Amazon Neptune: Graph Data Management in the Cloud

Brad Bebee, Daniel Choi, Ankit Gupta, Andi Gutmans, Ankesh Khandelwal,
Yigit Kiran, Sainath Mallidi, Bruce McGaughy, Mike Personick, Karthik Rajan,
Simone Rondelli, Alexander Ryazanov, Michael Schmidt, Kunal Sengupta,
Bryan Thompson, Divij Vaidya, and Shawn Wang

Amazon Web Services, Seattle, WA 98101, USA

Abstract. We present Amazon Neptune, a fast, reliable, and fully managed graph
database service. Supporting both the Apache Tinkerpop Gremlin stack as well
as the RDF 1.1 / SPARQL 1.1 W3C standards, Amazon Neptune efficiently
stores and navigates highly connected data, allowing developers to create inter-
active graph applications that can query billions of relationships with millisecond
latency. As a purpose-built cloud service, Neptune is seamlessly embedded into
the Amazon Web Service (AWS) ecosystem and comes with a broad set of en-
terprise features including SDKs for deployment and configuration, high availa-
bility and scale-up using replication, automatic backup and restore functionality,
point in time recovery, monitoring, encryption-at-rest, security using VPCs and
integrated access management, as well as audit logs. In our presentation, we will
sketch customer use cases, put them into context with Neptune architecture and
features, and discuss options for engagement with the Semantic Web community.

Keywords: RDF, SPARQL, Property Graphs, Gremlin, Databases, Cloud

1   Amazon Neptune

In discussions with our customers, we have been seeing an increasing need for graph
database use cases including social networks, recommendation engines, knowledge
graphs, fraud detection, network and IT operations, and life sciences. *Pearson*, for in-
stance, helps more than 75 million people each year embrace their passion and ambition
for learning new skills. They have been evaluating Neptune to reduce complexity in
several of their key systems and to standardize how the organization approaches heavily
connected, graph data. *Blackfynn* is a life science software startup formed to change
how Epilepsy, Alzheimer's disease, Parkinson's disease, ALS, and other neurological
diseases are treated. They look forward to using Neptune to "connect the dots between
genomics, pathology, neurochemistry, device and patient clinical data, efficiently and
at scale" and drive breakthrough discoveries. *Intuit* is powering prosperity for millions
of small businesses and self-employed customers around the world; they are exploring
Neptune for heavy lifting of their graph infrastructure management.

Addressing the needs of our customer's use cases, we have designed Amazon Nep-
tune with a focus on performance, ease of use, reliability, and standards compliance.

Performance: Neptune allows users to build interactive graph applications that can query billions of relationships with millisecond latency. It is optimized for in-memory data processing and can be deployed on different instance types, backed by a durable storage service layer that scales to 100B+ triples. Query processing is fully ACID with immediate consistency. A dedicated bulk load API enables efficient load from S3 buckets in different (possibly compressed) input formats (NTriples, NQuads, RDF/XML, and Turtle for RDF; CSV for Property Graphs). In order to scale query throughput and provide high availability, Neptune allows to dynamically spin up up to 15 read replicas.

Ease of use: As a fully managed cloud service, Neptune can be deployed and scaled within minutes, thus significantly easing graph data management operations. Software upgrades are deployed automatically during configurable maintenance windows, and deployments are monitored permanently so as to initiate automatic failovers when problems are detected. Deployments can be triggered and maintained through a Web based console or using cloud SDKs (such as AWS CLI or CloudFormation, allowing to automate Neptune deployments and related infrastructure setup). To provide a high level of security, Neptune is deployed into VPCs and offers authentication via SigV4 signing[1] (SigV4 samples for Gremlin and SPARQL clients are available in Open Source[2]). Orthogonally, encryption-at-rest is supported using the AWS Key Management Service.

Reliability: AWS uses the concepts of regions, where each region is a separate geographic area and has multiple, isolated locations known as Availability Zones (AZs). Neptune maintains 6 copies of the data spread across 3 AZs and, in the rare case of a failure (such as a hardware outage), will automatically attempt to recover the database in a healthy AZ without data loss. In addition to automated restore functionality, the user can take and restore snapshots of the database using the console or SDKs. A variety of operational metrics are available to the database owner through AWS CloudWatch and can be used to define custom alerts and automate infrastructure management.

Standards Compliance: Neptune is fully compliant with the Apache Tinkerpop Gremlin stack and implements the W3C RDF 1.1 / SPARQL 1.1 standards. This allows our customers to leverage the broad ecosystem of Open Source and commercial tooling built around the Gremlin and SPARQL stacks. To provide an example, on the SPARQL side existing client libraries such as Apache ARQ/Jena or RDF4J[3] can be used out-of-the-box to interact with Neptune. Complementarily to Open Source projects, the AWS Partner Network (APN) provides value-added services such as graph management, visualization, exploration, search, and application building on top of Neptune.

In our presentation, we will sketch use cases and examples of how our customers are using Neptune, relate them to Neptune's architecture and features, and seek to start a dialog with the Semantic Web community, to establish partnership with research institutions and further the goal of providing an easy to use scalable graph database service. At AWS, we are customer-obsessed and work backwards from customer needs. We would love to hear how the community is using Neptune and what it would like to see in a fully managed graph service with support for RDF/SPARQL 1.1 and beyond.

---

[1] https://docs.aws.amazon.com/general/latest/gr/signature-version-4.html
[2] See e.g. https://github.com/aws/amazon-neptune-sparql-java-sigv4
[3] See https://jena.apache.org/ and http://rdf4j.org/