details/22096901) [copy](http://blog.csdn.net/seattle0564/article/details/22096901)

1. insert into sym\_node\_group (node\_group\_id, description)
2. values ('sunserver', '电站数据中心');
3. insert into sym\_node\_group (node\_group\_id, description)
4. values ('sunclient', '子电站数据服务器');

**（2）配置各节点组间的数据同步模式。有push（推）和wait（等待||拉）两种，由data\_event\_action制定，其中w（wait for pull）代表拉，p代表推（push）。**

**[sql]** [view plain](http://blog.csdn.net/seattle0564/article/details/22096901) [copy](http://blog.csdn.net/seattle0564/article/details/22096901)

1. insert into sym\_node\_group\_link (source\_node\_group\_id, target\_node\_group\_id, data\_event\_action)
2. values ('sunclient', 'sunserver', 'P');
3. insert into sym\_node\_group\_link (source\_node\_group\_id, target\_node\_group\_id, data\_event\_action)
4. values ('sunserver', 'sunclient', 'W');

**（3）配置各个节点信息。这里只要配置sun的节点信息就可，其他子节点sun1、sun2在注册节点后，会自动生成系统表并且插入数据**

**[sql]** [view plain](http://blog.csdn.net/seattle0564/article/details/22096901) [copy](http://blog.csdn.net/seattle0564/article/details/22096901)

1. insert into sym\_node (node\_id, node\_group\_id, external\_id, sync\_enabled)
2. values ('000', 'sunserver', '000', 1);
3. insert into sym\_node\_security (node\_id,node\_password,registration\_enabled,registration\_time,initial\_load\_enabled,initial\_load\_time,initial\_load\_id,initial\_load\_create\_by,rev\_initial\_load\_enabled,rev\_initial\_load\_time,rev\_initial\_load\_id,rev\_initial\_load\_create\_by,created\_at\_node\_id)
4. values ('000','123456',0,current\_timestamp,0,current\_timestamp,null,null,0,null,null,null,'000');
5. insert into sym\_node\_identity values ('000');

**（4）配置数据通道表。有外键关联的表，一定要定义在同一通道中，才能进行关联同步。**

**[sql]** [view plain](http://blog.csdn.net/seattle0564/article/details/22096901) [copy](http://blog.csdn.net/seattle0564/article/details/22096901)

1. insert into sym\_channel
2. (channel\_id, processing\_order, max\_batch\_size, enabled, description)
3. values('bus\_info', 1, 100000, 1, '电站信息同步通道');

**（5）定义触发器。在这里定义需要同步库中的哪些表，注意有外键关联的表，通道值要相同。sym\_trigger中的excluded\_column\_names字段，可以设置不想同步的表字段，这里并没有体现。**

**[sql]** [view plain](http://blog.csdn.net/seattle0564/article/details/22096901) [copy](http://blog.csdn.net/seattle0564/article/details/22096901)

1. insert into sym\_trigger
2. (trigger\_id,source\_table\_name,channel\_id,last\_update\_time,create\_time)
3. values('symmetric\_test\_area','symmetric\_test\_area','bus\_info',current\_timestamp,current\_timestamp);
4. insert into sym\_trigger
5. (trigger\_id,source\_table\_name,channel\_id,last\_update\_time,create\_time)
6. values('symmetric\_test\_bus','symmetric\_test\_bus','bus\_info',current\_timestamp,current\_timestamp);

**（6）配置数据路由。配置数据同步时数据的走向，即从哪个节点向哪个节点同步。其中ROUTER\_TYPE='column'就是指明由需同步的表中某列的值来决定数据流向，具体的条件则来表达式决定：ROUTER\_EXPRESSION='org\_code=:EXTERNAL\_ID' 。默认ROUTER\_TYPE=‘default’就好。**

**[sql]** [view plain](http://blog.csdn.net/seattle0564/article/details/22096901) [copy](http://blog.csdn.net/seattle0564/article/details/22096901)

