Package ‘paws.database’

March 14, 2025

Title 'Amazon Web Services' Database Services

Version 0.9.0

Description Interface to 'Amazon Web Services' database services, including 'Relational Database Service' ('RDS'), 'DynamoDB' 'NoSQL' database, and more <https://aws.amazon.com/>.

License Apache License (>= 2.0)

URL https://github.com/paws-r/paws, https://paws-r.r-universe.dev/paws.database

BugReports https://github.com/paws-r/paws/issues

Imports paws.common (>= 0.8.0)

Suggests testthat

Encoding UTF-8

RoxygenNote 7.3.2

Collate 'dax_service.R' 'dax_interfaces.R' 'dax_operations.R' 'docdb_service.R' 'docdb_interfaces.R' 'docdb_operations.R' 'docdbelastic_service.R' 'docdbelastic_interfaces.R' 'docdbelastic_operations.R' 'dynamodb_service.R' 'dynamodb_interfaces.R' 'dynamodb_operations.R' 'dynamodbstreams_service.R' 'dynamodbstreams_interfaces.R' 'dynamodbstreams_operations.R' 'elasticache_service.R' 'elasticache_interfaces.R' 'elasticache_operations.R' 'keyspaces_service.R' 'keyspaces_interfaces.R' 'keyspaces_operations.R' 'lakeformation_service.R' 'lakeformation_interfaces.R' 'lakeformation_operations.R' 'memorydb_service.R' 'memorydb_interfaces.R' 'memorydb_operations.R' 'neptune_service.R' 'neptune_interfaces.R' 'neptune_operations.R' 'neptunedata_service.R' 'neptunedata_interfaces.R' 'neptunedata_operations.R' 'qldb_service.R' 'qldb_interfaces.R' 'qldb_operations.R' 'qldbsession_service.R' 'qldbsession_interfaces.R' 'qldbsession_operations.R' 'rds_service.R' 'rds_operations.R' 'rds_custom.R'

'rds_interfaces.R' 'rdsdataservice_service.R'
'rdsdataservice_interfaces.R' 'rdsdataservice_operations.R'
'redshift_service.R' 'redshift_interfaces.R'
'redshift_operations.R' 'redshiftdataapiservice_service.R'
'redshiftdataapiservice_interfaces.R'
'redshiftdataapiservice_operations.R'
'redshiftserverless_service.R'
'redshiftserverless_interfaces.R'
'redshiftserverless_operations.R' 'reexports_paws.common.R'
'simpledb_service.R' 'simpledb_interfaces.R'
'simpledb_operations.R' 'timestreamquery_service.R'
'timestreamquery_interfaces.R' 'timestreamquery_operations.R'
'timestreamwrite_service.R' 'timestreamwrite_interfaces.R'
'timestreamwrite_operations.R'

NeedsCompilation no

Author David Kretch [aut],
Adam Banker [aut],
Dyfan Jones [cre],
Amazon.com, Inc. [cph]

Maintainer Dyfan Jones <dyfan.r.jones@gmail.com>

Repository CRAN

Date/Publication 2025-03-14 14:40:02 UTC

Contents

`dax` *Amazon DynamoDB Accelerator (DAX)*

Description

DAX is a managed caching service engineered for Amazon DynamoDB. DAX dramatically speeds up database reads by caching frequently-accessed data from DynamoDB, so applications can access that data with sub-millisecond latency. You can create a DAX cluster easily, using the AWS Management Console. With a few simple modifications to your code, your application can begin taking advantage of the DAX cluster and realize significant improvements in read performance.

Usage

```
dax(config = list(), credentials = list(), endpoint = NULL, region = NULL)
```

Arguments

`config` Optional configuration of credentials, endpoint, and/or region.

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy `https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e` `html`

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.

- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

Value

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- dax(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

create_cluster	Creates a DAX cluster
create_parameter_group	Creates a new parameter group
create_subnet_group	Creates a new subnet group

decrease_replication_factor	Removes one or more nodes from a DAX cluster
delete_cluster	Deletes a previously provisioned DAX cluster
delete_parameter_group	Deletes the specified parameter group
delete_subnet_group	Deletes a subnet group
describe_clusters	Returns information about all provisioned DAX clusters if no cluster identifier is specified, or a
describe_default_parameters	Returns the default system parameter information for the DAX caching software
describe_events	Returns events related to DAX clusters and parameter groups
describe_parameter_groups	Returns a list of parameter group descriptions
describe_parameters	Returns the detailed parameter list for a particular parameter group
describe_subnet_groups	Returns a list of subnet group descriptions
increase_replication_factor	Adds one or more nodes to a DAX cluster
list_tags	List all of the tags for a DAX cluster
reboot_node	Reboots a single node of a DAX cluster
tag_resource	Associates a set of tags with a DAX resource
untag_resource	Removes the association of tags from a DAX resource
update_cluster	Modifies the settings for a DAX cluster
update_parameter_group	Modifies the parameters of a parameter group
update_subnet_group	Modifies an existing subnet group

Examples

```
## Not run:
svc <- dax()
svc$create_cluster(
  Foo = 123
)

## End(Not run)
```

docdb *Amazon DocumentDB with MongoDB compatibility*

Description

Amazon DocumentDB is a fast, reliable, and fully managed database service. Amazon DocumentDB makes it easy to set up, operate, and scale MongoDB-compatible databases in the cloud. With Amazon DocumentDB, you can run the same application code and use the same drivers and tools that you use with MongoDB.

Usage

```
docdb(config = list(), credentials = list(), endpoint = NULL, region = NULL)
```

Arguments

`config` Optional configuration of credentials, endpoint, and/or region.

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

Value

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- docdb(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

add_source_identifier_to_subscription	Adds a source identifier to an existing event notification subscription
add_tags_to_resource	Adds metadata tags to an Amazon DocumentDB resource
apply_pending_maintenance_action	Applies a pending maintenance action to a resource (for example, to an Amazo
copy_db_cluster_parameter_group	Copies the specified cluster parameter group
copy_db_cluster_snapshot	Copies a snapshot of a cluster
create_db_cluster	Creates a new Amazon DocumentDB cluster
create_db_cluster_parameter_group	Creates a new cluster parameter group
create_db_cluster_snapshot	Creates a snapshot of a cluster
create_db_instance	Creates a new instance
create_db_subnet_group	Creates a new subnet group
create_event_subscription	Creates an Amazon DocumentDB event notification subscription
create_global_cluster	Creates an Amazon DocumentDB global cluster that can span multiple multipl
delete_db_cluster	Deletes a previously provisioned cluster
delete_db_cluster_parameter_group	Deletes a specified cluster parameter group
delete_db_cluster_snapshot	Deletes a cluster snapshot
delete_db_instance	Deletes a previously provisioned instance
delete_db_subnet_group	Deletes a subnet group
delete_event_subscription	Deletes an Amazon DocumentDB event notification subscription
delete_global_cluster	Deletes a global cluster
describe_certificates	Returns a list of certificate authority (CA) certificates provided by Amazon Do

describe_db_cluster_parameter_groups	Returns a list of DBClusterParameterGroup descriptions
describe_db_cluster_parameters	Returns the detailed parameter list for a particular cluster parameter group
describe_db_clusters	Returns information about provisioned Amazon DocumentDB clusters
describe_db_cluster_snapshot_attributes	Returns a list of cluster snapshot attribute names and values for a manual DB c
describe_db_cluster_snapshots	Returns information about cluster snapshots
describe_db_engine_versions	Returns a list of the available engines
describe_db_instances	Returns information about provisioned Amazon DocumentDB instances
describe_db_subnet_groups	Returns a list of DBSubnetGroup descriptions
describe_engine_default_cluster_parameters	Returns the default engine and system parameter information for the cluster da
describe_event_categories	Displays a list of categories for all event source types, or, if specified, for a spe
describe_events	Returns events related to instances, security groups, snapshots, and DB parame
describe_event_subscriptions	Lists all the subscription descriptions for a customer account
describe_global_clusters	Returns information about Amazon DocumentDB global clusters
describe_orderable_db_instance_options	Returns a list of orderable instance options for the specified engine
describe_pending_maintenance_actions	Returns a list of resources (for example, instances) that have at least one pendi
failover_db_cluster	Forces a failover for a cluster
failover_global_cluster	Promotes the specified secondary DB cluster to be the primary DB cluster in t
list_tags_for_resource	Lists all tags on an Amazon DocumentDB resource
modify_db_cluster	Modifies a setting for an Amazon DocumentDB cluster
modify_db_cluster_parameter_group	Modifies the parameters of a cluster parameter group
modify_db_cluster_snapshot_attribute	Adds an attribute and values to, or removes an attribute and values from, a ma
modify_db_instance	Modifies settings for an instance
modify_db_subnet_group	Modifies an existing subnet group
modify_event_subscription	Modifies an existing Amazon DocumentDB event notification subscription
modify_global_cluster	Modify a setting for an Amazon DocumentDB global cluster
reboot_db_instance	You might need to reboot your instance, usually for maintenance reasons
remove_from_global_cluster	Detaches an Amazon DocumentDB secondary cluster from a global cluster
remove_source_identifier_from_subscription	Removes a source identifier from an existing Amazon DocumentDB event not
remove_tags_from_resource	Removes metadata tags from an Amazon DocumentDB resource
reset_db_cluster_parameter_group	Modifies the parameters of a cluster parameter group to the default value
restore_db_cluster_from_snapshot	Creates a new cluster from a snapshot or cluster snapshot
restore_db_cluster_to_point_in_time	Restores a cluster to an arbitrary point in time
start_db_cluster	Restarts the stopped cluster that is specified by DBClusterIdentifier
stop_db_cluster	Stops the running cluster that is specified by DBClusterIdentifier
switchover_global_cluster	Switches over the specified secondary Amazon DocumentDB cluster to be the

Examples

```
## Not run:
svc <- docdb()
svc$add_source_identifier_to_subscription(
  Foo = 123
)

## End(Not run)
```

docdbelastic *Amazon DocumentDB Elastic Clusters*

Description

Amazon DocumentDB elastic clusters

Amazon DocumentDB elastic-clusters support workloads with millions of reads/writes per second and petabytes of storage capacity. Amazon DocumentDB elastic clusters also simplify how developers interact with Amazon DocumentDB elastic-clusters by eliminating the need to choose, manage or upgrade instances.

Amazon DocumentDB elastic-clusters were created to:

- provide a solution for customers looking for a database that provides virtually limitless scale with rich query capabilities and MongoDB API compatibility.
- give customers higher connection limits, and to reduce downtime from patching.
- continue investing in a cloud-native, elastic, and class leading architecture for JSON workloads.

