Logical Replication in 12c: What are the Options Now?

Vit Spinka
Dbvisit Software Limited
Auckland, NZ

Keywords
Logical replication, Oracle 12c, Streams, GoldenGate, Advanced Replication, CDC, Dbvisit Replicate, Dell SharePlex

Introduction

Oracle 12*c* introduced several new features, most important from the point of logical replication being probably the multitenant architecture. Up until 11*g*, Oracle Database offered a couple of logical replication options, introduced over the years. With 12*c*, the time came to prune this offering. This comes both from technical reasons – they would have to be updated to work with 12*c* multitenant – as well as business reasons: The future is Oracle GoldenGate, and Oracle expects us to pay for this future.

Logical replication

The topic of this session is logical replication, not a physical one. Physical standby works still the same way as it did before; now it protects the whole CDB. Similarly, storage-based replications still work, too.
However, here we discuss logical replications – replicating just the values and their changes. This is a much more complex problem, but it makes the replication much more versatile.

As Oracle writes changes to redo records row by row (with some optimizations for direct load and for array-bind operations), the replication also needs to work row by row. It needs a logical identifier to uniquely distinguish the rows – this is usually a primary (or unique) key. Unlike physical replication, which essentially uses a rowid.
If things don't work as expected, a conflict can occur (when applying to a database), or data is simply not consistent (when just sending changes as events). A physical replication can also fail, aborting the recovery; but since it's much simpler operation, such event is much less likely to happen.

A logical replication can be trigger-based: such implementation is easy to program and does not need any internal knowledge. However, it's slow, as it intercepts each and every DML happening on the replicated table. It's also prone to issues with DDL. Note that if such approach is used internally by a database feature, there are some optimization available to the feature developers.

A more performant, and with much less impact to the source database, is reading the data directly from the redo logs. This can either use LogMiner, which parses redo log and presents it as a table of change records; or by directly parsing the redo logs. The latter has a lower impact on the database, can handle even cases not supported by LogMiner, and can keep a sub-second gap behind the source (which add up to 2-3 seconds when the network and apply are included).

For a relational database, the target is updated with single-line SQL. We can check that the values before an update/delete are same as expected, and that inserts honour primary keys. If not, a conflict occurs. This is generally good – we want to know if data is different from what it should be. This can happen by someone changing data at the target, by triggers modifying the data on target, cascading constraints, or simply by a product bug. Essentially they ensure that we can trust the data at the target, they are verified on every DML.

Or we can push them out as events – be it a messaging queue like JMS, special API like XStream, or a directly supported external system (Apache Kafka, Apache NiFi, HDFS, CSV). We move the data to the non-relational world and have to work with them as such.

Options obsolete in 12*c*

Oracle CDC stores the changes as records in a table in the local database, making the API a simple one (although limited for more complex datatypes). It used to be a popular choice among various event capturing replication tools or ETL tools, especially those for which Oracle was just one of many supported sources.
It can be synchronous, using internal triggers; or asynchronous, by reading redo. The asynchronous requires EE license, but no other option is necessary.
However, it was deprecated in 11*g* and it's completely removed from 12*c*.

Oracle Streams were introduced in 9*i*R2 and read the redo logs. It was the most used replicate option, although it has its issues with large transactions and the ability to recover from issues. It's free, included in Enterprise Edition license.
It is still available in 12*c*, but it's deprecated and does not work with CDB databases. Nor any other new 12*c* features are supported.
The Streams code is still alive, though – it's base for the Oracle GoldenGate integrated capture, described shortly.

Advanced Replication is based on updateable materialized views and materialized view logs. Its main strength was in low volume, disconnected environments. The data is refreshed based on schedule/manual requests.
However, 12*c* makes it obsolete, and it does not support PDBs. NB that we can still use basic replication, i.e. read-only materialized views, and they can be remote.

Oracle GoldenGate

Oracle acquired *GoldenGate* in 2009 and sees it as the one-and-all tool for logical replication. It's a powerful tool, but expensive, not easy to use, and overkill for many use cases.
It supports many sources database platforms, but nowadays most development is done for Oracle source. The old, classic capture reads redo logs directly from disk; the new, integrated one uses API calls and the parsing is done inside the Oracle database, using code based on Streams. It's clearly the integrated capture that is the future, and the one that receives updates and new functionality. For example, only the integrated one supports multitenant.
On the other hand, as it relies on code inside the database, it needs 11.2.0.3 (or .4) or higher: and some bug fixes need an update (patch) of the source database, too.
Similarly, there is an API for apply, too, and we get integrated replicat.
These APIs are called XStream In and XStream Out; however, use of either require full GoldenGate license, so it's not a way to code advanced data processing, not a way to get a cheap Streams replacement.
As mentioned above, integrated extract supports multitenant; one process can capture changes from any number of PDBs. Replicat uses one process per target PDB.
Note, however, that "multitenant support" does not imply that all use cases are supported; especially since 12.2 introduces more PDB operations hot cloning, relocation, or common objects.

Oracle GoldenGate for Big Data is an (independently licensed) assortment of connectors to push the changes into the non-relational world, like Flume, HDFS, Hive, Hbase or Kafka.

Oracle Data Guard: Logical Standby

Logical standby was introduced a long time ago (9iR2), but not many people use it. Oracle still sees a future for it, though.
It is part of Data Guard, thus included in EE license. It uses Data Guard to set it up and to transfer the data, but the actual capture and apply are more akin to Streams.
Logical Standby is updated to support multitenant, although, in 12.2, it does not support application containers.

Actually, Oracle sees it as a special purpose tool – for near-zero downtime migrations. The strength of Logical Standby is ease of instantiation and that it can easily handle replication of the whole database. And most important, it can actually convert back from logical to physical, in such case.
For these migration scenarios, even application containers are supported. And further development is made to automate such migrations for non-trivial standby setups, using DBMS_ROLLING package, orchestrating the upgrade in waves, always protecting the current primary by a physical standby.

Third party options

Oracle is not the only player in logical replication, there are other, 3rd party products, that can provide such functionality. Often, their strength is in ease of use and price; on the other hand, they come with more limitations then GoldenGate.

SharePlex was acquired by Dell (by purchasing Quest) in 2012; now it is owned by Dell Software, a spin-off owned by private equity company Francisco Partners. We have yet to see who will be the next owner.
It does support multitenant, and each PDB has its own configuration, capture and apply process. Only Oracle is supported as a source. Multiple targets are supported, including MS SQL Server and Apache Kafka.

Dbvisit Replicate also uses its own redo parser, supports multitenant, again each PDB having its own mine and apply process. Oracle is the only supported source, the target can be for example MS SQL Server, MySQL, or Kafka.

Contact Address:
Vit Spinka
Dbvisit Software Limited
Level 1, 506 Point Chevalier Road
Point Chevalier
Auckland 1022
New Zealand

Telefon:	+64 9 950 3301
Fax:	+64 9 950 3302
E-Mail:	vit.spinka@dbvisit.com
Internet:	www.dbvisit.com