AWS CLI Cheatsheet

Config

Create profiles

```
aws configure --profile profilename
```

Output format

```
aws configure output format {json, yaml, yaml-stream, text, table}
```

Specify your AWS Region

```
aws configure region (region-name)
```

API Gateway

List API Gateway IDs and Names

```
aws apigateway get-rest-apis | jq -r '.items[ ] | .id+" "+.name'
```

List API Gateway keys

```
aws apigateway get-api-keys | jq -r '.items[ ] | .id+" "+.name'
```

List API Gateway domain names

```
aws apigateway get-domain-names | jq -r '.items[ ] |
.domainName+" "+.regionalDomainName'
```

List resources for API Gateway

```
aws apigateway get-resources --rest-api-id ee86b4cde | jq -r
'.items[ ] | .id+" "+.path'
```

Find Lambda for API Gateway resource

```
aws apigateway get-integration --rest-api-id (id) --resource-id
(resource id) --http-method GET | jq -r '.uri'
```

Amplify

List Amplify apps and source repository

```
aws amplify list-apps | jq -r '.apps[ ] | .name+"
"+.defaultDomain+"
```

CloudFront

List CloudFront distributions and origins

```
aws cloudfront list-distributions | jq -r
'.DistributionList.Items[ ] | .DomainName+"
"+.Origins.Items[0].DomainName'
```

Create a new invalidation

```
aws cloudfront create-invalidation [distribution-id]
```

CloudWatch

List information about an alarm

```
aws cloudwatch describe-alarms | jq -r '.MetricAlarms[ ] |
.AlarmName+" "+.Namespace+" "+.StateValue'
```

Delete an alarm or alarms (you can delete up to 100 at a time)

```
aws cloudwatch delete-alarms --alarm-names (alarmnames)
```

Cognito

List user pool IDs and names

```
aws cognito-idp list-user-pools --max-results 60 | jq -r
'.UserPools[ ] | .Id+" "+.Name'
```

List phone and email of all users

```
aws cognito-idp list-users --user-pool-id (resource) | jq -r
'.Users[ ].Attributes | from_entries | .sub + " " +
.phone_number + " " + .email'
```

DynamoDB

List DynamoDB tables

```
aws dynamodb list-tables | jq -r .TableNames [ ]
```

Get all items from a table

```
aws dynamodb scan --table-name events
```

Get item count from a table

```
aws dynamodb scan --table-name events --select count | jq
.ScannedCount
```

Get item using key

```
aws dynamodb get-item --table-name events --key
'{“email””"email@example.com”}}'
```

Get specific fields from an item

```
aws dynamodb get-item --table-name events --key
'{“email””"email@example.com"}}' --attributes-to-get event_type
```

Delete item using key

```
aws dynamodb delete-item --table-name events --key
'{"email""email@domain.com"}}'
```

EBS

Complete a Snapshot

```
aws ebs complete-snapshot (snapshot-id)
```

Start a Snapshot

```
aws ebs start-snapshot --volume-size (value)
```

Get a Snapshot block

```
aws ebs get-snapshot-block
--snapshot-id (value)
--block-index (value)
--block-token (value)
```

EC2

List Instance ID, Type and Name

```
aws ec2 describe-instances | jq -r
'.Reservations[].Instances[]|.InstanceId+" "+.InstanceType+"
"+(.Tags[] | select(.Key == "Name").Value)'
```

List Instances with public IP address and Name

```
aws ec2 describe-instances --query
'Reservations[*].Instances[?not_null(PublicIpAddress)]' | jq -r
'.[][]|.PublicIpAddress+"
"+(.Tags[]|select(.Key=="Name").Value)'
```

List VPCs and CIDR IP Block

```
aws ec2 describe-vpcs | jq -r '.Vpcs[]|.VpcId+"
"+(.Tags[]|select(.Key=="Name").Value)+" "+.CidrBlock'
```

List Subnets for a VPC

```
aws ec2 describe-subnets --filter Name=vpc-id,Values=vpc-
0d1c1cf4e980ac593 | jq -r '.Subnets[]|.SubnetId+" "+.CidrBlock+"
"+(.Tags[]|select(.Key=="Name").Value)'
```

List Security Groups

```
aws ec2 describe-security-groups | jq -r
'.SecurityGroups[]|.GroupId+" "+.GroupName'
```

