AWS Redshift

Low cost Analytics with SQL

Agenda

What is AWS Redshift

How it compares to a ‘normal’ Relational Database in RDS

Analytics with Redshift

Getting started yourself

What is AWS Redshift

Transactional SQL database

Fork from PostgreSQL version 8

Data stored in columns, compressed – not rows

Resizable Cluster

- dense storage or dense compute

Similarities with a RDS

Optional encryption connection / at rest

Access via IAM and security groups + Database users

Snapshots

User defined functions

Target for AWS Database Migration Service

Differences from a RDS

Clustered

No Multi AZ

No Read Replicas

Read Access during resize

User define functions*

Table restore

Firehose Target

Copy from

- S3
- EMR
- Remote Host (SSH)
- DynamoDB

Unload to S3

Redshift vs RDS costs

Compute $0.25/$0.30 close to r3.large

Storage $0.85/$0.95 close'ish to m4.2xlarge

Analytics with Redshift

Redshift Tables

No indexes

Column Compression Type

- Set on create table or by Copy command into empty table
- ANALYZE COMPRESSION to get Redshift recommendation

Distribution – Styles (Even | Key | All) and DISTKEY <single column>

Sort keys – Compound or interlaced

Primary keys and foreign keys

Creating a Redshift Cluster

Cluster Identifier

Database name

Port

Admin user & password

Node type

Cluster Type Single/Multi node

Node count

Redshift Parameter Group

Encrypt Database None/KMS/HSM

Target VPC

Allocate public ip if required

Cluster Subnet Group

Availability Zone

VPC security group

Add CloudWatch Alarm

IAM role(s)

Getting started

Configure Client:

- SQL Workbench/J
- Jdbc driver
- Check autocommit! Or make sure you commit or rollback in every batch

Loading data

PostgreSQL style copy to load data

- parallelised
- various options for source file layout and compression
- by default sets row compression encoding
- S3, EMR, Remote Host (SSH), DynamoDB

Load from AWS resources – IAM Roles or IAM Access key

copy dates from 's3://<bucket name>/dates.tsv‘ with CREDENTIALS 'aws_iam_role=arn:aws:iam::123456789123:role/redshiftdemo‘ delimiter '\t‘ gzip;

Unloading data

Unload in a similar style to the copy command

- parallelised
- various options for source file layout and compression (not lzop oddly)
- S3 Only using IAM Roles or IAM Access key

unload ('select * from dates where id < 10000') to 's3://<bucket name>/dates.tsv' with CREDENTIALS 'aws_iam_role=arn:aws:iam::123456789123:role/redshiftdemo' delimiter '\t' gzip;

Be Aware

Disk full

- Larger queries filling cluster
- Copy from – COMPUPDATE ON

Don’t select *

Try to change data / data model

- to limit number of where clause criteria
- use aggregate functions

Questions? & Links

docs.aws.amazon.com/redshift/latest/mgmt/welcome.html

docs.aws.amazon.com/redshift/latest/gsg/getting-started.html

docs.aws.amazon.com/redshift/latest/gsg/rs-gsg-prereq.html#rs-gsg-prereq-sql-client

Email: peter@catalystcomputing.co.uk

Web: catalystcomputing.co.uk

Blog: catalystcomputing.co.uk/peter-marriott

Twitter: @peter_marriott

GitHub: github.com/catalystcomputing