Online v Offline Migration

This document details the different approaches – Online and Offline – to migrate PostgreSQL Databases from across sources such as on premise, native PostgreSQL servers and PostgreSQL servers from other cloud platforms.

Database migrations involve setting up a target server, migrating schema & data, handling of foreign keys and triggers, adding firewall rules at both source and target etc. In both Online and Offline migration, the migration service is a hosted solution where we deploy a VM on Azure and automatically set up all the infrastructure needed for the migration. Unlike offline, in case of Online migration, a process such as logical decodingmay be involved to support migration by replicating the changes occurring during the migration to the target server. The user can choose the most favorable migration method for his/her server based on the information in this document.

The differences are summarized in the table below:

	Online	**Offline**
Database availability for reads during migration	Available	Available
Database availability for writes during migration	Available	Generally, not recommended. Any 'writes' initiated after the migration is not captured or migrated
Application Suitability	Applications that need maximum uptime	Applications that can afford a planned downtime window
Environment Suitability	Production environments	Usually Development, Testing environments and some production that can afford downtime
Suitability for Write-heavy workloads	Suitable but expected to reduce the workload during migration	Not Applicable. Writes at source after migration begins are not replicated to target.
Manual Cutover	Required	Not required
Downtime required	Less	More
Logical replication limitations (See Section 2)	Applicable	Not Applicable
Migration time required (See Section 3)	Depends on Database size and the write activity until cutover	Depends on Database size
Technology differences		
Processes Involved	pg_dump , pg_restore and logical replication	pg_dump and pg_restore
Resource scaling	CPU, Memory and IOPS may have to be scaled at Source Server since resources are shared for Logical replication	Not required

Internal Processes Explained

a) pg_dump: Extracts a PostgreSQL database into a script file or other archive file. Description, Options and Examples available here.
b) pg_restore: Restores a PostgreSQL database from an archive file created by pg_dump. Description, Options and Examples available here.
c) Logical replication: Logical replication is a method of replicating data objects and their changes, based upon their replication identity (usually a primary key). More details here.

Logical Decoding Limitations (Applicable only for Online Migration)

PostgreSQL instances that use Logical decoding have the following restrictions or missing functionality that may require changes in the application code for workarounds.

- **The database schema and DDL commands are not replicated.** The initial schema can be copied by hand using `pg_dump --schema-only`. Subsequent schema changes would need to be kept in sync manually. (Note, however, that there is no need for the schemas to be absolutely the same on both sides.) Logical replication is robust when schema definitions change in a live database: When the schema is changed on the publisher and replicated data starts arriving at the subscriber but does not fit into the table schema, replication will error until the schema is updated. In many cases, intermittent errors can be avoided by applying additive schema changes to the subscriber first.
- **Sequence data is not replicated**. The data in serial or identity columns backed by sequences will of course be replicated as part of the table, but the sequence may be reset on the subscriber. If the subscriber is used as a read-only database, then this should typically not be a problem. If, however, some kind of switchover or failover to the subscriber database is intended, then care should be taken in the application to ensure that sequences are updated to the latest values, either by copying the current data from the publisher (perhaps using `pg_dump`) or by determining a sufficiently high value from the tables themselves.
- **Replication of TRUNCATE commands is supported**, **but some care must be taken when truncating groups of tables connected by foreign keys**. When replicating a truncate action, the subscriber will truncate the same group of tables that was truncated on the publisher, either explicitly specified or implicitly collected via `CASCADE`, minus tables that are not part of the subscription. This will work correctly if all affected tables are part of the same subscription. But if some tables to be truncated on the subscriber have **foreign-key** links to tables that are not part of the same (or any) subscription, then the applying truncate action on the subscriber may fail.
- **Large objects are not replicated**. There is no workaround for that, other than storing data in normal tables.
- Replication is only supported by tables, including partitioned tables. Attempts to replicate other types of relations, such as views, materialized views, or foreign tables, will result in an error.
- **When replicating between partitioned tables, the actual replication originates, by default, from the leaf partitions on the publisher**, so partitions on the publisher must also exist on the

subscriber as valid target tables. (They could either be leaf partitions themselves, or they could be further subpartitioned, or they could even be independent tables.) Publications can also specify that changes are to be replicated using the identity and schema of the partitioned root table instead of that of the individual leaf partitions in which the changes actually originate (see CREATE PUBLICATION).

Migration Timeline v Database Size (Applicable for Online and Offline)

The approximate times for the various database sizes below includes the time taken for the dump and restore of the source database into the target.

Database Size	Approximate Time Taken (HH:MM)
1 GB	00:01
5 GB	00:04
10 GB	00:08
50 GB	00:46
100 GB	06:11
500 GB	09:28
1000 GB	09:25