Line-up of AWS Query in Place, Databases, and Analytics Services

	Query in Place, Schema-on-Read			Transactional Processing, Reporting and Analytics, Schema-on-Write
	S3 Select	**Amazon Athena**	**Amazon Redshift Spectrum**	**Amazon RDS**
Database Type	S3 feature, enables applications to retrieve only a subset of data from an object by using simple SQL expressions	Interactive query-in-place service that allows to analyse data directly in S3 using standard SQL	Feature of Amazon Redshift that allows SQL queries to reference data in S3 without loading or ETL pipelines.	**Relational DBMS**, supports the following engines: MySQL, MariaDB, PostgreSQL, Microsoft SQL Server, Aurora
Workloads	Query-in-place using a subset of SQL Serverless applications (e.g. Lambda) accessing data stored on S3 Big data frameworks accessing objects stored on S3	Interactive ad hoc queries against data directly in S3 without the need to setup or manage any servers	Direct query on PB/EB scale of S3 data in Apache Parquet, ORC, JSON, and Amazon Ion file formats ANSI SQL compliant OLAP	Primarily for OLTP workloads ERP, CRM, e-commerce Choose RDS whey your workload requires a low administrative burden and is relatively small **Edge cases when to prefer RDS over Aurora**: 1. RDS supports two MySQL engines (InnoDB and MyISAM) 2. You need to use the **latest** version of MySQL or PostgreSQL as RDS is ahead of Aurora in terms of supported engine versions 3. You can select storage type for RDS instance (e.g. you can use magnetic HDD for RDS) - cost optimisation
Data Type	CSV, JSON, Apache Parquet GZIP, GZIP2 (for CSV and JSON only)	Unstructured, semi-structured, and structured	Structured and Semi-Structured, Relational	Structured and Relational
Consistency Model	Not applicable	Not applicable	Same as Redshift	ACID
Architecture	Not applicable	Uses Presto with full standard SQL support	Used the Redshift Spectrum sever fleet independent of your cluster and fully managed by Amazon. Processing occurs in this Amazon Redshift Spectrum layer - offloads query activity from the main cluster. Final aggregations and joins with local Redshift tables are completed in-cluster. Employs MMP to process the large datasets in S3	Host-based traditional database model where Amazon manage the whole database lifecycle and maintenance.
Storage	S3	S3	S3	EBS-volumes attached to the DB instance nodes: choose between two SSD-backed options: - provisioned IOPS - general-purpose The traditional magnetic HDD storage options are also available.
Compute	Not applicable	Internal Amazon computation layer	Own Redshift Spectrum server cluster layer	You select DB instance classes - available classes are optimised for memory, performance or I/O (T, M or R families)
Data Size Limits	S3 console limits the amount of data returned to **40MB** (use AWS CLI or API) to retrieve more data Maximum length of a record in the input or output is **1MB**	Run queries at EB scale as it uses S3 as data source	Run queries at EB scale as it uses S3 as data source	Max **16TB** for SQL Server and **64TB** for all other engines, option for RDS storage auto scaling
Limits	You must have S3:GetObject permission Maximum length of SQL expression is **256KB** Can only emit nested data using the JSON output With Parquet objects: - only columnar compression GZIP/Snappy - no Parquet output (only CSV or JSON) - max uncompressed row group size is 256MB	Athena Federated Query feature is available in preview only in some regions	You **need an Redshift cluster** and a SQL client that's connected to your cluster to execute queries. The cluster and data files in S3 must be in the same AWS region. Requires AWS Glue data catalog for schema reference	Soft limit of 40 instances (depending on database engine)
Connectivity	VPC Gateway Endpoint	AWS Management Console JDBC/ODBC connection Athena API or CLI You can connect directly to Athena using VPC interface endpoint without traversing public internet.	Create external schema Create external tables Query external tables	Primary CNAME endpoint Read replica CNAME endpoints
Security	S3 security applies	You can run queries on encrypted data in S3 in the same region (supported encryption options are SSE-S3 and SSE-KMS). You can encrypt query results in S3 and the data in AWS Glue Data Catalog. To encrypt data in transit, Athena uses TLS (default)	Access to data is governed by IAM roles for Redshift	VPC isolation, your DB instance can be public or private IAM integration Secrets Management to store the database credentials Data encryption at rest (optional) with AWS KMS and in transit (SSL) Supports Transparent Data Encryption (TDE) for SQL Server.
High Availability and Durability	Not applicable	Athena is highly available and runs queries using compute resources across multiple AZs, automatically routing queries if particular AZ is unreachable. Uses S3 as underlying data store.	You can spin up multiple clusters across AZs and access data in S3 without having to load it into the cluster.	**Multi-AZ Deployment:** With Multi-AZ deployment RDS provisions and maintains a **synchronous** standby replica in a different. RDS automatically fails over to the **standby** in case of planned or unplanned outage of the primary - usually under 2 minutes. Physical replication for MySQL, MariaDB, and PostgreSQL. Logical replication for SQL Server (native mirroring technology). The standby instance is **not accessible** by the application. Failover automatically flips CNAME pointing to the new primary instance, no need to any application changes. **Read Replicas:** You can have up to 5 **asynchronous** read replicas per primary instance in multiple AZs and **cross-region.** Each read replica has a separate CNAME for read-only clients. You can also manually promote a replica to the primary instance (manual fail-over).