Usage

```
docdbelastic(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

`config` Optional configuration of credentials, endpoint, and/or region.

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.

- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

Value

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- docdbelastic(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

apply_pending_maintenance_action	The type of pending maintenance action to be applied to the resource
copy_cluster_snapshot	Copies a snapshot of an elastic cluster
create_cluster	Creates a new Amazon DocumentDB elastic cluster and returns its cluster structure
create_cluster_snapshot	Creates a snapshot of an elastic cluster
delete_cluster	Delete an elastic cluster
delete_cluster_snapshot	Delete an elastic cluster snapshot
get_cluster	Returns information about a specific elastic cluster
get_cluster_snapshot	Returns information about a specific elastic cluster snapshot
get_pending_maintenance_action	Retrieves all maintenance actions that are pending
list_clusters	Returns information about provisioned Amazon DocumentDB elastic clusters
list_cluster_snapshots	Returns information about snapshots for a specified elastic cluster
list_pending_maintenance_actions	Retrieves a list of all maintenance actions that are pending
list_tags_for_resource	Lists all tags on a elastic cluster resource
restore_cluster_from_snapshot	Restores an elastic cluster from a snapshot
start_cluster	Restarts the stopped elastic cluster that is specified by clusterARN
stop_cluster	Stops the running elastic cluster that is specified by clusterArn
tag_resource	Adds metadata tags to an elastic cluster resource
untag_resource	Removes metadata tags from an elastic cluster resource
update_cluster	Modifies an elastic cluster

Examples

```
## Not run:
svc <- docdbelastic()
svc$apply_pending_maintenance_action(
  Foo = 123
)

## End(Not run)
```

dynamodb *Amazon DynamoDB*

Description

Amazon DynamoDB is a fully managed NoSQL database service that provides fast and predictable performance with seamless scalability. DynamoDB lets you offload the administrative burdens of operating and scaling a distributed database, so that you don't have to worry about hardware provisioning, setup and configuration, replication, software patching, or cluster scaling.

With DynamoDB, you can create database tables that can store and retrieve any amount of data, and serve any level of request traffic. You can scale up or scale down your tables' throughput capacity without downtime or performance degradation, and use the Amazon Web Services Management Console to monitor resource utilization and performance metrics.

DynamoDB automatically spreads the data and traffic for your tables over a sufficient number of servers to handle your throughput and storage requirements, while maintaining consistent and fast performance. All of your data is stored on solid state disks (SSDs) and automatically replicated across multiple Availability Zones in an Amazon Web Services Region, providing built-in high availability and data durability.

Usage

```
dynamodb(config = list(), credentials = list(), endpoint = NULL, region = NULL)
```

Arguments

`config` Optional configuration of credentials, endpoint, and/or region.

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token

- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

Value

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- dynamodb(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

batch_execute_statement	This operation allows you to perform batch reads or writes on data stored in Dynam
batch_get_item	The BatchGetItem operation returns the attributes of one or more items from one or
batch_write_item	The BatchWriteItem operation puts or deletes multiple items in one or more tables
create_backup	Creates a backup for an existing table
create_global_table	Creates a global table from an existing table
create_table	The CreateTable operation adds a new table to your account
delete_backup	Deletes an existing backup of a table
delete_item	Deletes a single item in a table by primary key
delete_resource_policy	Deletes the resource-based policy attached to the resource, which can be a table or s
delete_table	The DeleteTable operation deletes a table and all of its items
describe_backup	Describes an existing backup of a table
describe_continuous_backups	Checks the status of continuous backups and point in time recovery on the specified
describe_contributor_insights	Returns information about contributor insights for a given table or global secondary
describe_endpoints	Returns the regional endpoint information
describe_export	Describes an existing table export
describe_global_table	Returns information about the specified global table
describe_global_table_settings	Describes Region-specific settings for a global table
describe_import	Represents the properties of the import
describe_kinesis_streaming_destination	Returns information about the status of Kinesis streaming
describe_limits	Returns the current provisioned-capacity quotas for your Amazon Web Services acc
describe_table	Returns information about the table, including the current status of the table, when i
describe_table_replica_auto_scaling	Describes auto scaling settings across replicas of the global table at once
describe_time_to_live	Gives a description of the Time to Live (TTL) status on the specified table
disable_kinesis_streaming_destination	Stops replication from the DynamoDB table to the Kinesis data stream
enable_kinesis_streaming_destination	Starts table data replication to the specified Kinesis data stream at a timestamp chos
execute_statement	This operation allows you to perform reads and singleton writes on data stored in D
execute_transaction	This operation allows you to perform transactional reads or writes on data stored in
export_table_to_point_in_time	Exports table data to an S3 bucket
get_item	The GetItem operation returns a set of attributes for the item with the given primary
get_resource_policy	Returns the resource-based policy document attached to the resource, which can be
import_table	Imports table data from an S3 bucket
list_backups	List DynamoDB backups that are associated with an Amazon Web Services accoun
list_contributor_insights	Returns a list of ContributorInsightsSummary for a table and all its global secondar
list_exports	Lists completed exports within the past 90 days
list_global_tables	Lists all global tables that have a replica in the specified Region
list_imports	Lists completed imports within the past 90 days
list_tables	Returns an array of table names associated with the current account and endpoint
list_tags_of_resource	List all tags on an Amazon DynamoDB resource
put_item	Creates a new item, or replaces an old item with a new item
put_resource_policy	Attaches a resource-based policy document to the resource, which can be a table or
query	You must provide the name of the partition key attribute and a single value for that a
restore_table_from_backup	Creates a new table from an existing backup
restore_table_to_point_in_time	Restores the specified table to the specified point in time within EarliestRestorableD
scan	The Scan operation returns one or more items and item attributes by accessing every
tag_resource	Associate a set of tags with an Amazon DynamoDB resource
transact_get_items	TransactGetItems is a synchronous operation that atomically retrieves multiple item
transact_write_items	TransactWriteItems is a synchronous write operation that groups up to 100 action re
untag_resource	Removes the association of tags from an Amazon DynamoDB resource

update_continuous_backups	UpdateContinuousBackups enables or disables point in time recovery for the specif
update_contributor_insights	Updates the status for contributor insights for a specific table or index
update_global_table	Adds or removes replicas in the specified global table
update_global_table_settings	Updates settings for a global table
update_item	Edits an existing item's attributes, or adds a new item to the table if it does not alrea
update_kinesis_streaming_destination	The command to update the Kinesis stream destination
update_table	Modifies the provisioned throughput settings, global secondary indexes, or Dynamo
update_table_replica_auto_scaling	Updates auto scaling settings on your global tables at once
update_time_to_live	The UpdateTimeToLive method enables or disables Time to Live (TTL) for the spec

Examples

```
## Not run:
svc <- dynamodb()
# This example reads multiple items from the Music table using a batch of
# three GetItem requests.  Only the AlbumTitle attribute is returned.
svc$batch_get_item(
  RequestItems = list(
    Music = list(
      Keys = list(
        list(
          Artist = list(
            S = "No One You Know"
          ),
          SongTitle = list(
            S = "Call Me Today"
          )
        ),
        list(
          Artist = list(
            S = "Acme Band"
          ),
          SongTitle = list(
            S = "Happy Day"
          )
        ),
        list(
          Artist = list(
            S = "No One You Know"
          ),
          SongTitle = list(
            S = "Scared of My Shadow"
          )
        )
      ),
      ProjectionExpression = "AlbumTitle"
    )
  )
)
```

```
## End(Not run)
```

dynamodbstreams *Amazon DynamoDB Streams*

Description

Amazon DynamoDB

Amazon DynamoDB Streams provides API actions for accessing streams and processing stream records. To learn more about application development with Streams, see Capturing Table Activity with DynamoDB Streams in the Amazon DynamoDB Developer Guide.

Usage

```
dynamodbstreams(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

`config` Optional configuration of credentials, endpoint, and/or region.

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy `https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e` `html`

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

Value

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- dynamodbstreams(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

describe_stream	Returns information about a stream, including the current status of the stream, its Amazon Resource Nam
get_records	Retrieves the stream records from a given shard
get_shard_iterator	Returns a shard iterator
list_streams	Returns an array of stream ARNs associated with the current account and endpoint

Examples

```
## Not run:
svc <- dynamodbstreams()
# The following example describes a stream with a given stream ARN.
svc$describe_stream(
  StreamArn = "arn:aws:dynamodb:us-west-2:111122223333:table/Forum/stream/2..."
)

## End(Not run)
```

elasticache *Amazon ElastiCache*

Description

Amazon ElastiCache is a web service that makes it easier to set up, operate, and scale a distributed cache in the cloud.

With ElastiCache, customers get all of the benefits of a high-performance, in-memory cache with less of the administrative burden involved in launching and managing a distributed cache. The service makes setup, scaling, and cluster failure handling much simpler than in a self-managed cache deployment.

In addition, through integration with Amazon CloudWatch, customers get enhanced visibility into the key performance statistics associated with their cache and can receive alarms if a part of their cache runs hot.

