Using SQLTXPLAIN with Data Guard Physical Standby Databases

SQLTXPLAIN is a great tool designed to help with SQL tuning problems, but its effectiveness is limited when the database is a read-only Data Guard physical standby database. Read only databases like a Data Guard physical standby databases cannot be written to by utilities such as SQLTXPLAIN. In this chapter we explore the special tools created just to deal with Data Guard.

Data Guard Physical Standby Database

Data Guard is a piece of technology, developed by Oracle in response to a “what if” scenario. What if my data center is completely wiped out by flooding? What if a fire destroys the building my database is housed in? Traditionally this has been answered by timely backups, shipped off site and stored in a secure location that can then be accessed within an agreed time scale and restored to backup hardware made available at another site (presumably a site not in the disaster zone). This strategy had many difficulties: taking the backup, shipping the backup to a safe location, getting the right backup from the secure location, recreating your systems with the use of the backup and your restore procedures, and finally getting access to those systems for the staff required to use them. Finally and most importantly, you need to test these procedures on a regular basis, otherwise come the day of the disaster you may have all the data and equipment, but you’ll have no idea how to put it all together to make a working system. Of course, this restore procedure can take a considerable amount of time. You have tested your recovery procedure haven’t you? One site I worked with took this so seriously that on the day of the test, they would go around and put red stickers on people and equipment and tell the people they were not available for this test (presumably they had been abducted by the aliens in the “Aliens abduct your data center” scenario). It was hard to test these scenarios; and not surprisingly, many sites did not do adequate testing and crossed their IT fingers and hoped for the best.

Data Guard makes preserving your data site much simpler. All of the aforementioned complications can potentially be eliminated. The technology that makes Data Guard physical standby work is in concept very simple. The archive logs (or redo logs), that track every change in your database are transferred by special processes on your source system to special processes on the standby system, where the changes are applied, block by block to a copy of your source database. This process called propagation can apply data on the standby database. The data can be applied in lock step with the primary database or it can lag behind the primary database by a preset amount. You can even have multiple standby databases all collecting changes from the primary database. With Data Guard active standby, you can even allow the failover (the process whereby the primary fails and the standby has to take over) to happen with the minimum of fuss.

Data Guard is a huge leap forward in disaster-recovery technology. Complete coverage goes well beyond the scope of this book; but a basic, high-level understanding of the tool is useful.
Note I could not hope to cover any more than the briefest details of Data Guard here. It is a book-length topic by itself. The latest book is entitled *Oracle Data Guard 11g Handbook* by Larry Carpenter, et. al. (Oracle Press 2009).

Suffice it to say that Oracle Data Guard Physical Standby allows the data center to be online all of the time. There is no down time. Naturally you still need backups for those pesky systematic errors (you deleted the wrong data). Data Guard makes your testing scenario much simpler. Now you need only carry out a switchover (a controlled switch of the computer roles as opposed to the failover), and once the switchover is complete you can start your testing immediately.

In 11g release 1, Oracle changed Data Guard so that not only was the physical standby ready for a failover or switchover operation, it was also available for read-only operations. Typically this would be to allow reports to run on the standby database. Many sites found that this was a massive boon to their operations. Now they could move their expensive reporting operations (in terms of resource usage) to the standby database and free up the primary for On-Line Transaction Processing (OLTP).

**SQLTXTRSBY**

Once reporting operations moved to the physical standby (and logical standby) we could once more have poorly performing SQL. Performance problems that are on a read/write database are the sort of thing that SQLT can help us with. The only problem being that SQLTXTRACT and SQLTXECUTE need read/write access to the database (to store data in the SQLT repository and to install packages and procedures). How is it possible for SQLT to help us if we can’t even store data about the performance on the database with the performance problem? This is where SQLTXTRSBY comes into play.

It provides some special procedures that deal with these special circumstances. As Data Guard (and other read-only databases) became more popular more of the performance problems mentioned above appeared on Data Guard instances. To deal with these problems SQLTXTRACT was adapted to work without any local storage. This was done by making special routines that ran from a read/write database and reached out across a database link to the read-only database, collected the required information and collated it and presented it on the read-write database. Let’s take a more detailed look at some of SQLTXTRACT’s limitations and then discuss using SQLTXTRSBY.

**SQLTXTRACT Limitations**

I always think of SQLTXTRACT (alias XTRACT) and SQLTXTRSBY (alias XTRSBY) as a superhero and his (or her) sidekick. XTRACT the superhero can seemingly do it all, but every superhero has their Achilles heel, in this case it is the inability to work on a read-only database. XTRSBY has those special skills that the main hero does not have. In this case XTRSBY can go into a read-only Data Guard system and get out with the information, where XTRACT cannot. Physical standbys cannot tolerate write operations. There cannot be two systems potentially updating the same data (unless you have multi-master replication of course, but that’s a whole other book), not unless there was some way for the primary to know what the failover system was doing, and Data Guard does not allow for that. This is why the physical standby database is open in read-only mode. If you do happen to wander onto a read-only database and try and insert some data you’ll get an error message like this:

```
SQL> insert into test1 values (1);
insert into test1 values (1)
* 
ERROR at line 1:
ORA-00604: error occurred at recursive SQL level 1
ORA-16000: database open for read-only access
```