弹性数据库 API 接口文档

一、数据库接口

1. 数据库列表(keyspaces list)

Method	GET
URL	/api/keyspaces/
响应参数	content-type:application/json; ["gw_keyspace"]

说明： 接口返回所有可用逻辑库列表的名称， 这块后续需要考虑根据用户授权返回。 只返回用户有权限访问的逻辑库列表

2. 创建数据库

Method	POST
URL	/api/vtctl/
请求参数	content-type:application/json; ["CreateKeyspace", "gwgggg"]

响应参数	content-type:application/json;
	{
	"Error": "",
	"Output": ""
	}
	
	Error: 成功: 空字符串 失败: 对应的错误信息
	Output: 对应的输出信息

说明：接口使用通用接口，参数通过在 CreateKeyspace 后面增加需要创建逻辑库的名称来实现。

3. 删除数据库

Method	POST
URL	/api/vtctl/
请求参数	content-type:application/json; ["DeleteKeyspace", "-recursive", "gwgggg"]
响应参数	content-type:application/json; { "Error": "",

	<pre>{
	"Output": ""
	}</pre>
	Error: 成功: 空字符串 失败: 对应的错误信息
	Output: 对应的输出信息

说明： 接口使用通用接口， 参数通过在 DeleteKeyspace 后面增加需要删除逻辑库的名称和对应的参数来实现：

-recursive： 递归删除，删除所有相关的数据

4. 数据库验证

接口提供数据库有效性验证

Method	POST
URL	/api/vtctl/

请求参数	content-type:application/json;
	有效性验证有如下几种类型:
	1.
	<pre>[
	"ValidateKeyspace",
	"-ping-tablets",
	"gw_keyspace" //逻辑库名称
	]</pre>
	2.
	<pre>[
	"ValidateSchemaKeyspace",
	"gw_keyspace" //逻辑库名称
	]</pre>
	3.
	<pre>[
	"ValidateVersionKeyspace",
	"test" //逻辑库名称
	]</pre>

响应参数	content-type:application/json;
	<pre>{
	"Error": "",
	"Output": ""
	}</pre>
	Error: 成功: 空字符串 失败: 对应的错误信息
	Output: 对应的输出信息

说明： 提供数据库有效性验证，包含逻辑库验证，逻辑库下的 Tablets 验证，版本验证

5. RebuildKeyspaceGraph

Method	POST
URL	/api/vtctl/
请求参数	content-type:application/json; ["RebuildKeyspaceGraph", "gw_keyspace"]

响应参数	content-type:application/json;
	{
	"Error": "",
	"Output": ""
	}
	
	Error: 成功：空字符串 失败：对应的错误信息
	Output: 对应的输出信息

6. 创建分片

Method	POST
URL	/api/vtctl/
请求参数	content-type:application/json; ["CreateShard", "test/-10"]

响应参数	content-type:application/json;
	{
	"Error": "",
	"Output": ""
	}
	
	Error: 成功：空字符串 失败：对应的错误信息
	Output: 对应的输出信息

说明： 使用通用接口， 根据参数 CreateShard 创建对应的分片

7. 删除分片

Method	POST
URL	/api/vtctl/
请求参数	content-type:application/json ["DeleteShard",

	<pre>"-recursive",
	"-even_if_serving",
	"test/-10"
	]</pre>

响应参数	content-type:application/json;
	<pre>{
	"Error": "",
	"Output": ""
	}</pre>
	Error：成功：空字符串　失败：对应的错误信息
	Output：对应的输出信息

8. 获取分片列表

Method	GET
URL	/api/shards/{keyspace}/

响应参数	content-type:application/json;
	<pre>[
	"-10",
	"10-"
	]</pre>

9. 获取 Tablets 列表

Method	POST
URL	/api/tablets/
请求参数	content-type:application/x-www-form-urlencoded shard=gw_keyspace/-10

响应参数	content-type:application/json;
	<pre>[
	{
	"cell": "test",
	"uid": 1201
	},
	{
	"cell": "test",
	"uid": 1202
	},
	{
	"cell": "test",
	"uid": 1203
	},
	{
	"cell": "test",
	"uid": 1204
	}</pre>