Cross-Region Support (out of the box)	S3	No cross-region support	No cross-region support	Up to 5 **asynchronous** replicas **cross-region**: **Read Only**
Scalability	Not applicable	There is no infrastructure to manage. Best practices to improve performance and scalability: - Partition data - Use compression - Optimise file sizes (avoid files under 128MB) - Use columnar file formats - Query tuning - Use approximate functions	With Redshift Spectrum you can run multiple Redshift clusters accessing the same data in S3 and use different clusters for different use cases	Storage auto scaling Scaling out by adding read replicas (you can have read replicas of read replicas) Scaling up by selecting larger DB instances
Backup and Restore	Not applicable	Not applicable	Not applicable - as all data is external	**Automated backups** (system snapshot and transaction log), retention period 0-35 days **Manual snapshots** You can copy snapshot cross-region, share between accounts, and export to S3 **PITR:** up to the last 5 minutes **Database native backup** options: can be used to meet regulatory and disaster recovery requirements. No RDS interference with these operations. Your backup storage is equivalent to the sum of the database storage for all instances in that Region. **AWS Backup**: you can also use AWS Backup to manage backups of RDS DB instances. AWS Backup provides a centralised console offering backup scheduling, retention management, and backup monitoring. All restores create a new DB instance
Maintenance	Not applicable	Serverless, fully managed by Amazon	Fully managed by Amazon	Updates are applied to RDS DB instances during system maintenance windows. The update is first applied to secondary instance in case of Multi-AZ deployment. Shared management responsibility between Amazon (Scaling, HA, Backups, DB software patching, OS, Server, Hardware) and you (App optimisation)
Monitoring	S3	CloudTrail to monitor API calls and user actions CloudWatch events Use workgroups to separate users, teams, applications, or workloads and to set query limits and control query costs	You can monitor Redshift Spectrum queries using the Redshift system views (SVL_S3QUERY, SVL_S3QUERY_SUMMARY)	Database log files - you can publish all logs to CloudWatch RDS Events RDS Recommendations Standard and Enhanced Monitoring Performance Insights CloudTrail to audit any actions taken by a user, role or AWS service on your RDS database
Billing	- GB of data scanned per GB - GB of data returned per GB - Select requests	**- Data scanned**: amount of S3 data scanned to execute your query (per TB) **- AWS Glue Data Catalog usage** charge (per request)	**- Data scanned**: amount of S3 data scanned to execute your query (per TB). No charge if you are not running queries **- Data Storage**: standard S3 rates Best practices: - Partition external data - Use columnar file format - Compress data	**- DB instance-hours** based on the class. Multi-AZ deployments charged doubled (as they use standby instances) **- Database Storage** (per GB-month) - provisioned to DB instance **- Backup Storage** - no charge up to 100% of your total database storage for a region **- I/O consumed** (requests) - for magnetic storage and Aurora only **- Provisioned IOPS** - for provisioned IOPS only, regardless of IOPS consumed **- Snapshot export** to S3 **- Data transfer** - charges for cross-AZ traffic and data transferred OUT, data transferred between RDS and EC2 instances in the same AZ is free
Special Features and Considerations	Query pushdown using S3 Select is supported by Spark, Hive, Press and EMR. Amazon Athena, Redshift and EMR all support S3 Select. **Glacier Select:** allows to perform filtering directly against a Glacier object using standard SQL statements.	**AWS Glue Data Catalog** to reference Meta Data for schema-on-read queries Integrates with **Amazon QuickSight** for data visualisation **Athena Federated Query**: to query data in sources other than S3. You can query data in place or build pipelines that extract data from multiple data sources and store them in S3 **Integration with other AWS Services**: you can query data from the following AWS services in Athena: - CloudTrail Logs - CloudFront Logs - ALB/CLB Logs - VPC Flow Logs - CloudFormation Integration with: - Glue Data Catalog - QuickSight - IAM **Workgroups**: used to separate users, teams, applications, or workloads and to set limits on amount of data each query or workgroup can process.	Can be used together with **AWS Lake Formation**: - use Lake Formation as centralised place for permission and polices management - create external tables and run queries on data in the lake	You can migrate (copy) data to Amazon Aurora MySQL or PostgreSQL from RDS MySQL or PostgreSQL: - Migrating an RDS snapshot to Aurora - Migrating to Aurora DB Cluster using Aurora Read Replica **AWS Database Migration Service**: migrate or replicate existing databased to RDS **Performance Insights**: database performance tuning and monitoring feature that helps you quickly assess the load on your database and determine when and where to take action.