1. insert into sym\_router
2. (router\_id,source\_node\_group\_id,target\_node\_group\_id,router\_type,create\_time,last\_update\_time)
3. values('client\_2\_server', 'sunclient', 'sunserver', 'default',current\_timestamp, current\_timestamp);

**（7）最后建立触发器与路由的关联。SymmetricDS中定义的触发器只有被路由关联后，SymmetricDS才会为这个表自动生成相应的触发器。**

**[sql]** [view plain](http://blog.csdn.net/seattle0564/article/details/22096901) [copy](http://blog.csdn.net/seattle0564/article/details/22096901)

1. insert into sym\_trigger\_router
2. (trigger\_id,router\_id,initial\_load\_order,last\_update\_time,create\_time)
3. values('symmetric\_test\_area','client\_2\_server', 200, current\_timestamp, current\_timestamp);
4. insert into sym\_trigger\_router
5. (trigger\_id,router\_id,initial\_load\_order,last\_update\_time,create\_time)
6. values('symmetric\_test\_bus','client\_2\_server', 200, current\_timestamp, current\_timestamp);

**七、启动SymmetricDS**

（1）注册所有子节点，命令行进入D:\develop\symmetric\sun\symmetric\engines 顶级节点目录。

分别执行注册节点命令

**[plain]** [view plain](http://blog.csdn.net/seattle0564/article/details/22096901) [copy](http://blog.csdn.net/seattle0564/article/details/22096901)

1. ..\bin\symadmin --engine sunserver-000 open-registration sunclient 001
2. ..\bin\symadmin --engine sunserver-000 open-registration sunclient 002

再分别执行注入节点命令

**[plain]** [view plain](http://blog.csdn.net/seattle0564/article/details/22096901) [copy](http://blog.csdn.net/seattle0564/article/details/22096901)

1. ..\bin\symadmin --engine sunserver-000 reload-node 001
2. ..\bin\symadmin --engine sunserver-000 reload-node 002

（2）启动服务。

分别启动三个节点的服务，先执行子节点。

命令行进入 D:\develop\symmetric\sun1\symmetric\engines

执行

**[plain]** [view plain](http://blog.csdn.net/seattle0564/article/details/22096901) [copy](http://blog.csdn.net/seattle0564/article/details/22096901)

1. ..\bin\sym --port 7070

命令行进入 D:\develop\symmetric\sun2\symmetric\engines

执行 .

**[plain]** [view plain](http://blog.csdn.net/seattle0564/article/details/22096901) [copy](http://blog.csdn.net/seattle0564/article/details/22096901)