Usage

```
elasticache(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

`config` Optional configuration of credentials, endpoint, and/or region.

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

Value

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- elasticache(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

add_tags_to_resource	A tag is a key-value pair where the key and value are case-sensitive
authorize_cache_security_group_ingress	Allows network ingress to a cache security group
batch_apply_update_action	Apply the service update
batch_stop_update_action	Stop the service update
complete_migration	Complete the migration of data
copy_serverless_cache_snapshot	Creates a copy of an existing serverless cache's snapshot
copy_snapshot	Makes a copy of an existing snapshot
create_cache_cluster	Creates a cluster
create_cache_parameter_group	Creates a new Amazon ElastiCache cache parameter group
create_cache_security_group	Creates a new cache security group
create_cache_subnet_group	Creates a new cache subnet group
create_global_replication_group	Global Datastore offers fully managed, fast, reliable and secure cross-r
create_replication_group	Creates a Valkey or Redis OSS (cluster mode disabled) or a Valkey or R
create_serverless_cache	Creates a serverless cache
create_serverless_cache_snapshot	This API creates a copy of an entire ServerlessCache at a specific mom
create_snapshot	Creates a copy of an entire cluster or replication group at a specific mo
create_user	For Valkey engine version 7
create_user_group	For Valkey engine version 7
decrease_node_groups_in_global_replication_group	Decreases the number of node groups in a Global datastore
decrease_replica_count	Dynamically decreases the number of replicas in a Valkey or Redis OS

delete_cache_cluster	Deletes a previously provisioned cluster
delete_cache_parameter_group	Deletes the specified cache parameter group
delete_cache_security_group	Deletes a cache security group
delete_cache_subnet_group	Deletes a cache subnet group
delete_global_replication_group	Deleting a Global datastore is a two-step process:
delete_replication_group	Deletes an existing replication group
delete_serverless_cache	Deletes a specified existing serverless cache
delete_serverless_cache_snapshot	Deletes an existing serverless cache snapshot
delete_snapshot	Deletes an existing snapshot
delete_user	For Valkey engine version 7
delete_user_group	For Valkey engine version 7
describe_cache_clusters	Returns information about all provisioned clusters if no cluster identifi
describe_cache_engine_versions	Returns a list of the available cache engines and their versions
describe_cache_parameter_groups	Returns a list of cache parameter group descriptions
describe_cache_parameters	Returns the detailed parameter list for a particular cache parameter gro
describe_cache_security_groups	Returns a list of cache security group descriptions
describe_cache_subnet_groups	Returns a list of cache subnet group descriptions
describe_engine_default_parameters	Returns the default engine and system parameter information for the sp
describe_events	Returns events related to clusters, cache security groups, and cache par
describe_global_replication_groups	Returns information about a particular global replication group
describe_replication_groups	Returns information about a particular replication group
describe_reserved_cache_nodes	Returns information about reserved cache nodes for this account, or ab
describe_reserved_cache_nodes_offerings	Lists available reserved cache node offerings
describe_serverless_caches	Returns information about a specific serverless cache
describe_serverless_cache_snapshots	Returns information about serverless cache snapshots
describe_service_updates	Returns details of the service updates
describe_snapshots	Returns information about cluster or replication group snapshots
describe_update_actions	Returns details of the update actions
describe_user_groups	Returns a list of user groups
describe_users	Returns a list of users
disassociate_global_replication_group	Remove a secondary cluster from the Global datastore using the Globa
export_serverless_cache_snapshot	Provides the functionality to export the serverless cache snapshot data t
failover_global_replication_group	Used to failover the primary region to a secondary region
increase_node_groups_in_global_replication_group	Increase the number of node groups in the Global datastore
increase_replica_count	Dynamically increases the number of replicas in a Valkey or Redis OSS
list_allowed_node_type_modifications	Lists all available node types that you can scale with your cluster's repl
list_tags_for_resource	Lists all tags currently on a named resource
modify_cache_cluster	Modifies the settings for a cluster
modify_cache_parameter_group	Modifies the parameters of a cache parameter group
modify_cache_subnet_group	Modifies an existing cache subnet group
modify_global_replication_group	Modifies the settings for a Global datastore
modify_replication_group	Modifies the settings for a replication group
modify_replication_group_shard_configuration	Modifies a replication group's shards (node groups) by allowing you to
modify_serverless_cache	This API modifies the attributes of a serverless cache
modify_user	Changes user password(s) and/or access string
modify_user_group	Changes the list of users that belong to the user group
purchase_reserved_cache_nodes_offering	Allows you to purchase a reserved cache node offering
rebalance_slots_in_global_replication_group	Redistribute slots to ensure uniform distribution across existing shards

reboot_cache_cluster	Reboots some, or all, of the cache nodes within a provisioned cluster
remove_tags_from_resource	Removes the tags identified by the TagKeys list from the named resour
reset_cache_parameter_group	Modifies the parameters of a cache parameter group to the engine or sy
revoke_cache_security_group_ingress	Revokes ingress from a cache security group
start_migration	Start the migration of data
test_failover	Represents the input of a TestFailover operation which tests automatic
test_migration	Async API to test connection between source and target replication gro

Examples

```
## Not run:
svc <- elasticache()
# Adds up to 10 tags, key/value pairs, to a cluster or snapshot resource.
svc$add_tags_to_resource(
  ResourceName = "arn:aws:elasticache:us-east-1:1234567890:cluster:my-mem-cluster",
  Tags = list(
    list(
      Key = "APIVersion",
      Value = "20150202"
    ),
    list(
      Key = "Service",
      Value = "ElastiCache"
    )
  )
)

## End(Not run)
```

keyspaces   *Amazon Keyspaces*

Description

Amazon Keyspaces (for Apache Cassandra) is a scalable, highly available, and managed Apache Cassandra-compatible database service. Amazon Keyspaces makes it easy to migrate, run, and scale Cassandra workloads in the Amazon Web Services Cloud. With just a few clicks on the Amazon Web Services Management Console or a few lines of code, you can create keyspaces and tables in Amazon Keyspaces, without deploying any infrastructure or installing software.

In addition to supporting Cassandra Query Language (CQL) requests via open-source Cassandra drivers, Amazon Keyspaces supports data definition language (DDL) operations to manage keyspaces and tables using the Amazon Web Services SDK and CLI, as well as infrastructure as code (IaC) services and tools such as CloudFormation and Terraform. This API reference describes the supported DDL operations in detail.

For the list of all supported CQL APIs, see Supported Cassandra APIs, operations, and data types in Amazon Keyspaces in the *Amazon Keyspaces Developer Guide*.

To learn how Amazon Keyspaces API actions are recorded with CloudTrail, see Amazon Keyspaces information in CloudTrail in the *Amazon Keyspaces Developer Guide*.

For more information about Amazon Web Services APIs, for example how to implement retry logic or how to sign Amazon Web Services API requests, see Amazon Web Services APIs in the *General Reference*.

Usage

```
keyspaces(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

`config` Optional configuration of credentials, endpoint, and/or region.

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy `https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e` `html`

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.

- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

Value

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- keyspaces(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

create_keyspace	The CreateKeyspace operation adds a new keyspace to your account
create_table	The CreateTable operation adds a new table to the specified keyspace
create_type	The CreateType operation creates a new user-defined type in the specified keyspace

delete_keyspace	The DeleteKeyspace operation deletes a keyspace and all of its tables
delete_table	The DeleteTable operation deletes a table and all of its data
delete_type	The DeleteType operation deletes a user-defined type (UDT)
get_keyspace	Returns the name of the specified keyspace, the Amazon Resource Name (ARN), the replic
get_table	Returns information about the table, including the table's name and current status, the keysp
get_table_auto_scaling_settings	Returns auto scaling related settings of the specified table in JSON format
get_type	The GetType operation returns information about the type, for example the field definitions,
list_keyspaces	The ListKeyspaces operation returns a list of keyspaces
list_tables	The ListTables operation returns a list of tables for a specified keyspace
list_tags_for_resource	Returns a list of all tags associated with the specified Amazon Keyspaces resource
list_types	The ListTypes operation returns a list of types for a specified keyspace
restore_table	Restores the table to the specified point in time within the earliest_restorable_timestamp an
tag_resource	Associates a set of tags with a Amazon Keyspaces resource
untag_resource	Removes the association of tags from a Amazon Keyspaces resource
update_keyspace	Adds a new Amazon Web Services Region to the keyspace
update_table	Adds new columns to the table or updates one of the table's settings, for example capacity r

Examples

```
## Not run:
svc <- keyspaces()
svc$create_keyspace(
  Foo = 123
)

## End(Not run)
```

lakeformation *AWS Lake Formation*

Description

Lake Formation

Defines the public endpoint for the Lake Formation service.

Usage

```
lakeformation(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

`config` Optional configuration of credentials, endpoint, and/or region.

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

Value

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- lakeformation(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

add_lf_tags_to_resource	Attaches one or more LF-tags to an existing resource
assume_decorated_role_with_saml	Allows a caller to assume an IAM role decorated as the SAML user
batch_grant_permissions	Batch operation to grant permissions to the principal
batch_revoke_permissions	Batch operation to revoke permissions from the principal
cancel_transaction	Attempts to cancel the specified transaction
commit_transaction	Attempts to commit the specified transaction
create_data_cells_filter	Creates a data cell filter to allow one to grant access to certain colum
create_lake_formation_identity_center_configuration	Creates an IAM Identity Center connection with Lake Formation to
create_lake_formation_opt_in	Enforce Lake Formation permissions for the given databases, tables
create_lf_tag	Creates an LF-tag with the specified name and values
create_lf_tag_expression	Creates a new LF-Tag expression with the provided name, descriptio
delete_data_cells_filter	Deletes a data cell filter
delete_lake_formation_identity_center_configuration	Deletes an IAM Identity Center connection with Lake Formation
delete_lake_formation_opt_in	Remove the Lake Formation permissions enforcement of the given d
delete_lf_tag	Deletes the specified LF-tag given a key name
delete_lf_tag_expression	Deletes the LF-Tag expression
delete_objects_on_cancel	For a specific governed table, provides a list of Amazon S3 objects t
deregister_resource	Deregisters the resource as managed by the Data Catalog
describe_lake_formation_identity_center_configuration	Retrieves the instance ARN and application ARN for the connection
describe_resource	Retrieves the current data access role for the given resource register

describe_transaction	Returns the details of a single transaction
extend_transaction	Indicates to the service that the specified transaction is still active ar
get_data_cells_filter	Returns a data cells filter
get_data_lake_principal	Returns the identity of the invoking principal
get_data_lake_settings	Retrieves the list of the data lake administrators of a Lake Formatior
get_effective_permissions_for_path	Returns the Lake Formation permissions for a specified table or data
get_lf_tag	Returns an LF-tag definition
get_lf_tag_expression	Returns the details about the LF-Tag expression
get_query_state	Returns the state of a query previously submitted
get_query_statistics	Retrieves statistics on the planning and execution of a query
get_resource_lf_tags	Returns the LF-tags applied to a resource
get_table_objects	Returns the set of Amazon S3 objects that make up the specified go
get_temporary_glue_partition_credentials	This API is identical to GetTemporaryTableCredentials except that t
get_temporary_glue_table_credentials	Allows a caller in a secure environment to assume a role with permi
get_work_unit_results	Returns the work units resulting from the query
get_work_units	Retrieves the work units generated by the StartQueryPlanning opera
grant_permissions	Grants permissions to the principal to access metadata in the Data C
list_data_cells_filter	Lists all the data cell filters on a table
list_lake_formation_opt_ins	Retrieve the current list of resources and principals that are opt in to
list_lf_tag_expressions	Returns the LF-Tag expressions in caller's account filtered based on
list_lf_tags	Lists LF-tags that the requester has permission to view
list_permissions	Returns a list of the principal permissions on the resource, filtered b
list_resources	Lists the resources registered to be managed by the Data Catalog
list_table_storage_optimizers	Returns the configuration of all storage optimizers associated with a
list_transactions	Returns metadata about transactions and their status
put_data_lake_settings	Sets the list of data lake administrators who have admin privileges o
register_resource	Registers the resource as managed by the Data Catalog
remove_lf_tags_from_resource	Removes an LF-tag from the resource
revoke_permissions	Revokes permissions to the principal to access metadata in the Data
search_databases_by_lf_tags	This operation allows a search on DATABASE resources by TagCon
search_tables_by_lf_tags	This operation allows a search on TABLE resources by LFTags
start_query_planning	Submits a request to process a query statement
start_transaction	Starts a new transaction and returns its transaction ID
update_data_cells_filter	Updates a data cell filter
update_lake_formation_identity_center_configuration	Updates the IAM Identity Center connection parameters
update_lf_tag	Updates the list of possible values for the specified LF-tag key
update_lf_tag_expression	Updates the name of the LF-Tag expression to the new description a
update_resource	Updates the data access role used for vending access to the given (re
update_table_objects	Updates the manifest of Amazon S3 objects that make up the specif
update_table_storage_optimizer	Updates the configuration of the storage optimizers for a table

Examples

```
## Not run:
svc <- lakeformation()
svc$add_lf_tags_to_resource(
  Foo = 123
)

## End(Not run)
```

memorydb *Amazon MemoryDB*

Description

MemoryDB is a fully managed, Redis OSS-compatible, in-memory database that delivers ultra-fast performance and Multi-AZ durability for modern applications built using microservices architectures. MemoryDB stores the entire database in-memory, enabling low latency and high throughput data access. It is compatible with Redis OSS, a popular open source data store, enabling you to leverage Redis OSS' flexible and friendly data structures, APIs, and commands.

Usage