Top-Level Documentation Link	https://docs.aws.amazon.com/AmazonS3/latest/dev/selecting-content-from-objects.html	https://docs.aws.amazon.com/athena/index.html	https://docs.aws.amazon.com/redshift/latest/dg/c-getting-started-using-spectrum.html	https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Welcome.html
FAQ Link	https://aws.amazon.com/s3/faqs/	https://aws.amazon.com/athena/faqs/	https://aws.amazon.com/redshift/faqs/	https://aws.amazon.com/rds/faqs/

	Transactional Processing, Reporting and Analytics, Schema-on-Write		NoSQL	
	Amazon Aurora	**Amazon Redshift**	**Amazon DynamoDB**	**Amazon DocumentDB**
Database Type	High-end fully managed **relational** database engine compatible with MySQL and PostgreSQL.	Petabyte-scale fully managed cloud cluster Data warehouse with columnar data storage supporting Massively Parallel Processing (MPP)	Non-relational scalable **key-value** store with milliseconds latency and high throughput	Store **documents** and quickly access querying on any attribute
Workloads	OLTP or light analytics and reporting workloads (however consider the max data size of 64TB). Aurora supports only MySQL InnoDB and PostgreSQL engines. Choose Aurora when your workload has any of the following requirements: high concurrency, high volume, low latency, and advanced replication.	Managed data warehouse solution using standard SQL Primarily for reporting and analytics (OLAP) workloads	Non-transactional workloads, shopping cart, customer preferences, gaming, e-commerce, ultra-high scale applications, transactional (OLTP)	Content management, catalogs, user profiles. Supports MongoDB workloads (implements Apache 2.0 MongoDB 3.6 API) - can be used to emulate MongoDB server and to use existing MongoDB drivers and tools.
Data Type	Structured and Relational	Structured and Semi-Structured, Relational	Semistructured Tables contain Items and each item - any number of attributes Keys Indexes	Data is stored in JSON-like documents
Consistency Model	ACID Reads from the primary - always strongly consistent Reads from any replica - eventually consistent	Shared the same consistency model as S3 when data is copied or accessed in S3 All regions provide read-after-write consistency for uploads of new objects with unique object keys. Cross-region COPY operations are eventually consistent.	BASE Eventually consistent reads (default) Strongly consistent reads (they are not HA however) ACID transactions: serialisable, can contain up to 25 operators, use up to 4MB of data	BASE Reads from the primary - always strongly consistent Reads from any replica - eventually consistent
Architecture	Built on SSD-backed purpose-built distributed storage system shared between all cluster nodes within one AWS region. All data is distributed in three different AZs across multiple storage 10GB-nodes. The Aurora MySQL- and PostgreSQL-compatible database engines are customised to take advantage of the fast distributed storage.	Cluster with one leader leader node and multiple compute nodes with attached SSD-based storage. The leader node manages communications with clients and with all compute nodes. It also creates execution plan and distributes the pre-compiled code to the compute nodes. A compute node is partitioned into slices.	Fully serverless (from client point of view) Data is stored in partitions (max 10GB per partition) based on partition key (hash of partition key) - all items for this partition key are kept together. Items in the partition are sorted by the sort key. DynamoDB automatically shards data across partitions.	Similar to Aurora architecture. Distributed storage volume (cluster volume) - distributed fault-tolerant, self-healing system. The primary write/read instance and all replicas share and access the same cluster volume. Decouples compute and storage.
