AWS DOCUMENTATION

Filter

aws provider
> Guides
> ACM
> ACM PCA
> API Gateway (REST APIs)
> API Gateway v2 (WebSocket and HTTP APIs)
> Access Analyzer
> Account
> Amazon Managed Service for Prometheus (AMP)
> Amplify Console
> App Runner
> AppConfig
> AppMesh
> AppStream
> AppSync
> Application Autoscaling
> Athena
> Autoscaling
> Autoscaling Plans
> Backup
> Batch
> Budgets

Resource: aws_dynamodb_table

Provides a DynamoDB table resource

Note: It is recommended to use `lifecycle` `ignore_changes` for `read_capacity` and/or `write_capacity` if there's autoscaling policy attached to the table.

Example Usage

The following dynamodb table description models the table and GSI shown in the AWS SDK example documentation

```
resource "aws_dynamodb_table" "basic-dyna
  name           = "GameScores"
  billing_mode   = "PROVISIONED"
  read_capacity  = 20
  write_capacity = 20
  hash_key       = "UserId"
  range_key      = "GameTitle"

  attribute {
    name = "UserId"
    type = "S"
  }

  attribute {
    name = "GameTitle"
```

```
    type = "S"
  }

  attribute {
    name = "TopScore"
    type = "N"
  }

  ttl {
    attribute_name = "TimeToExist"
    enabled        = false
  }

  global_secondary_index {
    name               = "GameTitleIndex"
    hash_key           = "GameTitle"
    range_key          = "TopScore"
    write_capacity     = 10
    read_capacity      = 10
    projection_type    = "INCLUDE"
    non_key_attributes = ["UserId"]
  }

  tags = {
    Name        = "dynamodb-table-1"
    Environment = "production"
  }
}
```

Global Tables

This resource implements support for DynamoDB Global Tables V2 (version 2019.11.21) via `replica` configuration blocks. For working with DynamoDB Global Tables V1 (version 2017.11.29), see the `aws_dynamodb_global_table` resource.

```
resource "aws_dynamodb_table" "example" {
  name             = "example"
  hash_key         = "TestTableHashKey"
  billing_mode     = "PAY_PER_REQUEST"
  stream_enabled   = true
  stream_view_type = "NEW_AND_OLD_IMAGES"

  attribute {
    name = "TestTableHashKey"
```

```
    type = "S"
  }

  replica {
    region_name = "us-east-2"
  }

  replica {
    region_name = "us-west-2"
  }
}
```

Argument Reference

The following arguments are supported:

- `name` - (Required) The name of the table, this needs to be unique within a region.
- `billing_mode` - (Optional) Controls how you are charged for read and write throughput and how you manage capacity. The valid values are `PROVISIONED` and `PAY_PER_REQUEST`. Defaults to `PROVISIONED`.
- `hash_key` - (Required, Forces new resource) The attribute to use as the hash (partition) key. Must also be defined as an `attribute`, see below.
- `range_key` - (Optional, Forces new resource) The attribute to use as the range (sort) key. Must also be defined as an `attribute`, see below.
- `write_capacity` - (Optional) The number of write units for this table. If the `billing_mode` is `PROVISIONED`, this field is required.
- `read_capacity` - (Optional) The number of read units for this table. If the `billing_mode` is `PROVISIONED`, this field is required.
- `attribute` - (Required) List of nested attribute definitions. Only required for

`hash_key` and `range_key` attributes. Each attribute has two properties:

  - `name` - (Required) The name of the attribute
  - `type` - (Required) Attribute type, which must be a scalar type: `S`, `N`, or `B` for (S)tring, (N)umber or (B)inary data

- `ttl` - (Optional) Defines ttl, has two properties, and can only be specified once:
  - `enabled` - (Required) Indicates whether ttl is enabled (true) or disabled (false).
  - `attribute_name` - (Required) The name of the table attribute to store the TTL timestamp in.

- `local_secondary_index` - (Optional, Forces new resource) Describe an LSI on the table; these can only be allocated *at creation* so you cannot change this definition after you have created the resource.
- `global_secondary_index` - (Optional) Describe a GSI for the table; subject to the normal limits on the number of GSIs, projected attributes, etc.
- `replica` - (Optional) Configuration block(s) with DynamoDB Global Tables V2 (version 2019.11.21) replication configurations. Detailed below.
- `stream_enabled` - (Optional) Indicates whether Streams are to be enabled (true) or disabled (false).
- `stream_view_type` - (Optional) When an item in the table is modified, StreamViewType determines what information is written to the table's stream. Valid values are `KEYS_ONLY`, `NEW_IMAGE`, `OLD_IMAGE`, `NEW_AND_OLD_IMAGES`.