```
memorydb(config = list(), credentials = list(), endpoint = NULL, region = NULL)
```

Arguments

config Optional configuration of credentials, endpoint, and/or region.

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy `https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html`

credentials Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID

- **secret_access_key**: AWS secret access key
- **session_token**: AWS temporary session token

- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

endpoint Optional shorthand for complete URL to use for the constructed client.

region Optional shorthand for AWS Region used in instantiating the client.

Value

A client for the service. You can call the service's operations using syntax like svc$operation(...), where svc is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- memorydb(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

batch_update_cluster	Apply the service update to a list of clusters supplied
copy_snapshot	Makes a copy of an existing snapshot
create_acl	Creates an Access Control List
create_cluster	Creates a cluster
create_multi_region_cluster	Creates a new multi-Region cluster
create_parameter_group	Creates a new MemoryDB parameter group
create_snapshot	Creates a copy of an entire cluster at a specific moment in time
create_subnet_group	Creates a subnet group
create_user	Creates a MemoryDB user
delete_acl	Deletes an Access Control List
delete_cluster	Deletes a cluster
delete_multi_region_cluster	Deletes an existing multi-Region cluster
delete_parameter_group	Deletes the specified parameter group
delete_snapshot	Deletes an existing snapshot
delete_subnet_group	Deletes a subnet group
delete_user	Deletes a user
describe_ac_ls	Returns a list of ACLs
describe_clusters	Returns information about all provisioned clusters if no cluster identifier is speci
describe_engine_versions	Returns a list of the available Redis OSS engine versions
describe_events	Returns events related to clusters, security groups, and parameter groups
describe_multi_region_clusters	Returns details about one or more multi-Region clusters
describe_parameter_groups	Returns a list of parameter group descriptions
describe_parameters	Returns the detailed parameter list for a particular parameter group
describe_reserved_nodes	Returns information about reserved nodes for this account, or about a specified r
describe_reserved_nodes_offerings	Lists available reserved node offerings
describe_service_updates	Returns details of the service updates
describe_snapshots	Returns information about cluster snapshots
describe_subnet_groups	Returns a list of subnet group descriptions
describe_users	Returns a list of users
failover_shard	Used to failover a shard
list_allowed_multi_region_cluster_updates	Lists the allowed updates for a multi-Region cluster
list_allowed_node_type_updates	Lists all available node types that you can scale to from your cluster’s current no
list_tags	Lists all tags currently on a named resource
purchase_reserved_nodes_offering	Allows you to purchase a reserved node offering
reset_parameter_group	Modifies the parameters of a parameter group to the engine or system default va
tag_resource	A tag is a key-value pair where the key and value are case-sensitive
untag_resource	Use this operation to remove tags on a resource
update_acl	Changes the list of users that belong to the Access Control List
update_cluster	Modifies the settings for a cluster
update_multi_region_cluster	Updates the configuration of an existing multi-Region cluster
update_parameter_group	Updates the parameters of a parameter group
update_subnet_group	Updates a subnet group
update_user	Changes user password(s) and/or access string

Examples

```
## Not run:
svc <- memorydb()
svc$batch_update_cluster(
  Foo = 123
)

## End(Not run)
```

---

neptune *Amazon Neptune*

Description

Amazon Neptune is a fast, reliable, fully-managed graph database service that makes it easy to build and run applications that work with highly connected datasets. The core of Amazon Neptune is a purpose-built, high-performance graph database engine optimized for storing billions of relationships and querying the graph with milliseconds latency. Amazon Neptune supports popular graph models Property Graph and W3C's RDF, and their respective query languages Apache TinkerPop Gremlin and SPARQL, allowing you to easily build queries that efficiently navigate highly connected datasets. Neptune powers graph use cases such as recommendation engines, fraud detection, knowledge graphs, drug discovery, and network security.

This interface reference for Amazon Neptune contains documentation for a programming or command line interface you can use to manage Amazon Neptune. Note that Amazon Neptune is asynchronous, which means that some interfaces might require techniques such as polling or callback functions to determine when a command has been applied. In this reference, the parameter descriptions indicate whether a command is applied immediately, on the next instance reboot, or during the maintenance window. The reference structure is as follows, and we list following some related topics from the user guide.

Usage

```
neptune(config = list(), credentials = list(), endpoint = NULL, region = NULL)
```

Arguments

`config` Optional configuration of credentials, endpoint, and/or region.

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.

- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy `https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e` `html`

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

Value

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- neptune(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

add_role_to_db_cluster	Associates an Identity and Access Management (IAM) role with an Neptune D
add_source_identifier_to_subscription	Adds a source identifier to an existing event notification subscription
add_tags_to_resource	Adds metadata tags to an Amazon Neptune resource
apply_pending_maintenance_action	Applies a pending maintenance action to a resource (for example, to a DB inst
copy_db_cluster_parameter_group	Copies the specified DB cluster parameter group
copy_db_cluster_snapshot	Copies a snapshot of a DB cluster
copy_db_parameter_group	Copies the specified DB parameter group
create_db_cluster	Creates a new Amazon Neptune DB cluster
create_db_cluster_endpoint	Creates a new custom endpoint and associates it with an Amazon Neptune DB
create_db_cluster_parameter_group	Creates a new DB cluster parameter group
create_db_cluster_snapshot	Creates a snapshot of a DB cluster
create_db_instance	Creates a new DB instance
create_db_parameter_group	Creates a new DB parameter group
create_db_subnet_group	Creates a new DB subnet group
create_event_subscription	Creates an event notification subscription
create_global_cluster	Creates a Neptune global database spread across multiple Amazon Regions
delete_db_cluster	The DeleteDBCluster action deletes a previously provisioned DB cluster
delete_db_cluster_endpoint	Deletes a custom endpoint and removes it from an Amazon Neptune DB cluste
delete_db_cluster_parameter_group	Deletes a specified DB cluster parameter group
delete_db_cluster_snapshot	Deletes a DB cluster snapshot
delete_db_instance	The DeleteDBInstance action deletes a previously provisioned DB instance
delete_db_parameter_group	Deletes a specified DBParameterGroup
delete_db_subnet_group	Deletes a DB subnet group
delete_event_subscription	Deletes an event notification subscription
delete_global_cluster	Deletes a global database
describe_db_cluster_endpoints	Returns information about endpoints for an Amazon Neptune DB cluster
describe_db_cluster_parameter_groups	Returns a list of DBClusterParameterGroup descriptions
describe_db_cluster_parameters	Returns the detailed parameter list for a particular DB cluster parameter group
describe_db_clusters	Returns information about provisioned DB clusters, and supports pagination
describe_db_cluster_snapshot_attributes	Returns a list of DB cluster snapshot attribute names and values for a manual I
describe_db_cluster_snapshots	Returns information about DB cluster snapshots
describe_db_engine_versions	Returns a list of the available DB engines

describe_db_instances	Returns information about provisioned instances, and supports pagination
describe_db_parameter_groups	Returns a list of DBParameterGroup descriptions
describe_db_parameters	Returns the detailed parameter list for a particular DB parameter group
describe_db_subnet_groups	Returns a list of DBSubnetGroup descriptions
describe_engine_default_cluster_parameters	Returns the default engine and system parameter information for the cluster da
describe_engine_default_parameters	Returns the default engine and system parameter information for the specified
describe_event_categories	Displays a list of categories for all event source types, or, if specified, for a spe
describe_events	Returns events related to DB instances, DB security groups, DB snapshots, and
describe_event_subscriptions	Lists all the subscription descriptions for a customer account
describe_global_clusters	Returns information about Neptune global database clusters
describe_orderable_db_instance_options	Returns a list of orderable DB instance options for the specified engine
describe_pending_maintenance_actions	Returns a list of resources (for example, DB instances) that have at least one p
describe_valid_db_instance_modifications	You can call DescribeValidDBInstanceModifications to learn what modificatio
failover_db_cluster	Forces a failover for a DB cluster
failover_global_cluster	Initiates the failover process for a Neptune global database
list_tags_for_resource	Lists all tags on an Amazon Neptune resource
modify_db_cluster	Modify a setting for a DB cluster
modify_db_cluster_endpoint	Modifies the properties of an endpoint in an Amazon Neptune DB cluster
modify_db_cluster_parameter_group	Modifies the parameters of a DB cluster parameter group
modify_db_cluster_snapshot_attribute	Adds an attribute and values to, or removes an attribute and values from, a man
modify_db_instance	Modifies settings for a DB instance
modify_db_parameter_group	Modifies the parameters of a DB parameter group
modify_db_subnet_group	Modifies an existing DB subnet group
modify_event_subscription	Modifies an existing event notification subscription
modify_global_cluster	Modify a setting for an Amazon Neptune global cluster
promote_read_replica_db_cluster	Not supported
reboot_db_instance	You might need to reboot your DB instance, usually for maintenance reasons
remove_from_global_cluster	Detaches a Neptune DB cluster from a Neptune global database
remove_role_from_db_cluster	Disassociates an Identity and Access Management (IAM) role from a DB clus
remove_source_identifier_from_subscription	Removes a source identifier from an existing event notification subscription
remove_tags_from_resource	Removes metadata tags from an Amazon Neptune resource
reset_db_cluster_parameter_group	Modifies the parameters of a DB cluster parameter group to the default value
reset_db_parameter_group	Modifies the parameters of a DB parameter group to the engine/system default
restore_db_cluster_from_snapshot	Creates a new DB cluster from a DB snapshot or DB cluster snapshot
restore_db_cluster_to_point_in_time	Restores a DB cluster to an arbitrary point in time
start_db_cluster	Starts an Amazon Neptune DB cluster that was stopped using the Amazon con
stop_db_cluster	Stops an Amazon Neptune DB cluster

Examples

```
## Not run:
svc <- neptune()
svc$add_role_to_db_cluster(
  Foo = 123
)

## End(Not run)
```

neptunedata *Amazon NeptuneData*

Description

Neptune Data API

The Amazon Neptune data API provides SDK support for more than 40 of Neptune's data operations, including data loading, query execution, data inquiry, and machine learning. It supports the Gremlin and openCypher query languages, and is available in all SDK languages. It automatically signs API requests and greatly simplifies integrating Neptune into your applications.

Usage

```
neptunedata(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

`config` Optional configuration of credentials, endpoint, and/or region.

