Mastering PostgreSQL

Advanced SQL

- DCL (Data Control Language): GRANT, REVOKE, and ALTER ROLE commands for managing user permissions.
- Advanced JOINs: Explore advanced join types like nested-loop joins, merge joins, and hash joins to optimize complex queries involving multiple tables.
- Window Functions: Utilize window functions like RANK, PERCENTILE_CONT, and ROW_NUMBER to perform aggregations and calculations within result sets.
- Subqueries: Master subqueries, including Common Table Expressions (CTEs), to create more readable and efficient queries that can reference data from multiple tables.
- Table Inheritance: Explore table inheritance for creating hierarchical relationships between tables and code reusability.
- Triggers: Implement triggers to automate database operations in response to INSERT, UPDATE, or DELETE events.
- GiST Indexes: Understand Generalized Search Trees (GiST) indexes for efficient full-text search capabilities.
- JSON Functions: Leverage built-in JSON functions to work with JSON data stored within PostgreSQL tables.

Performance Optimization and Indexing

- EXPLAIN Analyze: Dive deep into query performance analysis using EXPLAIN Analyze to understand how PostgreSQL executes your queries and identify bottlenecks.
- Indexing Strategies: Learn about different indexing strategies like B-Tree, hash, and GIN indexes to create appropriate indexes that significantly improve query speed.
- Materialized Views: Explore materialized views, a technique for pre-computing frequently used queries and storing the results for faster retrieval.
- Vacuuming and Autovacuum: Understand the importance of vacuuming and autovacuum for efficient storage management by reclaiming unused space and preventing database bloat.
- Query Caching and Cost-Based Optimization: Understand query caching mechanisms and cost-based optimization techniques to further enhance query performance.
- pg_stat_statements: Utilize pg_stat_statements to analyze historical query execution statistics and identify optimization opportunities.

Advanced Database Administration

- DCL (Data Control Language): GRANT, REVOKE, and ALTER ROLE commands for managing user permissions.
- Advanced JOINs: Explore advanced join types like nested-loop joins, merge joins, and hash joins to optimize complex queries involving multiple tables.
- Window Functions: Utilize window functions like RANK, PERCENTILE_CONT, and ROW_NUMBER to perform aggregations and calculations within result sets.
- Subqueries: Master subqueries, including Common Table Expressions (CTEs), to create more readable and efficient queries that can reference data from multiple tables.
- Table Inheritance: Explore table inheritance for creating hierarchical relationships between tables and code reusability.
- Triggers: Implement triggers to automate database operations in response to INSERT, UPDATE, or DELETE events.
- GiST Indexes: Understand Generalized Search Trees (GiST) indexes for efficient full-text search capabilities.
- JSON Functions: Leverage built-in JSON functions to work with JSON data stored within PostgreSQL tables.

Scaling and High Availability

- Horizontal Scaling: Learn about horizontal scaling techniques like sharding with pgpool or pglogical to distribute data across multiple servers and handle increasing workloads.
- Vertical Scaling: Explore vertical scaling methods by adding more CPUs or memory to your existing PostgreSQL server to improve its processing power and capacity.
- Replication: Understand the concepts of synchronous, asynchronous, and high availability replication to ensure data redundancy and maintain continuous database operations.
- Streaming Replication: Understand streaming replication for real-time data replication between PostgreSQL servers, enabling high availability.
- Failover Techniques: Explore various failover techniques to ensure minimal downtime in case of server outages.
- Disaster Recovery Planning: Create a comprehensive disaster recovery plan to effectively respond to data loss or system failures.

- Percona Toolkit: Explore Percona Toolkit, a collection of open-source command-line tools for managing and optimizing MySQL performance, which can also be valuable for PostgreSQL.
- pt-querydigest: Analyze slow query logs generated by PostgreSQL to identify inefficient queries that require optimization.
- pt-explain: Gain insights into query execution plans using pt-explain to understand how PostgreSQL retrieves data and identify potential bottlenecks.
- Slow Query Log Analysis: Configure PostgreSQL's slow query log to capture queries exceeding a specific threshold and analyze them using Percona Toolkit or other open-source tools to pinpoint slow-running queries.
- EXPLAIN Analyze Visualization Tools: Utilize graphical tools that visualize the output of EXPLAIN Analyze, making it easier to comprehend query execution plans and identify optimization opportunities.
- Consider exploring other open-source alternatives like (pg_query_analyze) or pgAdmin for query performance analysis and optimization.