Print Security Groups for an Instance

```
aws ec2 describe-instances --instance-ids i-0dae5d4daa47fe4a2 |
jq -r '.Reservations[].Instances[].SecurityGroups[]|.GroupId+"
"+.GroupName'
```

Edit Security Groups of an Instance

```
aws ec2 modify-instance-attribute --instance-id i-0dae5d4daa47fe4a2 --groups sg-02a63c67684d8deed sg-0dae5d4daa47fe4a2
```

Print Security Group Rules as FromAddress and ToPort

```
aws ec2 describe-security-groups --group-ids sg-02a63c67684d8deed | jq -r '.SecurityGroups[].IpPermissions[]|. as $parent|(.IpRanges[].CidrIp+" "+($parent.ToPort|tostring))'
```

Add Rule to Security Group

```
aws ec2 authorize-security-group-ingress --group-id sg-02a63c67684d8deed --protocol tcp --port 443 --cidr 35.0.0.1
```

Delete Rule from Security Group

```
aws ec2 revoke-security-group-ingress --group-id sg-02a63c67684d8deed --protocol tcp --port 443 --cidr 35.0.0.1
```

Edit Rules of Security Group

```
aws ec2 update-security-group-rule-descriptions-ingress --group-id sg-02a63c67684d8deed --ip-permissions 'ToPort=443,IpProtocol=tcp,IpRanges=[{CidrIp=202.171.186.133/32,Description=Home}]'
```

Delete Security Group

```
aws ec2 delete-security-group --group-id sg-02a63c67684d8deed
```

ECS

Create an ECS cluster

```
aws ecs create-cluster --cluster-name=NAME --generate-cli-skeleton
```

Create an ECS service

```
aws ecs create-service
```

EKS

Create a cluster

```
aws eks create-cluster --name (cluster name)
```

Delete a cluster

```
aws eks delete-cluster --name (cluster name)
```

List descriptive information about a cluster

```
aws eks describe-cluster --name (cluster name)
```

List clusters in your default region

```
aws eks list-clusters
```

Tag a resource

```
aws eks tag-resource --resource-arn (resource_ARN) --tags (tags)
```

Untag a resource

```
aws eks untag-resource --resource-arn (resource_ARN) --tag-keys
(tag-key)
```

ElastiCache

Get information about a specific cache cluster

```
aws elasticache describe-cache-clusters | jq -r '.CacheClusters[
] | .CacheNodeType+" "+.CacheClusterId'
```

List ElastiCache replication groups

```
aws elasticache describe-replication-groups | jq -r
'.ReplicationGroups [ ] | .ReplicationGroupId+" "+.NodeGroups[
].PrimaryEndpoint.Address'
```

List ElastiCache snapshots

```
aws elasticache describe-snapshots | jq -r '.Snapshots[ ] |
.SnapshotName'
```

Create ElastiCache snapshot

```
aws elasticache create-snapshot --snapshot-name backend-login-hk-snap-1 --replication-group-id backend-login-hk --cache-cluster-id backend-login-hk
```

Delete ElastiCache snapshot

```
aws elasticache delete-snapshot --snapshot-name login-snap-1
```

Scale up/down ElastiCache replica

```
aws elasticache increase-replica-count --replication-group-id backend-login --apply-immediately
```

```
aws elasticache decrease-replica-count --replication-group-id backend-login --apply-immediately
```

ELB

List ELB Hostnames

```
aws elbv2 describe-load-balancers --query 'LoadBalancers[*].DNSName' | jq -r 'to_entries[ ] | .value'
```

List ELB ARNs

```
aws elbv2 describe-load-balancers | jq -r '.LoadBalancers[ ] | .LoadBalancerArn'
```

List of ELB target group ARNs

```
aws elbv2 describe-target-groups | jq -r '.TargetGroups[ ] | .TargetGroupArn'
```

Find instances for a target group

```
aws elbv2 describe-target-health --target-group-arn arn:aws:elasticloadbalancing:ap-northwest-1:20394823094:targetgroup/wordpress-ph/203942b32a23 | jq -r '.TargetHealthDescriptions[ ] | .Target.Id'
```