	]

10. 获取 Tablets 中表信息

Method	POST
URL	/api/vtctl/
请求参数	content-type:application/json; ["GetSchema", "test-1201"]

响应参数	content-type:application/json;
	{
	"Error": "",
	"Output": "xxxxxxx"
	}
	
	Error: 成功: 空字符串 失败: 对应的错误信息
	Output: 对应的输出信息

11. 执行 sql

Method	POST
URL	/api/schema/apply

请求参数	content-type:application/json
	{
	"Keyspace": "gw_keyspace",
	"SQL": "select now()"
	}
	
	Keyspace: 需要操作的逻辑库
	SQL: 需要执行的 sql

响应参数	content-type:application/json;
	{
	"Error": "",
	"Output": "xxxxxxx"
	}
	
	Error: 成功: 空字符串 失败: 对应的错误信息
	Output: 对应的输出信息

12. 初始化 Master

Method	POST
URL	/api/vtctl/
请求参数	content-type:application/json; ["InitShardMaster",

	"-force", "gw_keyspace/10-", "test-001"]

响应参数	content-type:application/json;
	{
	"Error": "",
	"Output": "xxxxxxx"
	}
	
	Error: 成功: 空字符串 失败: 对应的错误信息
	Output: 对应的输出信息

13. ReparentTablet

Method	POST
URL	/api/vtctl/
请求参数	content-type:application/json; ["ReparentTablet", "test-1301"]

响应参数	content-type:application/json;
	{
	"Error": "",
	"Output": "xxxxxxx"
	}
	
	Error: 成功: 空字符串 失败: 对应的错误信息
	Output: 对应的输出信息

二、集群接口

1. 获取 POD 列表

Method	GET
URL	/api/k8sns/{namespace}/pods

响应参数	content-type:application/json;
	{
	"Code": 0,
	"Message": "success",
	"Data": {
	"metadata": {
	"selfLink": "/api/v1/namespaces/default/pods",
	"resourceVersion": "5453094"

<table>
<tr><td></td><td>
<pre>
    },
    "items": [
      {
        "metadata": {
          "name": "centosbase",
          "namespace": "default"
        },
        "spec": {
          "volumes": [
            {
              "name": "test"
            }
          ],
          "containers": [
            {
              "name": "centosbase",
              "image": "192.168.212.19/base_worker/go-jd-centos
6.6"
            }
          ],
          "restartPolicy": "Always",
          "terminationGracePeriodSeconds": 30,
          "dnsPolicy": "ClusterFirst",
          "nodeName": "192.168.177.12",
          "securityContext": { }
        },
        "status": {
          "phase": "Running"
        }
      }
    ]
  }}
</pre>
由于接口返回参数比较多，这里只列出了大的部分，详细部分可以直接调用接口参考， 根据需要取得对应的参数

Error: 成功: 空字符串　失败: 对应的错误信息
Output: 对应的输出信息
</td></tr>
</table>

2. 创建 POD

<table>
<tr><td>Method</td><td>POST</td></tr>
<tr><td>URL</td><td>/api/v1/namespaces/{namespace}/pods</td></tr>
<tr><td>请求参数</td><td>content-type:application/json;
<pre>
{
  "Uid": "100",   //全局唯一 id,必须是数字
  "Keyspace": "test_ks",   //逻辑库名称
  "ShardLable": "xx-80",   //
</pre>
</td></tr>
</table>

	<pre>"Alias": "test-0000000100",
	"VitessImage": "192.168.212.19/vitesss/lite:0.1.7", //镜像地址
	"Port": 3333,
	"GrpcPort": 34556,
	"TabletType": "replica",
	"BackupFlags": "/export",
	"VtdatarootVolume": "empdir: {}",
	"Shard": "-80",
	"TabletSubdir": "test-0000000100"
	}</pre>

响应参数	content-type:application/json;
	<pre>{
	"Error": "",
	"Output": "xxxxxxx"
	}</pre>
	Error: 成功: 空字符串 失败: 对应的错误信息
	Output: 对应的输出信息