Storage	Aurora uses single virtual SSD-backed cluster volume accessible from all nodes of the Aurora cluster. A cluster volume consists of copies of the data across all AZs in a single AWS Region. The storage scales automatically from 10GB up to 64TB in 10GB increments with no impact to database performance. Replica lag is usually less than 100 milliseconds	Storage types depends on node type used: - RA3 node type with up to **64TB**/node - scalable **managed storage**, uses local SSD as tier-1 cache and S3 for longer-term durable storage - DC2 node type with **160GB** or **2.6TB**/node - fixed local SSD - legacy DS2 node type with **2TB** or **16TB**/node - fixed local HDD	SSD-backed distributed storage Data is replicated in three different locations in three different AZs within a single region	SSD-backed distributed storage
Compute	You can select DB instance class (t- and r- instances are available) DB instances are not data bearing	RA3 node type with 12vCPU/node or 48vCPU/node DC2 node type with 2vCPU/node or 32vCPU/node Can contain 1-128 nodes (128 nodes with ra3.16xlarge)	Managed by Amazon, no access and no setup by the client	Different types of DB instances (varies by the region). r5 and t3 are widely supported DB instances are not data bearing
Data Size Limits	Up to **64TB** total storage size, auto growth **64TB** maximum table size	No hard limits, PB scale, e.g. **8.2PB** with 128x ra3.16xlarge nodes	Item size up to **400KB** (including all keys and attributes)	Up 64TB of data per cluster, scales automatically Individual document can be up to **16MB**
Limits		Max 128 nodes in cluster by default (adjustable) Max 9900 or 20000 tables (depends on node type) Max 60 databases per cluster	Default quota (adjustable) of 40000 read or write request per second per table or 80000 read/writes per second per account Max 256 tables (adjustable) per account per region Hard limit of 5 LSI per table Max 20 GSI (adjustable) per table Max length of a partition key is 2048 bytes Max length of a sort key is 1024 bytes	
Connectivity	- Cluster read/write endpoint - Reader endpoint - read-only connection for all replicas - Instance endpoint for each individual node - Custom endpoints for any combination of nodes	JDBC/ODBC connection using industry standard drivers Example endpoint: redshift-cluster-1.ccdmuz4og8ln.us-east-1.redshift.amazonaws.com:5439/dev Example connection string: jdbc:redshift://redshift-cluster-1.ccdmuz4og8ln.us-east-1.redshift.amazonaws.com:5439/dev	DynamoDB is a public Amazon service You can use VPC Gateway Endpoint to enable EC2 instances to access DynamoDB without traversing the public internet (no public IP required)	- Cluster endpoint read/write endpoint (automatically redirects to a new primary instance in case of fail-over) - Reader endpoint - read-only connections across all available replicas, provides also **connection** load balancing - Instance endpoint - direct connections to individual instances
Security	By default, Aurora is created in a VPC with disabled network access and you can control access to the database in usual manner (using security groups, NACLs and subnets). Aurora uses SSL (AES-256) to secure data in transit and has optional data at rest encryption using keys managed through KMS.	Encryption (optional) using AES-256 SSL encryption in transit AWS KMS or CloudHSM VPC security groups Identity federation for SSO and MFA Column-level access control	Managed service is protected by the AWS infrastructure security procedures User management via IAM Data is encrypted at rest by default using KMS keys Encryption in transit for all connections Fine-grained identity and access control	VPC isolation IAM resource-level permissions Encryption at rest (AES-256), AWS KMS Encryption in transit TLS by default
High Availability and Durability	Aurora DB cluster is fault tolerant by design. The cluster volume spans multiple AZs in a single AWS Region. Aurora can tolerate loss of two copies without affecting write availability and up to three copies without affecting reads. Aurora storage is continuously scanned for errors and repaired automatically. Using the shared cluster volume storage reduces RTO to less than 60 seconds in most cases. If the primary instance in DB cluster fails, Aurora **automatically** fails over to a new primary instance in one of the two ways (same region only): 1. By promoting an existing Aurora **Replica** to the new primary instance (you can assign failover target priority to your existing replicas) - RTO is normally under 120 seconds 2. By creating a new primary instance (if the DB cluster doesn't contain any replicas) - RTO is normally under 10 min Aurora implements two types of read replication: - **Logical** - know as cross-region replicas, supported for both MySQL and PostgreSQL - **Physical** - know as Aurora Global, supported only by MySQL The following replicas can be created and used for HA and Disaster Recovery: - Up to **15 in-region** Aurora Replicas - act as failover target with no data loss and with **auto** failover - Up to **5 cross-region** MySQL Replicas (based on MySQL binlog replication with latency > 1 second) - act as failover target with potentially minutes of data loss and **no auto** failover - You can use **Aurora Global** to replicate via AWS backbone to up to 5 secondary regions with latency of under a second (physical replication). You can promote a secondary region to primary in under minute. Cross-region failover is a manual process. Aurora Global supports writing to the secondary instances (as of recent). You can add up to 16 read replicas in each secondary region. Aurora automatically updates CNAME to point at the healthy replica (for same region failover)	**Single-AZ** cluster Data is replicated within the cluster (each drive's data is mirrored to other nodes) Backups are stored on S3 (eleven nines of durability) Automated provision of a node and rebuild of a drive from replicas in case of