IAM Group

List groups

```
aws iam list-groups | jq -r .Groups[ ].GroupName
```

Add/Delete groups

```
aws iam create-group --group-name (groupName)
```

List policies and ARNs

```
aws iam list-policies | jq -r '.Policies[ ]|.PolicyName+" "+.Arn'
```

```
aws iam list-policies --scope AWS | jq -r '.Policies[ ]|.PolicyName+" "+.Arn'
```

```
aws iam list-policies --scope Local | jq -r '.Policies[
]|.PolicyName+" "+.Arn'
```

List user/group/roles for a policy

```
aws iam list-entities-for-policy --policy-arn
arn:aws:iam:2308345:policy/example-ReadOnly
```

List policies for a group

```
aws iam list-attached-group-policies --group-name (groupname)
```

Add policy to a group

```
aws iam attach-group-policy --group-name (groupname) --policy-
arn arn:aws:iam::aws:policy/exampleReadOnlyAccess
```

Add user to a group

```
aws iam add-user-to-group --group-name (groupname) --user-name
(username)
```

Remove user from a group

```
aws iam remove-user-from-group --group-name (groupname) --user-
name (username)
```

List users in a group

```
aws iam get-group --group-name (groupname)
```

List groups for a user

```
aws iam list-groups-for-user --user-name (username)
```

Attach/detach policy to a group

```
aws iam attach-group-policy --group-name (groupname) --policy-arn arn:aws:iam::aws:policy/DynamoDBFullAccess
```

```
aws iam detach-group-policy --group-name (groupname) --policy-arn arn:aws:iam::aws:policy/DynamoDBFullAccess
```

IAM User

List userId and UserName

```
aws iam list-users | jq -r '.Users[ ]|.UserId+" "+.UserName'
```

Get single user

```
aws iam get-user --user-name (username)
```

Add user

```
aws iam create-user --user-name (username)
```

Delete user

```
aws iam delete-user --user-name (username)
```

List access keys for user

```
aws iam list-access-keys --user-name (username) | jq -r
.AccessKeyMetadata[ ].AccessKeyId
```

Delete access key for user

```
aws iam delete-access-key --user-name (username) --access-key-id
(accessKeyID)
```

Activate/deactivate access key for user

```
aws iam update-access-key --status Active --user-name (username)
--access-key-id (access key)
```

```
aws iam update-access-key --status Inactive --user-name
(username) --access-key-id (access key)
```

Generate new access key for user

```
aws iam create-access-key --user-name (username) | jq -r
'.AccessKey | .AccessKeyId+" "+.SecretAccessKey'
```

Lambda

List Lambda functions, runtime, and memory

```
aws lambda list-functions | jq -r '.Functions[ ] |
.FunctionName+" "+.Runtime+" "+(.MemorySize|tostring)'
```

List Lambda layers

```
aws lambda list-layers | jq -r '.Layers[ ] | .LayerName'
```

List source event for Lambda

```
aws lambda list-event-source-mappings | jq -r
'.EventSourceMappings[ ] | .FunctionArn+" "+.EventSourceArn'
```

Download Lambda code

```
aws lambda get-function --function-name DynamoToSQS | jq -r
.Code.Location
```

RDS

List DB clusters

```
aws rds describe-db-clusters | jq -r '.DBClusters[ ] |
.DBClusterIdentifier+" "+.Endpoint'
```

List DB instances

```
aws rds describe-db-instances | jq -r '.DBInstances[ ] |
.DBInstanceIdentifier+" "+.DBInstanceClass+"
"+.Endpoint.Address'
```

Take DB Instance Snapshot

```
aws rds create-db-snapshot --db-snapshot-identifier snapshot-1 -
-db-instance-identifier dev-1
```

```
aws rds describe-db-snapshots --db-snapshot-identifier snapshot-
1 --db-instance-identifier general
```

Take DB cluster snapshot

```
aws rds create-db-cluster-snapshot --db-cluster-snapshot-
identifier
```

Route53

Create hosted zone

```
aws route53 create-hosted-zone --name exampledomain.com
```