3. 删除 POD

Method	DELETE
URL	/api/k8sns/{namespace}/pods/{name}

响应参数	content-type:application/json;
	<pre>{
	"Code": 0,
	"Message": "success"
	}</pre>
	Error: 成功: 空字符串 失败: 对应的错误信息
	Output: 对应的输出信息

4. 获取 POD 详细信息

Method	GET
URL	/api/k8sns/{namespace}/pods/{name}

响应参数	content-type:application/json;
	<pre>{
	"Code": 0,
	"Message": "success",
	"Data": {
	"Pod": {
	"metadata": {
	"name": "vttablet-2000",
	"labels": {
	"app": "vitess",</pre>

```
            "component": "vttablet",
            "keyspace": "test_ts",
            "shard": "xx-80",
            "tablet": "test-0000002000"
          }
        },
        "spec": {
          "volumes": [
            {
              "name": "syslog",
              "hostPath": {
                "path": "/dev/log"
              }
            },
            {
              "name": "vtdataroot",
              "emptyDir": { }
            }
          ],
          "containers": [
            {
              "name": "vttablet",
              "image": "vitess/root",
              "command": [ ],
              "volumeMounts": [
                {
                  "name": "syslog",
                  "mountPath": "/dev/log"
                },
                {
                  "name": "vtdataroot",
                  "mountPath": "/vt/vtdataroot"
                }
              ],
              "terminationMessagePath": "/dev/termination-log",
              "imagePullPolicy": "Always"
            }
          ],
          "restartPolicy": "Always",
          "terminationGracePeriodSeconds": 30,
          "dnsPolicy": "ClusterFirst",
          "nodeName": "192.168.170.164",
          "securityContext": { }
        },
        "status": {
          "phase": "Pending",
          "hostIP": "192.168.170.164",
          "podIP": "192.168.81.49",
          "startTime": "2017-03-20T05:56:25Z"
        }
      }
```

	} }

5. 获取 Service 列表

Method	GET
URL	/api/k8sns/{namespace}/services

响应参数	content-type:application/json;
	{
	"Code": 0,
	"Message": "success",
	"Data": {
	"Service": {
	"metadata": {
	"name": "vtctld",
	"namespace": "default",
	"selfLink": "/api/v1/namespaces/default/services/vtctld",
	"uid": "e172d035-0d22-11e7-8d40-e83935ef076c",
	"resourceVersion": "5462197",
	"creationTimestamp": "2017-03-20T04:08:32Z",
	"labels": {
	"app": "vitess",
	"component": "vtctld"
	}
	},
	"spec": {
	"ports": [
	{
	"name": "web",
	"protocol": "TCP",
	"port": 15000,
	"targetPort": 15000
	},
	{
	"name": "grpc",
	"protocol": "TCP",
	"port": 15999,
	"targetPort": 15999
	}
	],
	"selector": {
	"app": "vitess",
	"component": "vtctld"
	},
	"clusterIP": "10.0.87.8",
	"type": "ClusterIP",
	"externalIPs": [
	"192.168.80.240"
	],
	"deprecatedPublicIPs": [

	<pre>"192.168.80.240"
	],
	"sessionAffinity": "None"
	},
	"status": {
	"loadBalancer": { }
	}
	}
	}}</pre>
	Error: 成功：空字符串　失败：对应的错误信息
	Output: 对应的输出信息

6. 创建 Service

Method	POST
URL	/api/vtctl/

请求参数	Services 分别需要为不同服务提供，以下分别说明不同的应用程序服务创建对应的参数：
	
	content-type:application/json;
	
	1. Etcd
	<pre>{
	"Type": "etcdlb/etcdcb",　　//创建的 Service 类型
	//etcdlb:load balancing
	//etcdcb:etcd cluster bootstrap.
	"Cell": "test"　　//数据中心名称
	}</pre>
	2. Vtctld
	<pre>{
	"Type": "vtctld",　　//创建的 Service 类型
	"ServiceType": "ClusterIP"
	}</pre>
	3. Vtgate
	<pre>{
	"Type": "Vtgate",
	"Cell": "gw-test",
	"MysqlServerPort": 3306
	}</pre>