failure. The most frequently queried data is restored first. The cluster will be unavailable for queries and updates until a replacement is provisioned and added to the cluster. Single-node clusters do not support replication. In case of AZ failure you can restore the cluster from the backup in a different AZ - this will create a new cluster with a new endpoint. You can run several clusters in multiple AZs by loading data into your clusters from the same set of Amazon S3 input files. You can restore a cluster in a **new region** from the cross-region snapshot (cross-region snapshots should be enabled)	DynamoDB is highly available within AWS region. Data is replicated in three different locations in three different AZs within a single region. Multi-region active-active replication is possible with **global tables**. Global tables are write/read region-local replicas of the original table, replicated by DynamoDB streams. When an application writes data to a replica table in one region, DynamoDB propagates the write to the other replica tables in other regions automatically. "Last-writer-wins" reconciliation applied to resolve concurrent updates.	Designed for 99.99% availability and replicates 6 copies of data across 3 AZs. Continuous monitoring of database instance health and auto failover to read replica - typically less than 30 seconds. Supports up to 15 replicas in the same AWS region. Replicas supports only read operations. Recovery process: - Auto detects node or AZ failure - Primary role moves to read replica - Resume writing to new primary - Node is replaced in background Fails to replica with the lowest numbered priority tier HA recommendations: - Set failover tier for a specialised high-traffic instance to high number - Utilise replica set with read preference of secondary preferred
Cross-Region Support (out of the box)	Aurora Global: **Read/Write**	No replication cross-region (only restore from a snapshot) Can do COPY operations where the cluster and the bucket are in different regions.	DynamoDB Global Tables: **Read/Write**	No cross-region support No replication cross-region You cannot share snapshots cross-region
Scalability	Horizontal scaling (for reads) by creating up to 15 read replicas for a DB cluster that uses single-master replication. Vertical scaling by choosing larger DB instances times (write and computation scaling) You can define a policy for Replica Autoscaling based on target metric, minimum and maximum capacity. Aurora Serverless autoscaling	AQUA (Advanced Query Accelerator) Vertical - increasing node size Horizontal - increase number of nodes - downtime from minutes (elastic resize) to hours (classic resize) With the Concurrency Scaling Redshift automatically adds additional cluster capacity to process an increase in concurrent read queries. The cluster remains fully available during concurrency scaling	Linear write/read scalability Provides three model for throughput: - on demand (you pay per request) - provisioned write and read capacity in WCU and RCU - **auto scaling** of write and read capacity based on demand Adaptive capacity per partition is available by default (automatically increases throughput capacity for partitions that receive more traffic) **DynamoDB Accelerator (DAX)**: write-through in-memory caching, microsecond read response at scale. The DAX cluster consists of up to 10 nodes: Item cache and Query cache.	Scale out read capacity on demand: adding new read replicas (takes minutes regardless of data size) Scale up read capacity on demand: adding a larger read replica instance (instance sizes do not need to match), available within minutes because no data replication is required Scale up write capacity on demand: you can add a large instance as read replica and promote it to primary
Backup and Restore	Automated continuous and incremental backups are always enabled and they **do not impact** database performance (retention 1-35 days) Point-in-time recovery with the retention period is automatically enabled (5 min granularity) Manual snapshots Aurora automatically maintains 6 copies of the data across 3 AZs and automatically recovers the database in a healthy AZ with no data loss You can copy the snapshots **cross-region** and share **cross-account or publicly** or **export to S3** Restore from a DB snapshot or point-in-time restore creates a new Aurora instance Aurora mySQL supports "rewinding" a DB cluster to specific time (without restoring data from a backup and creating a new Aurora instance) - **Backtrack**. Backtrack must be enabled during new cluster creation and provides a up to 72 hours backtrack window. Aurora also supports "**database cloning**" (up to 15 clones from the same source) which uses copy-on-write protocol. The clone is attached to the original cluster and stores only the data changes and no additional storage required at the point of clone creation.	Automated snapshot with retention 1-35 days **Cross-region** snapshots are possible (asynchronous replication). For cross-region replication of the encrypted snapshots you have to create a new KMS key in a new region. Manual snapshots with up to 3653 days retention Can configure **cross-account** access to snapshots A new cluster is created from "restore from snapshot" Redshift maintains three copies of data (original, replica on the compute nodes, and a backup on S3). Free backup storage = total size of storage on the nodes	Automated full backups of all data with no performance impact to your table PITR within the retention period All restores create a new table	**Automated snapshots** (1-35 days) - daily continuous incremental updates **Manual snapshots** - on demand full backup, retained indefinitely **PIRT** - enabled by continuous backup with incremental updates after daily automatic snapshot. Any second within retention period (last 5 min transactions are unavailable) All restores create a new cluster
