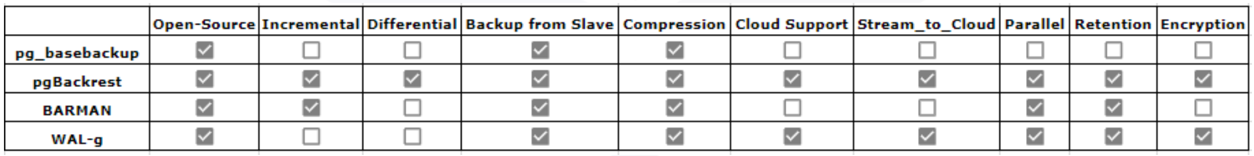
PostgreSQL Database Project & Support Services

Database Health Check

a) DB Services availability status check
b) DB error check: Postgres error log/Daily log analysis using pgbadger tool
c) Backup status Check (pg_basebackup/pg_dump/Barman/Pg_backrest)
d) DR synchronization check
e) Database Space check
f) OS Space check
g) Performance check
   - Invalid objects
   - Table locks
   - Invalid sessions
   - Postmaster health check status

Backup & Disaster Recovery Management

a) Maintain and managing the full and incremental backup of databases using:
   - pg_backrest (for large size of databases)
   - SQL Dump
   - File system backup
b) Re-Run failed backups
c) Ensure, all backups (data & WAL logs) are stored in safer remote location for restore consistency.
d) Table refresh, schema refresh, full db refresh using pg_dump,pg_restore.
e) Continuous Archiving and Point-in-Time Recovery (PITR)
f) Restoring user objects in case of any human error on end user data management.



	Open-Source	Incremental	Differential	Backup from Slave	Compression	Cloud Support	Stream_to_Cloud	Parallel	Retention	Encryption
pg_basebackup	☑	☐	☐	☑	☑	☐	☐	☐	☐	☐
pgBackrest	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑
BARMAN	☑	☑	☐	☑	☑	☐	☐	☑	☑	☐
WAL-g	☑	☐	☐	☑	☑	☑	☑	☑	☑	☑

Standby Database Management (Database Replication)

a) Warm Standby: A standby that cannot be connected to until it is promoted to a master server.
b) Hot Standby: A standby that can accept connections and serve read-only queries.
c) Failover: Switching to standby after an abnormal termination of the master server.
d) Logical replication: Table level replication to same version as well as different version of postgresql database.
e) Replication and Failover management for postgres using Repmgr tool
f) Re-sync Primary & Standby, if any gaps

- File or disk based
- Log shipping based (WAL)
- SQL based

Database User and Security Management

a) Restricted privileges/authentication using pg_hba.conf (mini firewall for your DB)
b) RLS – Row Level Security Policies
c) IP TABLES
d) postgressql.conf
e) Data Encryption
   - Whole-disk encryption
   - Per Column encryption
   - Pgcrypto: query level (unencrypted data and key can appear in log)
   - SSL Cofiguration
f) Restrict Access
   - Database to group
   - SCHEMA to group
   - TABLE to group
   - COLUMN to group

Database Capacity Planning & Space Management

a. Designing database storage requirement by considering immediate future growth.
b. Performing periodic capacity review (daily, Weekly, Monthly, Quarterly) based on customer requirement & send database growth reports regularly.
c. Add additional space by taking approval from management.
d. Pro-active housekeeping activities to clear old WAL logs, Backup files, Backup logs & database logs from OS drive, as per customer standard to avoid last minute space hiccups.
e. Archiving the wallog files to new destination and maintaining the space constraints.

Database Performance Tuning:

a) Frequent monitoring of running queries
b) Max connections monitoring and clean up idle connections.
c) Finding slow, long running & blocking queries and report to application team.
d) Index validity check
e) Finding blocking session/wait events
f) Checking Bloat Tables, Dead Tuples and analyze based on performance.
g) Maintenance & remove bloat with pg_repack and vaccum.
h) Log analysis using PGbadger tool.
i) EXPLAIN PLAN – Const analysis
j) Extended Statistics
k) checkpoint_segments temporarily during bulk data loads

l) Query re-write
m) Implementing Table level partitioning for better performance.
n) Tune parameters like:
  - wal segments,
  - checkpoints
  - shared_buffer
  - wal_buffer
  - work_mem(session level)
  - maintenance_work_mem
  - synchronous_commit
  - pg_hint_plan
o) Postgres OS level tunning:
  - hugepages
  - kernel parameters
  - vm.swappiness
  - vm.overcommit_memory
  - vm.overcommit_ratio
  - vm.dirty_backgroud_ratio
  - vm.dirty_background_bytes
  - vm.dirty_ratio
  - vm.dirty_bytes

Database Patch & Upgrade

a) Upgrade Postgres to any higher version.
b) **dbpatch** - PostgreSQL database patch change management extension. This extension supports conducting database changes and deploying them in a robust and automated way.