Delete hosted zone

```
aws route53 delete-hosted-zone --id example
```

Get hosted zone

```
aws route53 get-hosted-zone --id example
```

List hosted zones

```
aws route53 list-hosted-zones
```

Create a record set

To do this you'll first need to create a JSON file with a list of change items in the body and use the CREATE action. For example the JSON file would look like this.

```
{
     "Comment": "CREATE/DELETE/UPSERT a record",
     "Changes": [{
     "Action": "CREATE",
          "ResourceRecordSet":{
                    "Name": "a.example.com",
                    "Type": "A",
                    "TTL": 300,
          "ResourceRecords":[{"Value":"4.4.4.4"}]
}}]
}
```

Once you have a JSON file with the correct information like above you will be able to enter the command

```
aws route53 change-resource-record-sets --hosted-zone-id (zone-id) --change-batch file://exampleabove.json
```

Update a record set

To do this you'll first need to create a JSON file with a list of change items in the body and use the UPSERT action. This will either create a new record set with the specified value, or updates a record set if it already exists. For example the JSON file would look like this.

```
{
      "Comment": "CREATE/DELETE/UPSERT a record",
      "Changes": [{
      "Action": "UPSERT",
            "ResourceRecordSet":{
                  "Name": "a.example.com",
                  "Type": "A",
                  "TTL": 300,
            "ResourceRecords": [{"Value":"4.4.4.4"}]
}}]
}
```

Once you have a JSON file with the correct information like above you will be able to enter the command

```
aws route53 change-resource-record-sets --hosted-zone-id (zone-id) --change-batch file://exampleabove.json
```

Delete a record set

To do this you'll first need to create a JSON file with a list of the record set values you want to delete in the body and use the DELETE action. For example the JSON file would look like this.

```
{
      "Comment": "CREATE/DELETE/UPSERT a record",
      "Changes": [{
      "Action": "DELETE",
            "ResourceRecordSet": {
                  "Name": "a.example.com",
                  "Type": "A",
                  "TTL": 300,
            "ResourceRecords": [{"Value":"4.4.4.4"}]
}}]
}
```

Once you have a JSON file with the correct information like above you will be able to enter the following command.

```
aws route53 change-resource-record-sets --hosted-zone-id (zone-id) --change-batch file://exampleabove.json
```

S3

List Buckets

```
aws s3 ls
```

List files in a Bucket

```
aws s3 ls s3://mybucket
```

Create Bucket

```
aws s3 mb s3://bucket-name
make_bucket: bucket-name
```

Delete Bucket

```
aws s3 rb s3://bucket-name --force
```

Download S3 object to local

```
aws s3 cp s3://bucket-name
```

```
download: ./backup.tar from s3://bucket-name/backup.tar
```

Upload local file as S3 object

```
aws s3 cp backup.tar s3://bucket-name
upload: ./backup.tar to s3://bucket-name/backup.tar
```

Delete S3 object

```
aws s3 rm s3://bucket-name/secret-file.gz .
delete: s3://bucket-name/secret-file.gz
```

Download bucket to local

```
aws s3 sync s3://bucket-name/ /media/pasport-ultra/backup
```

Upload local directory to bucket

```
aws s3 sync (directory) s3://bucket-name/
```

Share S3 object without public access

```
aws s3 presign s3://bucket-name/file-name --expires-in (time
value)
```

```
https://bucket-name.s3.amazonaws.com/file-
name.pdf?AWSAccessKeyId=(key)&Expires=(value)&Signature=(value)
```

SNS

List SNS topics

```
aws sns list-topics | jq -r '.Topics[ ] | .TopicArn'
```

List SNS topic and related subscriptions

```
aws sns list-subscriptions | jq -r '.Subscriptions[ ] |
.TopicArn+" "+.Protocol+" "+.Endpoint'
```

Publish to SNS topic

```
aws sns publish --topic-arn arn:aws:sns:ap-southeast-
1:232398:backend-api-monitoring
```

SQS

List queues

```
aws sqs list-queues | jq -r '.QueueUrls[ ]'
```

Create queue

```
aws sqs create-queue --queue-name public-events.fifo | jq -r
.queueURL
```

Send message

```
aws sqs send-message --queue-url (url) --message-body (message)
```

Receive message

```
aws sqs receive-message --queue-url (url) | jq -r '.Messages[ ]
| .Body'
```

Delete message

```
aws sqs delete-message --queue url (url) --receipt-handle
(receipt handle)
```

Purge queue

```
aws sqs purge-queue --queue-url (url)
```

Delete queue

```
aws sqs delete-queue --queue-url (url)
```