Maintenance	Same as RDS Updates require a database restart - 20 to 30 seconds of downtime	You cannot access the cluster nodes directly Redshift periodically performs maintenance to apply fixes, enhancements and new features to your cluster (during defined maintenance windows)	Fully managed by Amazon	Fully managed by Amazon
Monitoring	Same as RDS	CloudWatch metrics on compute utilisation, storage, read/write traffic	CloudWatch DynamoDB Dashboard DAX Cluster Metrics	CloudWatch
Billing	**- DB instance-hours** consumed **- Database Storage**: "high-water mark" principle **- I/O consumed** (requests) **- Backup storage** (no charge for up to 100% of your total database storage for each DB cluster) **- Backtrack**: hourly rate for storing Change Records **- Snapshots:** Export to S3 **- Data Transfer**: charges for cross-AZ traffic and data transferred OUT **- Aurora Serverless** - you pay for database storage, database capacity in ACU (Aurora Capacity Units) and I/O the database consumes	**- Compute node hours**: total across all compute nodes. Only for running cluster. **- Managed storage**: fixed GB-months rate regardless of data size, you pay based on total data present in the managed storage **- Backup storage**: no charge for up to 100% of cluster provisioned storage (S3 standard rates for backup storage beyond the provisioned storage size and backup stored after the cluster termination) **- Data Transfer:** only cross-region (standard AWS data transfer rates)	**On-demand** capacity mode: number of write/read requests to the table **Provisioned** capacity mode: provisioned WCU/RCU For both modes: **- Data storage** over 25GB per month **- Backup** (continuous and on-demand) and restore per GB of data **- Global tables replication -** per request or rWCU **- DynamoDB Streams** - write/read requests **- DAX** - per instance-hour **- Data transfer OUT** - per GB	DocumentDB priced in four dimensions: **- On-demand instances** **- Database I/O** used when reading/writing data to the cluster storage volume **- Database storage** per GB/months **- Backup storage** in excess of cluster's storage usage
Special Features and Considerations	**Aurora Global** - allows a single Aurora database to span up to 5 secondary AWS regions. It replicates data with no impact on database performance with typical latency under a second, and provides disaster recovery from region-wide outages (no automatic failover cross-region). Each region has own read-only endpoint for region-local readers. **Aurora Multi-Master** - adds the ability to scale out write performance across multiple AZs and have minimal downtime fail-over. Currently limited to 2 read/write nodes. **Aurora Serverless** - on-demand, autoscaling configuration that automatically starts up, shuts down, scales capacity up or down based on the application's need. **Aurora Parallel Query** - ability to push down and distribute the computational load of a single query across all of CPUs in Aurora's storage layer nodes. Aurora can invoke Lambda events from stored procedures and triggers and run scripts	**Data lake integration**: Query data in the data lake in place without loading or ETL pipelines **Query Editor** in AWS Management Console Support for stored procedures using the PostgreSQL (can incorporate DDL, DML) **Redshift Advisor**: recommendations for data model, schema and data loading; guides optimisation of cluster operations and settings **Automatic tuning**: table sort, vacuum and analyse operations, workload management, efficient query plans **Workload Management (WLM):** helps manage workloads and enable short, fast-running queries to avoid getting stuck in queues behind long-running queries. Optional Resize Schedule and Pause and Resume Schedules You can configure usage limits: concurrency scaling usage limit and Redshift Spectrum usage limit All storage sizes are for compressed data (uncompressed data sizes are 3-4x larger)	**DynamoDB Streams:** captures a time-ordered sequence of item-level modifications (insert, update, delete) in a table, retention is 24 hours. **Global tables**: build high-performance, globally distributed active-active applications. **DynamoDB Accelerator (DAX)** **Local Secondary Index** (LSI) - up to 5 LSI per table, must be defined during table creation, shares the RCU/WCU with the table **Global Secondary Index (GSI)** - up to 20 GSI per table, can be defined any time, does not share the RCU/WCU with the table Time to live (TTL) can be set on per-item basis Partition key design is extremely important to avoid problems with "hot partitions" and performance	**Change streams** - a time-ordered sequence of change events that occur within your cluster's collections (disabled by default). Can be only opened from a connection to the primary instance (change stream from a replica is not supported). Events stored for 3 hours by default (1-24 hours). Applications can use change streams to subscribe to all data changes on a database or on individual collections.
Top-Level Documentation Link	https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/CHAP_AuroraOverview.html	https://docs.aws.amazon.com/redshift/	https://docs.aws.amazon.com/dynamodb/index.html	https://docs.aws.amazon.com/documentdb/index.html
FAQ Link	https://aws.amazon.com/rds/aurora/faqs/	https://aws.amazon.com/redshift/faqs/	https://aws.amazon.com/dynamodb/faqs/	https://aws.amazon.com/documentdb/faqs/

	NoSQL			
	Amazon ElastiCache	**Amazon Neptune**	**Amazon QLDB**	**Amazon Timestream (preview)**
Database Type	**In-memory** data store and cache that provides scalability to maintain sub-millisecond response times with fluctuating application demand Query by key with microsecond latency Fully managed implementation of **Redis** and **Memcached**	**Graph** - navigate relationships between data entities	Fully managed **ledger** database that provides a transparent, immutable, and **cryptographically verifiable transaction log** owned by a central trusted authority	Fully managed **time-series** database purpose built for time-series data