- `server_side_encryption` - (Optional) Encryption at rest options. AWS DynamoDB tables are automatically encrypted at rest with an AWS owned Customer Master Key if this argument isn't specified.
- `tags` - (Optional) A map of tags to populate on the created table. If configured with a provider `default_tags` configuration block present, tags with matching keys will overwrite those defined at the provider-level.
- `point_in_time_recovery` - (Optional) Point-in-time recovery options.
- `table_class` - (Optional) The storage class of the table. Valid values are `STANDARD` and `STANDARD_INFREQUENT_ACCESS`.

Timeouts

The `timeouts` block allows you to specify timeouts for certain actions:

- `create` - (Defaults to 10 mins) Used when creating the table
- `update` - (Defaults to 60 mins) Used when updating the table configuration and reset for each individual Global Secondary Index and Replica update
- `delete` - (Defaults to 10 mins) Used when deleting the table

Nested fields

`local_secondary_index`

- `name` - (Required) The name of the index
- `range_key` - (Required) The name of the range key; must be defined

- `projection_type` - (Required) One of `ALL`, `INCLUDE` or `KEYS_ONLY` where `ALL` projects every attribute into the index, `KEYS_ONLY` projects just the hash and range key into the index, and `INCLUDE` projects only the keys specified in the *non_key_attributes* parameter.
- `non_key_attributes` - (Optional) Only required with `INCLUDE` as a projection type; a list of attributes to project into the index. These do not need to be defined as attributes on the table.

`global_secondary_index`

- `name` - (Required) The name of the index
- `write_capacity` - (Optional) The number of write units for this index. Must be set if billing_mode is set to PROVISIONED.
- `read_capacity` - (Optional) The number of read units for this index. Must be set if billing_mode is set to PROVISIONED.
- `hash_key` - (Required) The name of the hash key in the index; must be defined as an attribute in the resource.
- `range_key` - (Optional) The name of the range key; must be defined
- `projection_type` - (Required) One of `ALL`, `INCLUDE` or `KEYS_ONLY` where `ALL` projects every attribute into the index, `KEYS_ONLY` projects just the hash and range key into the index, and `INCLUDE` projects only the keys specified in the *non_key_attributes* parameter.
- `non_key_attributes` - (Optional) Only required with `INCLUDE` as a projection type; a list of attributes to project into the index.

These do not need to be defined as attributes on the table.

`replica`

The `replica` configuration block supports the following arguments:

- `region_name` - (Required) Region name of the replica.
- `kms_key_arn` - (Optional) The ARN of the CMK that should be used for the AWS KMS encryption.

`server_side_encryption`

- `enabled` - (Required) Whether or not to enable encryption at rest using an AWS managed KMS customer master key (CMK).
- `kms_key_arn` - (Optional) The ARN of the CMK that should be used for the AWS KMS encryption. This attribute should only be specified if the key is different from the default DynamoDB CMK, `alias/aws/dynamodb`.

If `enabled` is `false` then server-side encryption is set to AWS owned CMK (shown as `DEFAULT` in the AWS console). If `enabled` is `true` and no `kms_key_arn` is specified then server-side encryption is set to AWS managed CMK (shown as `KMS` in the AWS console). The AWS KMS documentation explains the difference between AWS owned and AWS managed CMKs.

`point_in_time_recovery`

- `enabled` - (Required) Whether to enable point-in-time recovery - note that it can take up to 10 minutes to enable for new tables. If

the `point_in_time_recovery` block is not provided then this defaults to `false`.

A note about attributes

Only define attributes on the table object that are going to be used as:

- Table hash key or range key
- LSI or GSI hash key or range key

The DynamoDB API expects attribute structure (name and type) to be passed along when creating or updating GSI/LSIs or creating the initial table. In these cases it expects the Hash / Range keys to be provided; because these get re-used in numerous places (i.e the table's range key could be a part of one or more GSIs), they are stored on the table object to prevent duplication and increase consistency. If you add attributes here that are not used in these scenarios it can cause an infinite loop in planning.

Attributes Reference

In addition to all arguments above, the following attributes are exported:

- `arn` - The arn of the table
- `id` - The name of the table
- `stream_arn` - The ARN of the Table Stream. Only available when `stream_enabled = true`
- `stream_label` - A timestamp, in ISO 8601 format, for this stream. Note that this timestamp is not a unique identifier for the stream on its own. However, the combination of AWS customer ID, table name and this field is guaranteed to be unique. It can be used for

creating CloudWatch Alarms. Only available when `stream_enabled = true`

- `tags_all` - A map of tags assigned to the resource, including those inherited from the provider `default_tags` configuration block.

Import

DynamoDB tables can be imported using the `name`, e.g.,

```
$ terraform import aws_dynamodb_table.basic-dy
```