```
[root@NTNX-244131ad-A ~]# ovs-vsctl --help
ovs-vsctl: ovs-vswitchd management utility
usage: ovs-vsctl [OPTIONS] COMMAND [ARG...]

Open vSwitch commands:
  init                        initialize database, if not yet initialized
  show                        print overview of database contents
  emer-reset                  reset configuration to clean state

Bridge commands:
  add-br BRIDGE               create a new bridge named BRIDGE
  add-br BRIDGE PARENT VLAN   create new fake BRIDGE in PARENT on VLAN
  del-br BRIDGE               delete BRIDGE and all of its ports
  list-br                     print the names of all the bridges
  br-exists BRIDGE            exit 2 if BRIDGE does not exist
  br-to-vlan BRIDGE           print the VLAN which BRIDGE is on
  br-to-parent BRIDGE         print the parent of BRIDGE
  br-set-external-id BRIDGE KEY VALUE  set KEY on BRIDGE to VALUE
  br-set-external-id BRIDGE KEY  unset KEY on BRIDGE
  br-get-external-id BRIDGE KEY  print value of KEY on BRIDGE
  br-get-external-id BRIDGE  list key-value pairs on BRIDGE

Port commands (a bond is considered to be a single port):
  list-ports BRIDGE           print the names of all the ports on BRIDGE
  add-port BRIDGE PORT        add network device PORT to BRIDGE
  add-bond BRIDGE PORT IFACE...  add bonded port PORT in BRIDGE from IFACES
  del-port [BRIDGE] PORT      delete PORT (which may be bonded) from BRIDGE
  port-to-br PORT             print name of bridge that contains PORT

Interface commands (a bond consists of multiple interfaces):
  list-ifaces BRIDGE          print the names of all interfaces on BRIDGE
  iface-to-br IFACE           print name of bridge that contains IFACE

Controller commands:
  get-controller BRIDGE      print the controllers for BRIDGE
  del-controller BRIDGE      delete the controllers for BRIDGE
  [--inactivity-probe=MSECS]
  set-controller BRIDGE TARGET...  set the controllers for BRIDGE
  get-fail-mode BRIDGE       print the fail-mode for BRIDGE
  del-fail-mode BRIDGE       delete the fail-mode for BRIDGE
  set-fail-mode BRIDGE MODE  set the fail-mode for BRIDGE to MODE

Manager commands:
  get-manager                print the managers
  del-manager                delete the managers
  [--inactivity-probe=MSECS]
  set-manager TARGET...      set the list of managers to TARGET...

SSL commands:
  get-ssl                     print the SSL configuration
  del-ssl                     delete the SSL configuration
```

```
  set-ssl PRIV-KEY CERT CA-CERT  set the SSL configuration

Auto Attach commands:
  add-aa-mapping BRIDGE I-SID VLAN   add Auto Attach mapping to BRIDGE
  del-aa-mapping BRIDGE I-SID VLAN   delete Auto Attach mapping VLAN from BRIDGE
  get-aa-mapping BRIDGE              get Auto Attach mappings from BRIDGE

Switch commands:
  emer-reset                  reset switch to known good state

Database commands:
  list TBL [REC]              list RECord (or all records) in TBL
  find TBL CONDITION...       list records satisfying CONDITION in TBL
  get TBL REC COL[:KEY]       print values of COLumns in RECord in TBL
  set TBL REC COL[:KEY]=VALUE set COLumn values in RECord in TBL
  add TBL REC COL [KEY=]VALUE add (KEY=)VALUE to COLumn in RECord in TBL
  remove TBL REC COL [KEY=]VALUE  remove (KEY=)VALUE from COLumn
  clear TBL REC COL           clear values from COLumn in RECord in TBL
  create TBL COL[:KEY]=VALUE  create and initialize new record
  destroy TBL REC             delete RECord from TBL
  wait-until TBL REC [COL[:KEY]=VALUE]  wait until condition is true
Potentially unsafe database commands require --force option.
Database commands may reference a row in each table in the following ways:
  AutoAttach:
    by UUID
    via "auto_attach" of Bridge with matching "name"
  Bridge:
    by UUID
    by "name"
  CT_Timeout_Policy:
    by UUID
  CT_Zone:
    by UUID
  Controller:
    by UUID
    via "controller" of Bridge with matching "name"
  Datapath:
    by UUID
  Flow_Sample_Collector_Set:
    by UUID
    by "id"
  Flow_Table:
    by UUID
    by "name"
  IPFIX:
    by UUID
    via "ipfix" of Bridge with matching "name"
  Interface:
    by UUID
    by "name"
  Manager:
```

```
    by UUID
    by "target"
  Mirror:
    by UUID
    by "name"
  NetFlow:
    by UUID
    via "netflow" of Bridge with matching "name"
  Open_vSwitch:
    by UUID
    as "."
  Port:
    by UUID
    by "name"
  QoS:
    by UUID
    via "qos" of Port with matching "name"
  Queue:
    by UUID
  SSL:
    by UUID
    as "."
  sFlow:
    by UUID
    via "sflow" of Bridge with matching "name"

Options:
  --db=DATABASE               connect to DATABASE
                              (default: unix:/var/run/openvswitch/db.sock)
  --no-wait                   do not wait for ovs-vswitchd to reconfigure
  --retry                     keep trying to connect to server forever
  -t, --timeout=SECS          wait at most SECS seconds for ovs-vswitchd
  --dry-run                   do not commit changes to database
  --oneline                   print exactly one line of output per command

Output formatting options:
  -f, --format=FORMAT         set output formatting to FORMAT
                              ("table", "html", "csv", or "json")
  -d, --data=FORMAT           set table cell output formatting to
                              FORMAT ("string", "bare", or "json")
  --no-headings               omit table heading row
  --pretty                    pretty-print JSON in output
  --bare                      equivalent to "--format=list --data=bare --no-headings"

Logging options:
  -vSPEC, --verbose=SPEC   set logging levels
  -v, --verbose            set maximum verbosity level
  --log-file[=FILE]        enable logging to specified FILE
                           (default: /var/log/openvswitch/ovs-vsctl.log)
  --syslog-method=(libc|unix:file|udp:ip:port)
                           specify how to send messages to syslog daemon
```

```
  --syslog-target=HOST:PORT  also send syslog msgs to HOST:PORT via UDP
  --no-syslog             equivalent to --verbose=vsctl:syslog:warn

Active database connection methods:
  tcp:HOST:PORT           PORT at remote HOST
  ssl:HOST:PORT           SSL PORT at remote HOST
  unix:FILE               Unix domain socket named FILE
Passive database connection methods:
  ptcp:PORT[:IP]          listen to TCP PORT on IP
  pssl:PORT[:IP]          listen for SSL on PORT on IP
  punix:FILE              listen on Unix domain socket FILE
PKI configuration (required to use SSL):
  -p, --private-key=FILE  file with private key
  -c, --certificate=FILE  file with certificate for private key
  -C, --ca-cert=FILE      file with peer CA certificate
  --bootstrap-ca-cert=FILE  file with peer CA certificate to read or create
SSL options:
  --ssl-protocols=PROTOS  list of SSL protocols to enable
  --ssl-ciphers=CIPHERS   list of SSL ciphers to enable

Other options:
  -h, --help                  display this help message
  -V, --version               display version information
[root@NTNX-244131ad-A ~]#
```