Workloads	Caching, session management, gaming leaderboards, geospatial applications Requirement for very low latency (sub-millisecond) Requirement for very high throughput (100000+ operations per second) Running a self-managed in-memory data store Requirement for sticky sessions Access patterns include repeat requests	Fraud detection, social networking, recommendation engine Graph, highly connected data, transactional	System-of-record, supply chain, registrations, banking transactions. Requirement for a cryptographically verifiable database.	IoT, DevOps, telemetry, App Monitoring
Data Type	Semistructured and unstructured	Graph structured	Ledger structured Semi-structured and nested Amazon Ion as data structure (based on JSON) PartiQL as query language	Time-series data for IoT and operational applications
Consistency Model	Depends on caching strategy, generally eventually consistent	ACID compliant	Sequentially recorded ACID Serialisable Immutable strong consistency	*Waiting for GA release*
Architecture	ElastiCache sits in your application as a separate tier. Application should implement a suitable cache strategy (lazy loading, write-through, TTL) Use of consistent hashing approach for key distribution is recommended Redis: Primary node, up to 5 replicas, Multi-AZ failover and sharding Memcached: Cluster uses nodes for data partitioning. Each node has its own read/write endpoint and the cluster has the configuration endpoint. Client application controls which hashing algorithm is used for partitioning in multi-node cluster.	Fully managed graph database service Optimised for processing graph queries Consist of cloud-native storage service (cluster volume), Neptune high-performance graph engine (Durable, ACID with immediate consistency) and two graph query engines: **Gremlin** and **SPARQL**	Serverless **Not** a blockchain or distributed ledger technology. Centralised ownership (system-of-record owned by a single centralised entity), transactions can execute without the need for multi-party consensus. The journal is the database	Serverless, auto scaling
Storage	Network attached RAM storage (on the cluster nodes)	SSD-backed cluster volume. Data grouped in 10GB segments. Replicated 6 times over 3 AZs. Automatically grows up to 64TB Need 4 of 6 copies for consistent writes Need 3 of 6 copies for consistent reads In case of failure, peer-to-peer replication is used to recover any lost copies of the data	The types of data storage in QLDB: - **Journal storage:** the disk space that is used bye a ledger's journal. The journal is append-only and contains the complete, immutable, and verifiable history of all the changes to the data - **Indexed storage**: the disk space that is used by a ledger's table, indexes, and indexed history. Indexed storage consist of ledger data that is optimised for high-performance queries	Stores data on in-memory, SSD, or magnetic store.
Compute	Supports nodes of M, R, and T classes	Different types of DB instances (varies by the region) r5 and r4 are widely supported		*Waiting for GA release*
Data Size Limits	No hard limit on cache item size	Up to **64TB** of data per cluster, scales automatically	Max document size is **128KB** Max transaction size is **4MB**	*Waiting for GA release*
Limits	Memcached: Soft limit of 20 nodes per cluster Redis: Soft limit of 90 nodes per cluster	No access from outside the VPC Total size of Gremlin and SPARQL HTTP requests must be less than 150MB	Soft quotas: - Max 5 active ledgers in a Region- - Max 2 active journal exports to S3 per ledger - Max 5 active journal streams to Kinesis per ledger Hard quotas: - Max 20 active tables - Max 40 total tables - Max 2 indexes per table	*Waiting for GA release*
Connectivity	Redis: **Node endpoint** for standalone node, **primary endpoint** and **reader endpoint** for cluster (cluster mode disabled), and **configuration endpoint** for cluster with cluster mode enabled Memcached: **Node endpoints** (individual nodes) or **configuration endpoint** (cluster). All memcached endpoints are read/write	Currently there is no access from outside the VPC - Cluster endpoint with failover support: read/write endpoint - Reader endpoint (round-robin routing for read-only connections to the DB cluster): **connection** load balancing - Instance endpoint (connects to specific DB instance)	Integrated with AWS PrivateLink (VPC Interface Endpoint) Need the AWS-provided QLDB driver	Write API available for Java, Python, Golang, Node.js and .NET, AWS CLI Adapters and plugins: Kinesis Data Analytics (Apache Flink), AWS Lambda, Telegraf, AWS IoT Core Query API: ANSI SQL, coming soon: QuickSight, Grafana, JDBC
Security	VPC isolation Redis: encryption at rest (optional) and in transit (TLS). Supports AUTH for client authentication. Memcached: no encryption IAM integration	VPC isolation IAM for endpoint access HTTPS-encrypted client connections Encryption at rest using AWS KMS	Amazon QLDB traffic stays within the AWS network by using private interface VPC endpoint (AWS PrivateLink) IAM Integration Encryption at rest (using AWS KMS) and in transit (SSL) by default. Does not support CMKs.	*Waiting for GA release*
High Availability and Durability	Redis and Memcached: - You can chose an AZ for a node - Automatically repairs by provisioning a new node instead of failed and ensures that DNS and IP address of the node remain the same. Redis: Automatic Multi-AZ failover. Up to 5 read replicas in multiple AZs with asynchronous replication. Replicas **will have an impact** on the performance of the primary node. Memcached: no replication, each node has own dataset	Designed to provide 99.99% availability. Fault-tolerant and self-healing storage with 6 copies of the data across 3 AZs. Continuous backup to S3 and automated recover from physical store failures. Instance failover typically less than 30 seconds. Up to 15 low-latency read replicas across 3 AZs, all replicas point to shared cluster volume. Reader endpoints balance across **connections to replicas** (DNS resolves each time to a new replica instance in round-robin fashion).	HA by default QLDB **journal storage** performs synchronous replication to multiple AZs on transaction commits. Additionally, the QLDB journal features asynchronous achieves to fault-tolerant storage. QLDB **indexed storage** is backed by replication to multiple AZs Full recovery from AZ failures. A write is acknowledged only after being written to a durable storage in multiple AZ and strongly durable.	*Waiting for GA release*