1. ..\bin\sym --port 9090

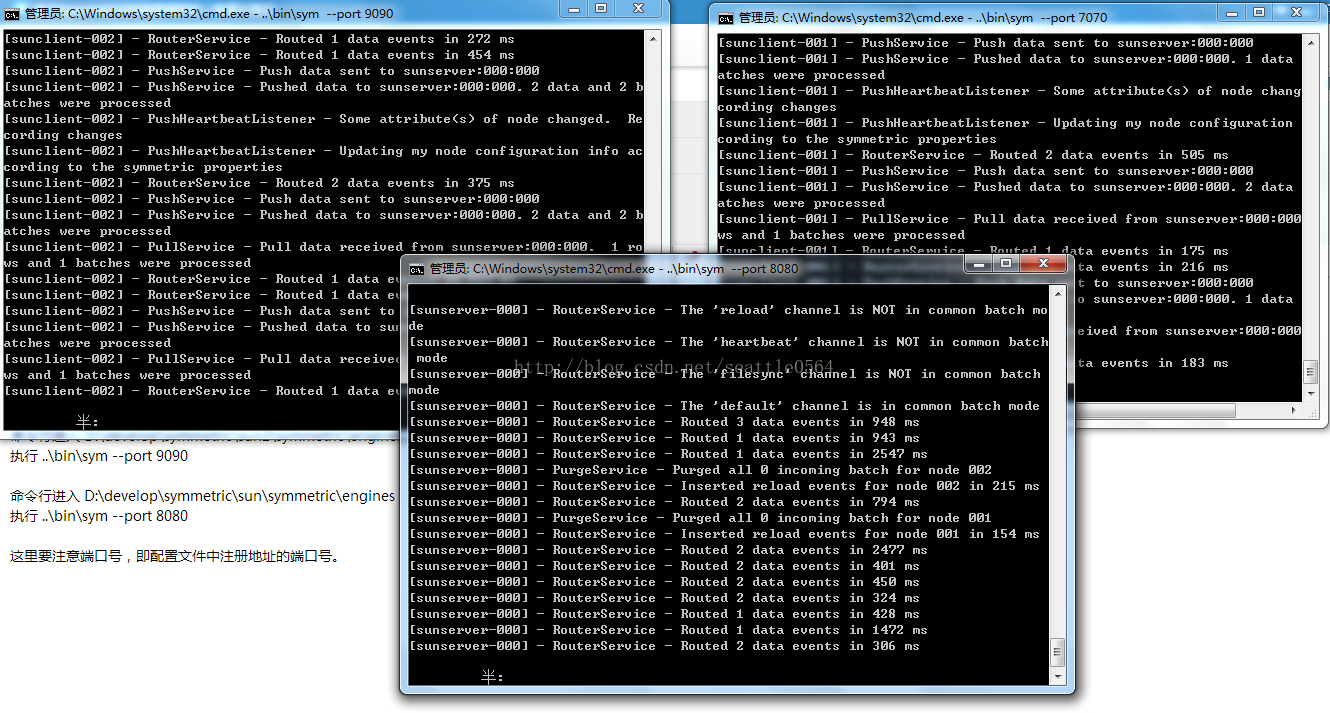
命令行进入 D:\develop\symmetric\sun\symmetric\engines

执行

**[plain]** [view plain](http://blog.csdn.net/seattle0564/article/details/22096901) [copy](http://blog.csdn.net/seattle0564/article/details/22096901)

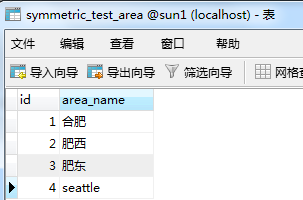
1. ..\bin\sym --port 8080

这里要注意端口号，即配置文件中注册地址的端口号。

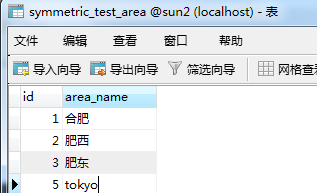


至此 三个服务都成功开启。下面可以进行测试了。。

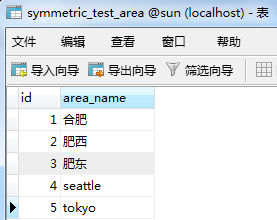
在sun1数据库symmetric\_test\_area表中新增一条数据4--seattle



在sun2数据库symmetric\_test\_area表中新增一条数据5--tokyo



最后直接查看sun的数据库，会发现：



**大功告成！**

**总结：**

SymmetricDS功能很复杂，这里只是暂时实现了多主一从结构的配置，深入到具体的同步细节，还需要了解很多。

优点：同步即时，基于推拉双机制的同步，对于数据的及时性完整性有保障。关于事件的并发与线程管理已封装。表结构变动对于配置好的数据字段的同步影响不大，但对于新增的字段如果也需要同步的话，需要重新注册节点，并重启服务。

缺点：每个节点都需要配置服务代码，且比较复杂。会在每个节点数据库中生成41张同步系统表，业务的控制体现在表数据中。适合用于固定或长期稳定的网络通道的机房环境。如果整合进产品往外推广，配置复杂后期维护也麻烦，适合本公司内部服务器的应用。

补充：其实我觉得最大的优点就是，对于数据同步这个需求，symmetricDS只需要配置不关心每个数据库的具体业务，只关心要同步哪个数据库哪张表。如果是写服务代码来定时抓取数据再插入数据的话，可能还要考虑到数据库中的业务，比如哪个表关联哪个表，插入数据的先后还有级联问题等，对于symmetricDS来说只要关联表在一个同步通道中即可。