Upgrade method	Pro	Cons
Dump/restore	• Simple • Safe • Somewhat flexible	• Slowest method • Per database approach has some pitfalls lurking
Binary in-place	• Fast / very fast (depending on chosen mode) • Old instance not affected in default mode	• More complex than Dump / Restore • Somewhat risky in “link” mode • Possibly loses standby servers • Double the disk space required in default mode
Logical Replication	• Shortest possible downtime • Safe, with possibility of thorough “live tests” • Very flexible	• Most complex method • Possibly some schema changes needed • Not everything is transferred (sequence state, large objects) • Possibly “slowish” • Always per database

Database Design & Configuration.

a. Design OS requirements: Choosing & configuring the appropriate CPU, Memory, storage and ancillary software
b. Design Database requirements: Creating scalable database architectures that allows for expansion
c. Conducting design reviews
d. Designing database for high-speed, high-volume transactions
e. Installation and configuration of PostgreSQL database from source or Packages on all types Operating Systems.
f. Installation and configuration of PostgreSQL related tools like: pgBouncer, pgpool etc.
g. Preparing PosgreSQL documentation

Database High-Availability configuration & Management

a) EFM cluster management for primary and multiple standby
b) PgCluster
c) Pgpool-II
d) RubyRep
e) Bucardo
f) Postgres-XC
g) Citus
h) Postgres-XL

Database Migration:

a) Postgresql database migration from one host to another host.
b) Cloud Migration: Migrate postgres database from on premises to any cloud. Eg: AWS EC2 postgres.
c) Cross Technologies Migration: Migration from other database Technologies to PostgreSQL database. Eg: Oracle to postgresql, MariaDB to postgresql and vice-versa
d) Cross Platform Migration: Migrate Postgres DB from one OS to another OS.
Eg: centos to Ubuntu to reduce the cost of license for OS.
e) DBlink creation (FDW) from oracle to postgresql and postgresql to mysql, mysql to postgres.
f) Database Cloning

Postgres cloud DB support & Devops Support

a) Docker for postgres databases (Standalone, master and slave setup) with docker images.
b) Implementation and administration of RDS Posgres & EC2 postgres.
c) Implementation and administration of Aurora postgres with cluster.
d) DB parameter group changes in RDS postgres.

PostgreSQL Extensions

Modify/Extend the way that Postgres works

- File_fdw
- Dblink
- Postgres_fdw
- Pg_stat_statement
- Pg_trgm
- Hstore
- Postgis
- Postgis_topology
- TimescaleDB
- Pg_cron
- Pg_metrics
- Pg_repack
- pgBadger
- pgAudit
- pg_Promethus

Support using various PostgreSQL Tools

Monitoring

- **PoWA** - PoWA (PostgreSQL Workload Analyzer) is a PostgreSQL workload analysis tool
- **PgCluu** - pgCluu is a PostgreSQL performance monitoring and auditing tool.
- **Pgwatch2** - It is based on Grafana and provides monitoring functions for the PostgreSQL DB.
- **PgAudit** – provides detailed session and/or object audit logging via standard postgresql logging facility

Logic and trigger-based replication tools

- **pgLogical -** It is a logical replication tool implemented in the form of PostgreSQL extension plug-ins.

Multi-master replication tool

- BDR - Bi-Directional Replication for PostgreSQL

High availability and failover tools

- Repmgr - Repmgr is an open-source tool for PostgreSQL server cluster replication and failover
- PAF - PostgreSQL Automatic Failover-Automatic Failover Tool
- Patroni - Template that uses Python for highly available solution for maximum usability
- Stolon - Stolon is a cloud-native PostgreSQL high-availability management tool

Connection Pooling Tools

- PgBouncer - PgBouncer allows client access to PostgreSQL database operations greater than the maximum number of connections it can provide
- PgPool-II - The functions it can provide include query-based replication, connection pool function, load balancing, parallel query, etc

<u>**Table partitioning tool**</u>

- Pg_Partman – It is an extension of PostgreSQL, used to create and manage time-based or sequence-based table partitions
- pg_Pathman - Optimized partition solutions for large distributed databases

<u>**Migration tool**</u>

- Ora2pg - Tool for migrating Oracle or MySQL databases to PostgreSQL
- pgloader - loads data into PostgreSQL and enables Continuous Migration from your existing database to PostgreSQL. It can load data from files like CSV or Fixed-File Format or convert an entire database to PostgreSQL

<u>**Scheduling Tool**</u>

- Pg_cron using for cronjob internally in postgres.