Cross-Region Support (out of the box)	Redis: can copy snapshots cross-region No replication cross-region	No cross-region support No replication cross-region You cannot share snapshots cross-region	Standard AWS VPC cross-region support (e.g. VPC peering), no specific features for QLDB Export to S3 can be replicated cross-region (S3 bucket replication)	*Waiting for GA release*
Scalability	Redis: ElastiCache scale out, in, and up to meet fluctuating application demands. Write and memory scaling are supported with sharding. You can scale out network by using network-optimised instances. Redis: Up to 5 read replicas in multiple AZ provide read scaling and HA. Memcached: scale horizontally by adding or removing nodes.	Scales up to 100000 graph queries per second using read replicas	QLDB automatically scales to support the demand of your application	Capable of processing trillions of events daily
Backup and Restore	Redis: Backup cluster metadata and all data to S3. You can restore a cluster from the backup/snapshot. Cross-region and cross-account copy of the snapshots is supported. Memcached: No backups/snapshots/restore	Continuous incremental backup to S3, daily, max retention 35 days Manual snapshots (can be shared with other AWS accounts) PITR with 1-second granularity, logs are streamed to S3 All restores create a new cluster	**On-demand journal export** (S3): can be used for data retention, analytics, and auditing. S3 bucket can be replicated for cross-region replication. QLDB **does not provide** a dedicated backup and related restore feature at this time. No PITR. **Journal streams** provides a continuous journal stream capability.	*Waiting for GA release*
Maintenance	Fully managed by Amazon Redis also allows certain online operations: scaling the cluster, upgrading the Redis engine version, applying patches and maintenance updates	Managed by Amazon within Neptune Maintenance Window	Fully managed by Amazon	*Waiting for GA release*
Monitoring	CloudWatch SNS notifications on important cluster events	CloudWatch Database event notifications (via SNS) CloudTrail	QLDB Dashboard CloudWatch	*Waiting for GA release*
Billing	Same pricing for Redis and Memcached options **- Cluster node hours**: billed per Node-hour consumed for each node type. **- Data Transfer**: No charge for data transfer between EC2 and ElastiCache within the same AZ. Standard EC2 regional data transfer charges apply, only for data transfer in or out of the EC2 instance		**- Database storage** for journal and indexed storage per GB-month **- I/O -** number of write/read requests **- Data transfer**: Only data OUT from QLDB to Internet. No charge for data transferred between QLDB and other AWS services within the same region. Data transferred across AWS Regions is charged on both sides of the transfer).	- Writes number - Memory/SSD/Magnetic Store per GB-hour - Query per TB of data scanned - Data transfer in and out: no additional charge for data transferred between Timestream and other AWS services within the same region. Data transferred between regions is charged on both sides of the transfer
Special Features and Considerations	Memcached and Redis each have their own design considerations. Use **Memcached** when: - Simple caching model - Multi-threading on large nodes with multiple cores Use **Redis** when: - Advanced data types - Sorting/Ranking in-memory, e.g. leaderboards - Pub/sub - Persistence and failover - Encryption and compliance standards - Geospatial support - Cluster mode with online resharding **Auto Discovery with Memcached**: automatic identification of nodes in the cache cluster for cluster scaling and node replacement Redis provides a richer set of functionality than Memcached: complex data structures, persistence, failover, atomic operations, pub-sub messaging Redis can work in three configurations: - Single Node (no replication, no sharding) - Cluster Mode Disabled (replication, single shard) - Cluster Mode Enabled (replication, up to 90 shards)	Supports load of data from S3 via S3 VPC endpoint by using **Neptune Loader.** **Amazon Neptune Workbench**: console experience to query your graph. Integrated with Amazon SageMaker Notebooks (query works with Gremlin and SPARQL) **Neptune Streams**: log every change to your graph in real-time and strictly sequentially. Log entries are stored in the same native storage as graph data. Can also be used for Neptune-to-Neptune replication setup.	You can export journal content to S3 **Streams**: you can create a stream in QLDB that captures every document revision that is committed to the journal and delivers data to Kinesis Data Streams in near-real time. QLDB Streams provide **at-least-once delivery** and **no ordering** guarantee. **Query editor** is available in AWS console. **PartiQL**: a new open standard query language that supports SQL-compatible access to QLDB's document-oriented data model that includes semi-structured and nested data.	Built-in analytics Write API and adapters Query API and plugins
Top-Level Documentation Link	https://docs.aws.amazon.com/elasticache/index.html	https://docs.aws.amazon.com/neptune/index.html	https://docs.aws.amazon.com/qldb/index.html	https://aws.amazon.com/timestream/
FAQ Link	https://aws.amazon.com/elasticache/faqs/	https://aws.amazon.com/neptune/faqs/	https://aws.amazon.com/qldb/faqs/	Not available