响应参数	响应参数包括对应的执行结果和创建后的 service 信息 content-type:application/json;

```
{
  "Code": 0,
  "Message": "success",
  "Data": {
    "Service": {
      "metadata": {
        "name": "etcd-gwtest-srv",
        "namespace": "default",
        "selfLink": "/api/v1/namespaces/default/services/etcd-gwtest-srv",
        "uid": "22b481a5-0d40-11e7-8d40-e83935ef076c",
        "resourceVersion": "5525297",
        "creationTimestamp": "2017-03-20T07:37:56Z",
        "labels": {
          "app": "vitess",
          "cell": "gwtest",
          "component": "etcd"
        }
      },
      "spec": {
        "ports": [
          {
            "name": "etcd-server",
            "protocol": "TCP",
            "port": 7001,
            "targetPort": 7001
          }
        ],
        "selector": {
          "app": "vitess",
          "cell": "gwtest",
          "component": "etcd"
        },
        "clusterIP": "None",
        "type": "ClusterIP",
        "sessionAffinity": "None"
      },
      "status": {
        "loadBalancer": { }
      }
    }
  }
}
```


Code：成功:0 失败：非 0

Message：对应的消息

Data:创建后 service 信息

7. 删除 Service

Method	DELETE
URL	/api/k8sns/{namespace}/services/{name}

响应参数	content-type:application/json;
	{
	"Code": 0,
	"Message": "success"
	}
	Code：成功:0 失败：非 0
	Message：对应的消息

8. 获取 Service 详细信息

Method	GET
URL	/api/k8sns/{namespace}/services/{name}

响应参数	content-type:application/json;
	{
	"Code": 0,
	"Message": "success",
	"Data": {
	"Service": {
	"metadata": {
	"name": "vtctld",
	"namespace": "default",
	"selfLink": "/api/v1/namespaces/default/services/vtctld",
	"uid": "e172d035-0d22-11e7-8d40-e83935ef076c",
	"resourceVersion": "5462197",
	"creationTimestamp": "2017-03-20T04:08:32Z",
	"labels": {
	"app": "vitess",
	"component": "vtctld"
	}
	},
	"spec": {
	"ports": [
	{
	"name": "web",
	"protocol": "TCP",
	"port": 15000,
	"targetPort": 15000
	},
	{
	"name": "grpc",
	"protocol": "TCP",
	"port": 15999,
	"targetPort": 15999
	}
	],
	"selector": {
	"app": "vitess",
	"component": "vtctld"

<table>
<tr><td></td><td>
<pre>
<code>
        },
        "clusterIP": "10.0.87.8",
        "type": "ClusterIP",
        "externalIPs": [
          "192.168.80.240"
        ],
        "deprecatedPublicIPs": [
          "192.168.80.240"
        ],
        "sessionAffinity": "None"
      },
      "status": {
        "loadBalancer": { }
      }
    }
  }
}
</code>
</pre>
</td></tr>
</table>

9. 获取 ReplicationControllerList

<table>
<tr><td>Method</td><td>GET</td></tr>
<tr><td>URL</td><td>/api/k8sns/{namespace}/rc</td></tr>
<tr><td>响应参数</td><td>content-type:application/json;
<pre>
<code>
{
  "Code": 0,
  "Message": "success",
  "Data": {
    "metadata": {
      "selfLink": "/api/v1/namespaces/default/replicationcontrollers
",
      "resourceVersion": "5531082"
    },
    "items": [
      {
        "metadata": {
          "name": "etcd-global",
          "namespace": "default"
        },
        "spec": {
          "replicas": 3,
          "template": {
            "metadata": {
              "creationTimestamp": null,
              "labels": {
                "app": "vitess",
                "cell": "global",
                "component": "etcd"
              }
            },
            "spec": {
</code>
</pre>
</td></tr>
</table>