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy `https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e` `html`

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

Value

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- neptunedata(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

cancel_gremlin_query	Cancels a Gremlin query
cancel_loader_job	Cancels a specified load job
cancel_ml_data_processing_job	Cancels a Neptune ML data processing job
cancel_ml_model_training_job	Cancels a Neptune ML model training job
cancel_ml_model_transform_job	Cancels a specified model transform job
cancel_open_cypher_query	Cancels a specified openCypher query
create_ml_endpoint	Creates a new Neptune ML inference endpoint that lets you query one specific model t
delete_ml_endpoint	Cancels the creation of a Neptune ML inference endpoint
delete_propertygraph_statistics	Deletes statistics for Gremlin and openCypher (property graph) data
delete_sparql_statistics	Deletes SPARQL statistics
execute_fast_reset	The fast reset REST API lets you reset a Neptune graph quicky and easily, removing a
execute_gremlin_explain_query	Executes a Gremlin Explain query
execute_gremlin_profile_query	Executes a Gremlin Profile query, which runs a specified traversal, collects various me
execute_gremlin_query	This commands executes a Gremlin query
execute_open_cypher_explain_query	Executes an openCypher explain request
execute_open_cypher_query	Executes an openCypher query
get_engine_status	Retrieves the status of the graph database on the host
get_gremlin_query_status	Gets the status of a specified Gremlin query
get_loader_job_status	Gets status information about a specified load job
get_ml_data_processing_job	Retrieves information about a specified data processing job
get_ml_endpoint	Retrieves details about an inference endpoint
get_ml_model_training_job	Retrieves information about a Neptune ML model training job
get_ml_model_transform_job	Gets information about a specified model transform job
get_open_cypher_query_status	Retrieves the status of a specified openCypher query
get_propertygraph_statistics	Gets property graph statistics (Gremlin and openCypher)
get_propertygraph_stream	Gets a stream for a property graph
get_propertygraph_summary	Gets a graph summary for a property graph
get_rdf_graph_summary	Gets a graph summary for an RDF graph
get_sparql_statistics	Gets RDF statistics (SPARQL)
get_sparql_stream	Gets a stream for an RDF graph
list_gremlin_queries	Lists active Gremlin queries
list_loader_jobs	Retrieves a list of the loadIds for all active loader jobs
list_ml_data_processing_jobs	Returns a list of Neptune ML data processing jobs
list_ml_endpoints	Lists existing inference endpoints
list_ml_model_training_jobs	Lists Neptune ML model-training jobs
list_ml_model_transform_jobs	Returns a list of model transform job IDs
list_open_cypher_queries	Lists active openCypher queries
manage_propertygraph_statistics	Manages the generation and use of property graph statistics
manage_sparql_statistics	Manages the generation and use of RDF graph statistics
start_loader_job	Starts a Neptune bulk loader job to load data from an Amazon S3 bucket into a Neptun
start_ml_data_processing_job	Creates a new Neptune ML data processing job for processing the graph data exported
start_ml_model_training_job	Creates a new Neptune ML model training job
start_ml_model_transform_job	Creates a new model transform job

Examples

```
## Not run:
svc <- neptunedata()
svc$cancel_gremlin_query(
  Foo = 123
)

## End(Not run)
```

---

qldb *Amazon QLDB*

Description

The resource management API for Amazon QLDB

Usage

```
qldb(config = list(), credentials = list(), endpoint = NULL, region = NULL)
```

Arguments

`config` Optional configuration of credentials, endpoint, and/or region.

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:

    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

Value

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- qldb(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

cancel_journal_kinesis_stream	Ends a given Amazon QLDB journal stream
create_ledger	Creates a new ledger in your Amazon Web Services account in the current Region
delete_ledger	Deletes a ledger and all of its contents
describe_journal_kinesis_stream	Returns detailed information about a given Amazon QLDB journal stream
describe_journal_s3_export	Returns information about a journal export job, including the ledger name, export I
describe_ledger	Returns information about a ledger, including its state, permissions mode, encryptio
export_journal_to_s3	Exports journal contents within a date and time range from a ledger into a specified
get_block	Returns a block object at a specified address in a journal
get_digest	Returns the digest of a ledger at the latest committed block in the journal
get_revision	Returns a revision data object for a specified document ID and block address
list_journal_kinesis_streams_for_ledger	Returns all Amazon QLDB journal streams for a given ledger
list_journal_s3_exports	Returns all journal export jobs for all ledgers that are associated with the current A
list_journal_s3_exports_for_ledger	Returns all journal export jobs for a specified ledger
list_ledgers	Returns all ledgers that are associated with the current Amazon Web Services accou
list_tags_for_resource	Returns all tags for a specified Amazon QLDB resource
stream_journal_to_kinesis	Creates a journal stream for a given Amazon QLDB ledger
tag_resource	Adds one or more tags to a specified Amazon QLDB resource
untag_resource	Removes one or more tags from a specified Amazon QLDB resource
update_ledger	Updates properties on a ledger
update_ledger_permissions_mode	Updates the permissions mode of a ledger

Examples

```
## Not run:
svc <- qldb()
svc$cancel_journal_kinesis_stream(
  Foo = 123
)

## End(Not run)
```

---

qldbsession *Amazon QLDB Session*

Description

The transactional data APIs for Amazon QLDB

Instead of interacting directly with this API, we recommend using the QLDB driver or the QLDB shell to execute data transactions on a ledger.

- If you are working with an AWS SDK, use the QLDB driver. The driver provides a high-level abstraction layer above this *QLDB Session* data plane and manages `send_command` API calls for you. For information and a list of supported programming languages, see Getting started with the driver in the *Amazon QLDB Developer Guide*.

- If you are working with the AWS Command Line Interface (AWS CLI), use the QLDB shell. The shell is a command line interface that uses the QLDB driver to interact with a ledger. For information, see Accessing Amazon QLDB using the QLDB shell.

Usage

```
qldbsession(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

`config` Optional configuration of credentials, endpoint, and/or region.

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy `https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html`

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

Value

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- qldbsession(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

send_command	Sends a command to an Amazon QLDB ledger

Examples

```
## Not run:
svc <- qldbsession()
```

```
svc$send_command(
  Foo = 123
)

## End(Not run)
```

rds *Amazon Relational Database Service*

Description

Amazon Relational Database Service (Amazon RDS) is a web service that makes it easier to set up, operate, and scale a relational database in the cloud. It provides cost-efficient, resizeable capacity for an industry-standard relational database and manages common database administration tasks, freeing up developers to focus on what makes their applications and businesses unique.

Amazon RDS gives you access to the capabilities of a MySQL, MariaDB, PostgreSQL, Microsoft SQL Server, Oracle, Db2, or Amazon Aurora database server. These capabilities mean that the code, applications, and tools you already use today with your existing databases work with Amazon RDS without modification. Amazon RDS automatically backs up your database and maintains the database software that powers your DB instance. Amazon RDS is flexible: you can scale your DB instance's compute resources and storage capacity to meet your application's demand. As with all Amazon Web Services, there are no up-front investments, and you pay only for the resources you use.

This interface reference for Amazon RDS contains documentation for a programming or command line interface you can use to manage Amazon RDS. Amazon RDS is asynchronous, which means that some interfaces might require techniques such as polling or callback functions to determine when a command has been applied. In this reference, the parameter descriptions indicate whether a command is applied immediately, on the next instance reboot, or during the maintenance window. The reference structure is as follows, and we list following some related topics from the user guide.

Amazon RDS API Reference

- For the alphabetical list of API actions, see API Actions.
- For the alphabetical list of data types, see Data Types.
- For a list of common query parameters, see Common Parameters.
- For descriptions of the error codes, see Common Errors.

Amazon RDS User Guide

- For a summary of the Amazon RDS interfaces, see Available RDS Interfaces.
- For more information about how to use the Query API, see Using the Query API.

Usage

```
rds(config = list(), credentials = list(), endpoint = NULL, region = NULL)
```

Arguments

`config` Optional configuration of credentials, endpoint, and/or region.

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

Value

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- rds(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

add_role_to_db_cluster	Associates an Identity and Access Management (IAM) role with a DB cl
add_role_to_db_instance	Associates an Amazon Web Services Identity and Access Management (
add_source_identifier_to_subscription	Adds a source identifier to an existing RDS event notification subscriptio
add_tags_to_resource	Adds metadata tags to an Amazon RDS resource
apply_pending_maintenance_action	Applies a pending maintenance action to a resource (for example, to a D
authorize_db_security_group_ingress	Enables ingress to a DBSecurityGroup using one of two forms of authori
backtrack_db_cluster	Backtracks a DB cluster to a specific time, without creating a new DB cl
build_auth_token	Return an authentication token for a database connection
build_auth_token_v2	Generates an auth token used to connect to a db with IAM credentials
cancel_export_task	Cancels an export task in progress that is exporting a snapshot or cluster
copy_db_cluster_parameter_group	Copies the specified DB cluster parameter group
copy_db_cluster_snapshot	Copies a snapshot of a DB cluster
copy_db_parameter_group	Copies the specified DB parameter group
copy_db_snapshot	Copies the specified DB snapshot
copy_option_group	Copies the specified option group
create_blue_green_deployment	Creates a blue/green deployment
create_custom_db_engine_version	Creates a custom DB engine version (CEV)
create_db_cluster	Creates a new Amazon Aurora DB cluster or Multi-AZ DB cluster
create_db_cluster_endpoint	Creates a new custom endpoint and associates it with an Amazon Aurora
create_db_cluster_parameter_group	Creates a new DB cluster parameter group

create_db_cluster_snapshot	Creates a snapshot of a DB cluster
create_db_instance	Creates a new DB instance
create_db_instance_read_replica	Creates a new DB instance that acts as a read replica for an existing sour
create_db_parameter_group	Creates a new DB parameter group
create_db_proxy	Creates a new DB proxy
create_db_proxy_endpoint	Creates a DBProxyEndpoint
create_db_security_group	Creates a new DB security group
create_db_shard_group	Creates a new DB shard group for Aurora Limitless Database
create_db_snapshot	Creates a snapshot of a DB instance
create_db_subnet_group	Creates a new DB subnet group
create_event_subscription	Creates an RDS event notification subscription
create_global_cluster	Creates an Aurora global database spread across multiple Amazon Web S
create_integration	Creates a zero-ETL integration with Amazon Redshift
create_option_group	Creates a new option group
create_tenant_database	Creates a tenant database in a DB instance that uses the multi-tenant con
delete_blue_green_deployment	Deletes a blue/green deployment
delete_custom_db_engine_version	Deletes a custom engine version
delete_db_cluster	The DeleteDBCluster action deletes a previously provisioned DB cluster
delete_db_cluster_automated_backup	Deletes automated backups using the DbClusterResourceId value of the s
delete_db_cluster_endpoint	Deletes a custom endpoint and removes it from an Amazon Aurora DB c
delete_db_cluster_parameter_group	Deletes a specified DB cluster parameter group
delete_db_cluster_snapshot	Deletes a DB cluster snapshot
delete_db_instance	Deletes a previously provisioned DB instance
delete_db_instance_automated_backup	Deletes automated backups using the DbiResourceId value of the source
delete_db_parameter_group	Deletes a specified DB parameter group
delete_db_proxy	Deletes an existing DB proxy
delete_db_proxy_endpoint	Deletes a DBProxyEndpoint
delete_db_security_group	Deletes a DB security group
delete_db_shard_group	Deletes an Aurora Limitless Database DB shard group
delete_db_snapshot	Deletes a DB snapshot
delete_db_subnet_group	Deletes a DB subnet group
delete_event_subscription	Deletes an RDS event notification subscription
delete_global_cluster	Deletes a global database cluster
delete_integration	Deletes a zero-ETL integration with Amazon Redshift
delete_option_group	Deletes an existing option group
delete_tenant_database	Deletes a tenant database from your DB instance
deregister_db_proxy_targets	Remove the association between one or more DBProxyTarget data struct
describe_account_attributes	Lists all of the attributes for a customer account
describe_blue_green_deployments	Describes one or more blue/green deployments
describe_certificates	Lists the set of certificate authority (CA) certificates provided by Amazon
describe_db_cluster_automated_backups	Displays backups for both current and deleted DB clusters
describe_db_cluster_backtracks	Returns information about backtracks for a DB cluster
describe_db_cluster_endpoints	Returns information about endpoints for an Amazon Aurora DB cluster
describe_db_cluster_parameter_groups	Returns a list of DBClusterParameterGroup descriptions
describe_db_cluster_parameters	Returns the detailed parameter list for a particular DB cluster parameter g
describe_db_clusters	Describes existing Amazon Aurora DB clusters and Multi-AZ DB cluste
describe_db_cluster_snapshot_attributes	Returns a list of DB cluster snapshot attribute names and values for a ma
describe_db_cluster_snapshots	Returns information about DB cluster snapshots

describe_db_engine_versions	Describes the properties of specific versions of DB engines
describe_db_instance_automated_backups	Displays backups for both current and deleted instances
describe_db_instances	Describes provisioned RDS instances
describe_db_log_files	Returns a list of DB log files for the DB instance
describe_db_parameter_groups	Returns a list of DBParameterGroup descriptions
describe_db_parameters	Returns the detailed parameter list for a particular DB parameter group
describe_db_proxies	Returns information about DB proxies
describe_db_proxy_endpoints	Returns information about DB proxy endpoints
describe_db_proxy_target_groups	Returns information about DB proxy target groups, represented by DBPr
describe_db_proxy_targets	Returns information about DBProxyTarget objects
describe_db_recommendations	Describes the recommendations to resolve the issues for your DB instanc
describe_db_security_groups	Returns a list of DBSecurityGroup descriptions
describe_db_shard_groups	Describes existing Aurora Limitless Database DB shard groups
describe_db_snapshot_attributes	Returns a list of DB snapshot attribute names and values for a manual DI
describe_db_snapshots	Returns information about DB snapshots
describe_db_snapshot_tenant_databases	Describes the tenant databases that exist in a DB snapshot
describe_db_subnet_groups	Returns a list of DBSubnetGroup descriptions
describe_engine_default_cluster_parameters	Returns the default engine and system parameter information for the clus
describe_engine_default_parameters	Returns the default engine and system parameter information for the spec
describe_event_categories	Displays a list of categories for all event source types, or, if specified, for
describe_events	Returns events related to DB instances, DB clusters, DB parameter group
describe_event_subscriptions	Lists all the subscription descriptions for a customer account
describe_export_tasks	Returns information about a snapshot or cluster export to Amazon S3
describe_global_clusters	Returns information about Aurora global database clusters
describe_integrations	Describe one or more zero-ETL integrations with Amazon Redshift
describe_option_group_options	Describes all available options for the specified engine
describe_option_groups	Describes the available option groups
describe_orderable_db_instance_options	Describes the orderable DB instance options for a specified DB engine
describe_pending_maintenance_actions	Returns a list of resources (for example, DB instances) that have at least
describe_reserved_db_instances	Returns information about reserved DB instances for this account, or abo
describe_reserved_db_instances_offerings	Lists available reserved DB instance offerings
describe_source_regions	Returns a list of the source Amazon Web Services Regions where the cur
describe_tenant_databases	Describes the tenant databases in a DB instance that uses the multi-tenan
describe_valid_db_instance_modifications	You can call DescribeValidDBInstanceModifications to learn what modif
disable_http_endpoint	Disables the HTTP endpoint for the specified DB cluster
download_db_log_file_portion	Downloads all or a portion of the specified log file, up to 1 MB in size
enable_http_endpoint	Enables the HTTP endpoint for the DB cluster
failover_db_cluster	Forces a failover for a DB cluster
failover_global_cluster	Promotes the specified secondary DB cluster to be the primary DB cluste
list_tags_for_resource	Lists all tags on an Amazon RDS resource
modify_activity_stream	Changes the audit policy state of a database activity stream to either lock
modify_certificates	Override the system-default Secure Sockets Layer/Transport Layer Secur
modify_current_db_cluster_capacity	Set the capacity of an Aurora Serverless v1 DB cluster to a specific value
modify_custom_db_engine_version	Modifies the status of a custom engine version (CEV)
modify_db_cluster	Modifies the settings of an Amazon Aurora DB cluster or a Multi-AZ DI
modify_db_cluster_endpoint	Modifies the properties of an endpoint in an Amazon Aurora DB cluster
modify_db_cluster_parameter_group	Modifies the parameters of a DB cluster parameter group
modify_db_cluster_snapshot_attribute	Adds an attribute and values to, or removes an attribute and values from,

modify_db_instance	Modifies settings for a DB instance
modify_db_parameter_group	Modifies the parameters of a DB parameter group
modify_db_proxy	Changes the settings for an existing DB proxy
modify_db_proxy_endpoint	Changes the settings for an existing DB proxy endpoint
modify_db_proxy_target_group	Modifies the properties of a DBProxyTargetGroup
modify_db_recommendation	Updates the recommendation status and recommended action status for t
modify_db_shard_group	Modifies the settings of an Aurora Limitless Database DB shard group
modify_db_snapshot	Updates a manual DB snapshot with a new engine version
modify_db_snapshot_attribute	Adds an attribute and values to, or removes an attribute and values from,
modify_db_subnet_group	Modifies an existing DB subnet group
modify_event_subscription	Modifies an existing RDS event notification subscription
modify_global_cluster	Modifies a setting for an Amazon Aurora global database cluster
modify_integration	Modifies a zero-ETL integration with Amazon Redshift
modify_option_group	Modifies an existing option group
modify_tenant_database	Modifies an existing tenant database in a DB instance
promote_read_replica	Promotes a read replica DB instance to a standalone DB instance
promote_read_replica_db_cluster	Promotes a read replica DB cluster to a standalone DB cluster
purchase_reserved_db_instances_offering	Purchases a reserved DB instance offering
reboot_db_cluster	You might need to reboot your DB cluster, usually for maintenance reaso
reboot_db_instance	You might need to reboot your DB instance, usually for maintenance rea
reboot_db_shard_group	You might need to reboot your DB shard group, usually for maintenance
register_db_proxy_targets	Associate one or more DBProxyTarget data structures with a DBProxyTa
remove_from_global_cluster	Detaches an Aurora secondary cluster from an Aurora global database cl
remove_role_from_db_cluster	Removes the asssociation of an Amazon Web Services Identity and Acce
remove_role_from_db_instance	Disassociates an Amazon Web Services Identity and Access Managemen
remove_source_identifier_from_subscription	Removes a source identifier from an existing RDS event notification subs
remove_tags_from_resource	Removes metadata tags from an Amazon RDS resource
reset_db_cluster_parameter_group	Modifies the parameters of a DB cluster parameter group to the default v
reset_db_parameter_group	Modifies the parameters of a DB parameter group to the engine/system d
restore_db_cluster_from_s3	Creates an Amazon Aurora DB cluster from MySQL data stored in an A
restore_db_cluster_from_snapshot	Creates a new DB cluster from a DB snapshot or DB cluster snapshot
restore_db_cluster_to_point_in_time	Restores a DB cluster to an arbitrary point in time
restore_db_instance_from_db_snapshot	Creates a new DB instance from a DB snapshot
restore_db_instance_from_s3	Amazon Relational Database Service (Amazon RDS) supports importing
restore_db_instance_to_point_in_time	Restores a DB instance to an arbitrary point in time
revoke_db_security_group_ingress	Revokes ingress from a DBSecurityGroup for previously authorized IP ra
start_activity_stream	Starts a database activity stream to monitor activity on the database
start_db_cluster	Starts an Amazon Aurora DB cluster that was stopped using the Amazon
start_db_instance	Starts an Amazon RDS DB instance that was stopped using the Amazon
start_db_instance_automated_backups_replication	Enables replication of automated backups to a different Amazon Web Se
start_export_task	Starts an export of DB snapshot or DB cluster data to Amazon S3
stop_activity_stream	Stops a database activity stream that was started using the Amazon Web
stop_db_cluster	Stops an Amazon Aurora DB cluster
stop_db_instance	Stops an Amazon RDS DB instance temporarily
stop_db_instance_automated_backups_replication	Stops automated backup replication for a DB instance
switchover_blue_green_deployment	Switches over a blue/green deployment
switchover_global_cluster	Switches over the specified secondary DB cluster to be the new primary
switchover_read_replica	Switches over an Oracle standby database in an Oracle Data Guard envir

Examples

```
## Not run:
svc <- rds()
# This example add a source identifier to an event notification
# subscription.
svc$add_source_identifier_to_subscription(
  SourceIdentifier = "mymysqlinstance",
  SubscriptionName = "mymysqleventsubscription"
)

## End(Not run)
```

---

rdsdataservice *AWS RDS DataService*

Description

RDS Data API

Amazon RDS provides an HTTP endpoint to run SQL statements on an Amazon Aurora DB cluster. To run these statements, you use the RDS Data API (Data API).

Data API is available with the following types of Aurora databases:

- Aurora PostgreSQL - Serverless v2, provisioned, and Serverless v1
- Aurora MySQL - Serverless v2, provisioned, and Serverless v1

For more information about the Data API, see Using RDS Data API in the *Amazon Aurora User Guide*.

Usage

```
rdsdataservice(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

`config` Optional configuration of credentials, endpoint, and/or region.

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID

        * **secret_access_key**: AWS secret access key
        * **session_token**: AWS temporary session token
    - **profile**: The name of a profile to use. If not given, then the default profile is used.
    - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

Value

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- rdsdataservice(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

batch_execute_statement	Runs a batch SQL statement over an array of data
begin_transaction	Starts a SQL transaction
commit_transaction	Ends a SQL transaction started with the BeginTransaction operation and commits the changes
execute_sql	Runs one or more SQL statements
execute_statement	Runs a SQL statement against a database
rollback_transaction	Performs a rollback of a transaction

Examples

```
## Not run:
svc <- rdsdataservice()
svc$batch_execute_statement(
  Foo = 123
)

## End(Not run)
```

redshift *Amazon Redshift*

Description

Overview

This is an interface reference for Amazon Redshift. It contains documentation for one of the programming or command line interfaces you can use to manage Amazon Redshift clusters. Note that Amazon Redshift is asynchronous, which means that some interfaces may require techniques, such as polling or asynchronous callback handlers, to determine when a command has been applied. In this reference, the parameter descriptions indicate whether a change is applied immediately, on the next instance reboot, or during the next maintenance window. For a summary of the Amazon Redshift cluster management interfaces, go to Using the Amazon Redshift Management Interfaces.

Amazon Redshift manages all the work of setting up, operating, and scaling a data warehouse: provisioning capacity, monitoring and backing up the cluster, and applying patches and upgrades to the Amazon Redshift engine. You can focus on using your data to acquire new insights for your business and customers.