<table>
<tr><td></td><td>

<pre>
"volumes": [
  {
    "name": "certs",
    "hostPath": {
      "path": "/etc/ssl/certs/ca-certificates.crt"
    }
  }
],
"containers": [
  {
    "name": "etcd",
    "image": "192.168.212.19/vitess/etcd:v2.0.13-lite
",
    "resources": {
      "limits": {
        "cpu": "100m",
        "memory": "128Mi"
      }
    }
  }
],
"restartPolicy": "Always",
"terminationGracePeriodSeconds": 30,
"dnsPolicy": "ClusterFirst",
"securityContext": { }
}
}
}
}
]
}}
</pre>

</td></tr>
</table>

10. 创建 ReplicationController

<table>
<tr><td>Method</td><td>POST</td></tr>
<tr><td>URL</td><td>/api/k8sns/{namespace}/rc</td></tr>
<tr><td>请求参数</td><td>content-type:application/json;
以下分表列出创建不同 rc 的参数
1. Etcd

<pre>
{
  "Type": "Etcd",
  "Cell": "gwtest",
  "Replicas": 2
}
</pre>

2. Vtctld

<pre>
{
</pre>

</td></tr>
</table>

	<pre>"Type": "Vtctld",
	"Cell": "gwtest",
	"VitessImage": "192.168.212.19/vitesss/lite57:0.1.6",
	"Replicas": 1,
	"BackupFlags": "-backup_storage_implementation file",
	"TestFlags": "-enable_queries"
	}</pre>
	3. Vtgate
	<pre>{
	"Type": "vtgate",
	"Cell": "gwtest",
	"VitessImage": "192.168.212.19/vitesss/lite57:0.1.6",
	"Replicas": 2,
	"MysqlServerPort": 3306
	}</pre>

响应参数	content-type:application/json;
	<pre>{
	"Code": 0,
	"Message": "success",
	"Data": {
	"ReplicationController": { }
	}
	}</pre>

11. 删除 ReplicationController

Method	DELETE
URL	/api/k8sns/{namespace}/rc/{name}

响应参数	content-type:application/json;
	<pre>{
	"Code": 0,
	"Message": "success"
	}</pre>

12. 获取 ReplicationController 详细信息

Method	GET
URL	/api/k8sns/{namespace}/rc/{name}

响应参数	content-type:application/json;
	<pre>{
	"Code": 0,
	"Message": "success",
	"Data": {</pre>

	"ReplicationController": { } } }

13. 获取 Namespace 列表

GET /api/v1/namespaces

Method	GET
URL	/api/k8sns

响应参数	content-type:application/json;
	{
	"Code": 0,
	"Message": "success",
	"Data": {
	"metadata": {
	"selfLink": "/api/v1/namespaces",
	"resourceVersion": "5543148"
	},
	"items": [
	{
	"metadata": {
	"name": "asdsadasd",
	"selfLink": "/api/v1/namespacesasdsadasd",
	"uid": "2cae5dc4-0895-11e7-90e1-e83935ef076c",
	"resourceVersion": "3862545",
	"creationTimestamp": "2017-03-14T09:04:05Z"
	},
	"spec": {
	"finalizers": [
	"kubernetes"
	]
	},
	"status": {
	"phase": "Active"
	}
	}
	]
	}
	}

14. 创建 Namespace

Method	POST
URL	/api/k8sns

请求参数	content-type:application/json;
	{
	"Name": "gwtest",
	"Labels": {
	"Key1": "Value",

	<pre>"Key2": "Value2",
	"Key3": "Value3",
	"Key4": "Value4",
	"Key5": "Value5"
	}
	}</pre>

响应参数	content-type:application/json;
	<pre>{
	"Code": 0,
	"Message": "success",
	"Data": {
	"Namespace": {}
	}
	}</pre>

15. 删除 Namespace

Method	DELETE
URL	/api/k8sns/{name}

响应参数	content-type:application/json;
	<pre>{
	"Code": 0,
	"Message": "success"
	}</pre>

16. 获取 Namespace 详细信息

Method	GET
URL	/api/k8sns/{name}

响应参数	content-type:application/json;
	<pre>{
	"Code": 0,
	"Message": "success",
	"Data": {
	"Namespace": {... }
	}
	}</pre>