If you are a first-time user of Amazon Redshift, we recommend that you begin by reading the Amazon Redshift Getting Started Guide.

If you are a database developer, the Amazon Redshift Database Developer Guide explains how to design, build, query, and maintain the databases that make up your data warehouse.

Usage

```
redshift(config = list(), credentials = list(), endpoint = NULL, region = NULL)
```

Arguments

`config` Optional configuration of credentials, endpoint, and/or region.

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy `https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e` `html`

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

Value

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- redshift(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

accept_reserved_node_exchange	Exchanges a DC1 Reserved Node for a DC2 Reserved Node with no cl
add_partner	Adds a partner integration to a cluster
associate_data_share_consumer	From a datashare consumer account, associates a datashare with the ac
authorize_cluster_security_group_ingress	Adds an inbound (ingress) rule to an Amazon Redshift security group
authorize_data_share	From a data producer account, authorizes the sharing of a datashare wi
authorize_endpoint_access	Grants access to a cluster
authorize_snapshot_access	Authorizes the specified Amazon Web Services account to restore the s
batch_delete_cluster_snapshots	Deletes a set of cluster snapshots
batch_modify_cluster_snapshots	Modifies the settings for a set of cluster snapshots
cancel_resize	Cancels a resize operation for a cluster
copy_cluster_snapshot	Copies the specified automated cluster snapshot to a new manual cluste
create_authentication_profile	Creates an authentication profile with the specified parameters
create_cluster	Creates a new cluster with the specified parameters
create_cluster_parameter_group	Creates an Amazon Redshift parameter group
create_cluster_security_group	Creates a new Amazon Redshift security group
create_cluster_snapshot	Creates a manual snapshot of the specified cluster
create_cluster_subnet_group	Creates a new Amazon Redshift subnet group
create_custom_domain_association	Used to create a custom domain name for a cluster
create_endpoint_access	Creates a Redshift-managed VPC endpoint
create_event_subscription	Creates an Amazon Redshift event notification subscription
create_hsm_client_certificate	Creates an HSM client certificate that an Amazon Redshift cluster will
create_hsm_configuration	Creates an HSM configuration that contains the information required b
create_integration	Creates a zero-ETL integration or S3 event integration with Amazon R
create_redshift_idc_application	Creates an Amazon Redshift application for use with IAM Identity Cer
create_scheduled_action	Creates a scheduled action
create_snapshot_copy_grant	Creates a snapshot copy grant that permits Amazon Redshift to use an
create_snapshot_schedule	Create a snapshot schedule that can be associated to a cluster and whic
create_tags	Adds tags to a cluster
create_usage_limit	Creates a usage limit for a specified Amazon Redshift feature on a clus
deauthorize_data_share	From a datashare producer account, removes authorization from the sp
delete_authentication_profile	Deletes an authentication profile
delete_cluster	Deletes a previously provisioned cluster without its final snapshot bein
delete_cluster_parameter_group	Deletes a specified Amazon Redshift parameter group
delete_cluster_security_group	Deletes an Amazon Redshift security group
delete_cluster_snapshot	Deletes the specified manual snapshot
delete_cluster_subnet_group	Deletes the specified cluster subnet group
delete_custom_domain_association	Contains information about deleting a custom domain association for a
delete_endpoint_access	Deletes a Redshift-managed VPC endpoint
delete_event_subscription	Deletes an Amazon Redshift event notification subscription
delete_hsm_client_certificate	Deletes the specified HSM client certificate
delete_hsm_configuration	Deletes the specified Amazon Redshift HSM configuration
delete_integration	Deletes a zero-ETL integration or S3 event integration with Amazon R
delete_partner	Deletes a partner integration from a cluster
delete_redshift_idc_application	Deletes an Amazon Redshift IAM Identity Center application
delete_resource_policy	Deletes the resource policy for a specified resource
delete_scheduled_action	Deletes a scheduled action

delete_snapshot_copy_grant	Deletes the specified snapshot copy grant
delete_snapshot_schedule	Deletes a snapshot schedule
delete_tags	Deletes tags from a resource
delete_usage_limit	Deletes a usage limit from a cluster
deregister_namespace	Deregisters a cluster or serverless namespace from the Amazon Web S
describe_account_attributes	Returns a list of attributes attached to an account
describe_authentication_profiles	Describes an authentication profile
describe_cluster_db_revisions	Returns an array of ClusterDbRevision objects
describe_cluster_parameter_groups	Returns a list of Amazon Redshift parameter groups, including parame
describe_cluster_parameters	Returns a detailed list of parameters contained within the specified Am
describe_clusters	Returns properties of provisioned clusters including general cluster pro
describe_cluster_security_groups	Returns information about Amazon Redshift security groups
describe_cluster_snapshots	Returns one or more snapshot objects, which contain metadata about y
describe_cluster_subnet_groups	Returns one or more cluster subnet group objects, which contain metad
describe_cluster_tracks	Returns a list of all the available maintenance tracks
describe_cluster_versions	Returns descriptions of the available Amazon Redshift cluster versions
describe_custom_domain_associations	Contains information about custom domain associations for a cluster
describe_data_shares	Shows the status of any inbound or outbound datashares available in th
describe_data_shares_for_consumer	Returns a list of datashares where the account identifier being called is
describe_data_shares_for_producer	Returns a list of datashares when the account identifier being called is
describe_default_cluster_parameters	Returns a list of parameter settings for the specified parameter group fa
describe_endpoint_access	Describes a Redshift-managed VPC endpoint
describe_endpoint_authorization	Describes an endpoint authorization
describe_event_categories	Displays a list of event categories for all event source types, or for a sp
describe_events	Returns events related to clusters, security groups, snapshots, and para
describe_event_subscriptions	Lists descriptions of all the Amazon Redshift event notification subscri
describe_hsm_client_certificates	Returns information about the specified HSM client certificate
describe_hsm_configurations	Returns information about the specified Amazon Redshift HSM config
describe_inbound_integrations	Returns a list of inbound integrations
describe_integrations	Describes one or more zero-ETL or S3 event integrations with Amazon
describe_logging_status	Describes whether information, such as queries and connection attemp
describe_node_configuration_options	Returns properties of possible node configurations such as node type, n
describe_orderable_cluster_options	Returns a list of orderable cluster options
describe_partners	Returns information about the partner integrations defined for a cluster
describe_redshift_idc_applications	Lists the Amazon Redshift IAM Identity Center applications
describe_reserved_node_exchange_status	Returns exchange status details and associated metadata for a reserved-
describe_reserved_node_offerings	Returns a list of the available reserved node offerings by Amazon Reds
describe_reserved_nodes	Returns the descriptions of the reserved nodes
describe_resize	Returns information about the last resize operation for the specified clu
describe_scheduled_actions	Describes properties of scheduled actions
describe_snapshot_copy_grants	Returns a list of snapshot copy grants owned by the Amazon Web Serv
describe_snapshot_schedules	Returns a list of snapshot schedules
describe_storage	Returns account level backups storage size and provisional storage
describe_table_restore_status	Lists the status of one or more table restore requests made using the Re
describe_tags	Returns a list of tags
describe_usage_limits	Shows usage limits on a cluster
disable_logging	Stops logging information, such as queries and connection attempts, fo
disable_snapshot_copy	Disables the automatic copying of snapshots from one region to anothe

disassociate_data_share_consumer	From a datashare consumer account, remove association for the specifi
enable_logging	Starts logging information, such as queries and connection attempts, fo
enable_snapshot_copy	Enables the automatic copy of snapshots from one region to another re
failover_primary_compute	Fails over the primary compute unit of the specified Multi-AZ cluster t
get_cluster_credentials	Returns a database user name and temporary password with temporary
get_cluster_credentials_with_iam	Returns a database user name and temporary password with temporary
get_reserved_node_exchange_configuration_options	Gets the configuration options for the reserved-node exchange
get_reserved_node_exchange_offerings	Returns an array of DC2 ReservedNodeOfferings that matches the pay
get_resource_policy	Get the resource policy for a specified resource
list_recommendations	List the Amazon Redshift Advisor recommendations for one or multip
modify_aqua_configuration	This operation is retired
modify_authentication_profile	Modifies an authentication profile
modify_cluster	Modifies the settings for a cluster
modify_cluster_db_revision	Modifies the database revision of a cluster
modify_cluster_iam_roles	Modifies the list of Identity and Access Management (IAM) roles that
modify_cluster_maintenance	Modifies the maintenance settings of a cluster
modify_cluster_parameter_group	Modifies the parameters of a parameter group
modify_cluster_snapshot	Modifies the settings for a snapshot
modify_cluster_snapshot_schedule	Modifies a snapshot schedule for a cluster
modify_cluster_subnet_group	Modifies a cluster subnet group to include the specified list of VPC sub
modify_custom_domain_association	Contains information for changing a custom domain association
modify_endpoint_access	Modifies a Redshift-managed VPC endpoint
modify_event_subscription	Modifies an existing Amazon Redshift event notification subscription
modify_integration	Modifies a zero-ETL integration or S3 event integration with Amazon
modify_redshift_idc_application	Changes an existing Amazon Redshift IAM Identity Center application
modify_scheduled_action	Modifies a scheduled action
modify_snapshot_copy_retention_period	Modifies the number of days to retain snapshots in the destination Ama
modify_snapshot_schedule	Modifies a snapshot schedule
modify_usage_limit	Modifies a usage limit in a cluster
pause_cluster	Pauses a cluster
purchase_reserved_node_offering	Allows you to purchase reserved nodes
put_resource_policy	Updates the resource policy for a specified resource
reboot_cluster	Reboots a cluster
register_namespace	Registers a cluster or serverless namespace to the Amazon Web Servic
reject_data_share	From a datashare consumer account, rejects the specified datashare
reset_cluster_parameter_group	Sets one or more parameters of the specified parameter group to their d
resize_cluster	Changes the size of the cluster
restore_from_cluster_snapshot	Creates a new cluster from a snapshot
restore_table_from_cluster_snapshot	Creates a new table from a table in an Amazon Redshift cluster snapsh
resume_cluster	Resumes a paused cluster
revoke_cluster_security_group_ingress	Revokes an ingress rule in an Amazon Redshift security group for a pr
revoke_endpoint_access	Revokes access to a cluster
revoke_snapshot_access	Removes the ability of the specified Amazon Web Services account to
rotate_encryption_key	Rotates the encryption keys for a cluster
update_partner_status	Updates the status of a partner integration

Examples

```
## Not run:
svc <- redshift()
svc$accept_reserved_node_exchange(
  Foo = 123
)

## End(Not run)
```

---

redshiftdataapiservice

Redshift Data API Service

Description

You can use the Amazon Redshift Data API to run queries on Amazon Redshift tables. You can run SQL statements, which are committed if the statement succeeds.

For more information about the Amazon Redshift Data API and CLI usage examples, see Using the Amazon Redshift Data API in the *Amazon Redshift Management Guide*.

Usage

```
redshiftdataapiservice(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

`config` Optional configuration of credentials, endpoint, and/or region.

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.

- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

Value

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- redshiftdataapiservice(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

batch_execute_statement	Runs one or more SQL statements, which can be data manipulation language (DML) or data defini
cancel_statement	Cancels a running query
describe_statement	Describes the details about a specific instance when a query was run by the Amazon Redshift Data
describe_table	Describes the detailed information about a table from metadata in the cluster
execute_statement	Runs an SQL statement, which can be data manipulation language (DML) or data definition langua
get_statement_result	Fetches the temporarily cached result of an SQL statement in JSON format
get_statement_result_v2	Fetches the temporarily cached result of an SQL statement in CSV format
list_databases	List the databases in a cluster
list_schemas	Lists the schemas in a database
list_statements	List of SQL statements
list_tables	List the tables in a database

Examples

```
## Not run:
svc <- redshiftdataapiservice()
svc$batch_execute_statement(
  Foo = 123
)

## End(Not run)
```

redshiftserverless *Redshift Serverless*

Description

This is an interface reference for Amazon Redshift Serverless. It contains documentation for one of the programming or command line interfaces you can use to manage Amazon Redshift Serverless.

Amazon Redshift Serverless automatically provisions data warehouse capacity and intelligently scales the underlying resources based on workload demands. Amazon Redshift Serverless adjusts capacity in seconds to deliver consistently high performance and simplified operations for even the most demanding and volatile workloads. Amazon Redshift Serverless lets you focus on using your data to acquire new insights for your business and customers.

To learn more about Amazon Redshift Serverless, see What is Amazon Redshift Serverless?.

Usage

```
redshiftserverless(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

`config` Optional configuration of credentials, endpoint, and/or region.

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy `https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html`

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

Value

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- redshiftserverless(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

convert_recovery_point_to_snapshot	Converts a recovery point to a snapshot
create_custom_domain_association	Creates a custom domain association for Amazon Redshift Serverless
create_endpoint_access	Creates an Amazon Redshift Serverless managed VPC endpoint
create_namespace	Creates a namespace in Amazon Redshift Serverless
create_scheduled_action	Creates a scheduled action
create_snapshot	Creates a snapshot of all databases in a namespace
create_snapshot_copy_configuration	Creates a snapshot copy configuration that lets you copy snapshots to another Amazon
create_usage_limit	Creates a usage limit for a specified Amazon Redshift Serverless usage type

create_workgroup	Creates an workgroup in Amazon Redshift Serverless
delete_custom_domain_association	Deletes a custom domain association for Amazon Redshift Serverless
delete_endpoint_access	Deletes an Amazon Redshift Serverless managed VPC endpoint
delete_namespace	Deletes a namespace from Amazon Redshift Serverless
delete_resource_policy	Deletes the specified resource policy
delete_scheduled_action	Deletes a scheduled action
delete_snapshot	Deletes a snapshot from Amazon Redshift Serverless
delete_snapshot_copy_configuration	Deletes a snapshot copy configuration
delete_usage_limit	Deletes a usage limit from Amazon Redshift Serverless
delete_workgroup	Deletes a workgroup
get_credentials	Returns a database user name and temporary password with temporary authorization t
get_custom_domain_association	Gets information about a specific custom domain association
get_endpoint_access	Returns information, such as the name, about a VPC endpoint
get_namespace	Returns information about a namespace in Amazon Redshift Serverless
get_recovery_point	Returns information about a recovery point
get_resource_policy	Returns a resource policy
get_scheduled_action	Returns information about a scheduled action
get_snapshot	Returns information about a specific snapshot
get_table_restore_status	Returns information about a TableRestoreStatus object
get_usage_limit	Returns information about a usage limit
get_workgroup	Returns information about a specific workgroup
list_custom_domain_associations	Lists custom domain associations for Amazon Redshift Serverless
list_endpoint_access	Returns an array of EndpointAccess objects and relevant information
list_managed_workgroups	Returns information about a list of specified managed workgroups in your account
list_namespaces	Returns information about a list of specified namespaces
list_recovery_points	Returns an array of recovery points
list_scheduled_actions	Returns a list of scheduled actions
list_snapshot_copy_configurations	Returns a list of snapshot copy configurations
list_snapshots	Returns a list of snapshots
list_table_restore_status	Returns information about an array of TableRestoreStatus objects
list_tags_for_resource	Lists the tags assigned to a resource
list_usage_limits	Lists all usage limits within Amazon Redshift Serverless
list_workgroups	Returns information about a list of specified workgroups
put_resource_policy	Creates or updates a resource policy
restore_from_recovery_point	Restore the data from a recovery point
restore_from_snapshot	Restores a namespace from a snapshot
restore_table_from_recovery_point	Restores a table from a recovery point to your Amazon Redshift Serverless instance
restore_table_from_snapshot	Restores a table from a snapshot to your Amazon Redshift Serverless instance
tag_resource	Assigns one or more tags to a resource
untag_resource	Removes a tag or set of tags from a resource
update_custom_domain_association	Updates an Amazon Redshift Serverless certificate associated with a custom domain
update_endpoint_access	Updates an Amazon Redshift Serverless managed endpoint
update_namespace	Updates a namespace with the specified settings
update_scheduled_action	Updates a scheduled action
update_snapshot	Updates a snapshot
update_snapshot_copy_configuration	Updates a snapshot copy configuration
update_usage_limit	Update a usage limit in Amazon Redshift Serverless
update_workgroup	Updates a workgroup with the specified configuration settings

Examples

```
## Not run:
svc <- redshiftserverless()
svc$convert_recovery_point_to_snapshot(
  Foo = 123
)

## End(Not run)
```

---

simpledb *Amazon SimpleDB*

Description

Amazon SimpleDB is a web service providing the core database functions of data indexing and querying in the cloud. By offloading the time and effort associated with building and operating a web-scale database, SimpleDB provides developers the freedom to focus on application development.

A traditional, clustered relational database requires a sizable upfront capital outlay, is complex to design, and often requires extensive and repetitive database administration. Amazon SimpleDB is dramatically simpler, requiring no schema, automatically indexing your data and providing a simple API for storage and access. This approach eliminates the administrative burden of data modeling, index maintenance, and performance tuning. Developers gain access to this functionality within Amazon's proven computing environment, are able to scale instantly, and pay only for what they use.

Visit http://aws.amazon.com/simpledb/ for more information.

Usage

```
simpledb(config = list(), credentials = list(), endpoint = NULL, region = NULL)
```

Arguments

`config` Optional configuration of credentials, endpoint, and/or region.

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.

- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy `https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html`

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

Value

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- simpledb(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

batch_delete_attributes	Performs multiple DeleteAttributes operations in a single call, which reduces round trips and latencie
batch_put_attributes	The BatchPutAttributes operation creates or replaces attributes within one or more items
create_domain	The CreateDomain operation creates a new domain
delete_attributes	Deletes one or more attributes associated with an item
delete_domain	The DeleteDomain operation deletes a domain
domain_metadata	Returns information about the domain, including when the domain was created, the number of items
get_attributes	Returns all of the attributes associated with the specified item
list_domains	The ListDomains operation lists all domains associated with the Access Key ID
put_attributes	The PutAttributes operation creates or replaces attributes in an item
select	The Select operation returns a set of attributes for ItemNames that match the select expression

Examples

```
## Not run:
svc <- simpledb()
svc$batch_delete_attributes(
  Foo = 123
)

## End(Not run)
```

`timestreamquery` *Amazon Timestream Query*

Description

Amazon Timestream Query

Usage

```
timestreamquery(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

`config` Optional configuration of credentials, endpoint, and/or region.

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy `https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html`

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

Value

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- timestreamquery(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

cancel_query	Cancels a query that has been issued
create_scheduled_query	Create a scheduled query that will be run on your behalf at the configured schedule
delete_scheduled_query	Deletes a given scheduled query
describe_account_settings	Describes the settings for your account that include the query pricing model and the configured m
describe_endpoints	DescribeEndpoints returns a list of available endpoints to make Timestream API calls against
describe_scheduled_query	Provides detailed information about a scheduled query
execute_scheduled_query	You can use this API to run a scheduled query manually
list_scheduled_queries	Gets a list of all scheduled queries in the caller's Amazon account and Region
list_tags_for_resource	List all tags on a Timestream query resource
prepare_query	A synchronous operation that allows you to submit a query with parameters to be stored by Time
query	Query is a synchronous operation that enables you to run a query against your Amazon Timestrea
tag_resource	Associate a set of tags with a Timestream resource
untag_resource	Removes the association of tags from a Timestream query resource

update_account_settings	Transitions your account to use TCUs for query pricing and modifies the maximum query compu
update_scheduled_query	Update a scheduled query

Examples

```
## Not run:
svc <- timestreamquery()
svc$cancel_query(
  Foo = 123
)

## End(Not run)
```

timestreamwrite *Amazon Timestream Write*

Description

Amazon Timestream is a fast, scalable, fully managed time-series database service that makes it easy to store and analyze trillions of time-series data points per day. With Timestream, you can easily store and analyze IoT sensor data to derive insights from your IoT applications. You can analyze industrial telemetry to streamline equipment management and maintenance. You can also store and analyze log data and metrics to improve the performance and availability of your applications.

Timestream is built from the ground up to effectively ingest, process, and store time-series data. It organizes data to optimize query processing. It automatically scales based on the volume of data ingested and on the query volume to ensure you receive optimal performance while inserting and querying data. As your data grows over time, Timestream's adaptive query processing engine spans across storage tiers to provide fast analysis while reducing costs.

Usage

```
timestreamwrite(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

`config` Optional configuration of credentials, endpoint, and/or region.

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID

        - **secret_access_key**: AWS secret access key
        - **session_token**: AWS temporary session token
    - **profile**: The name of a profile to use. If not given, then the default profile is used.
    - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

Value

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- timestreamwrite(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

create_batch_load_task	Creates a new Timestream batch load task
create_database	Creates a new Timestream database
create_table	Adds a new table to an existing database in your account
delete_database	Deletes a given Timestream database
delete_table	Deletes a given Timestream table
describe_batch_load_task	Returns information about the batch load task, including configurations, mappings, progress, and
describe_database	Returns information about the database, including the database name, time that the database was c
describe_endpoints	Returns a list of available endpoints to make Timestream API calls against
describe_table	Returns information about the table, including the table name, database name, retention duration o
list_batch_load_tasks	Provides a list of batch load tasks, along with the name, status, when the task is resumable until, a
list_databases	Returns a list of your Timestream databases
list_tables	Provides a list of tables, along with the name, status, and retention properties of each table
list_tags_for_resource	Lists all tags on a Timestream resource
resume_batch_load_task	Resume batch load task
tag_resource	Associates a set of tags with a Timestream resource
untag_resource	Removes the association of tags from a Timestream resource
update_database	Modifies the KMS key for an existing database
update_table	Modifies the retention duration of the memory store and magnetic store for your Timestream table
write_records	Enables you to write your time-series data into Timestream

Examples

```
## Not run:
svc <- timestreamwrite()
svc$create_batch_load_task(
  Foo = 123
)

## End(Not run)
```

Index