mysql-entrypoint.sh

#### entrypoint.sh-内容

|  |
| --- |
| #!/bin/bash  set -eo pipefail  shopt -s nullglob  # if command starts with an option, prepend mysqld  if [ "${1:0:1}" = '-' ]; then  set -- mysqld "$@"  fi  # skip setup if they want an option that stops mysqld  wantHelp=  for arg; do  case "$arg" in  -'?'|--help|--print-defaults|-V|--version)  wantHelp=1  break  ;;  esac  done  # usage: file\_env VAR [DEFAULT]  # ie: file\_env 'XYZ\_DB\_PASSWORD' 'example'  # (will allow for "$XYZ\_DB\_PASSWORD\_FILE" to fill in the value of  # "$XYZ\_DB\_PASSWORD" from a file, especially for Docker's secrets feature)  file\_env() {  local var="$1"  local fileVar="${var}\_FILE"  local def="${2:-}"  if [ "${!var:-}" ] && [ "${!fileVar:-}" ]; then  echo >&2 "error: both $var and $fileVar are set (but are exclusive)"  exit 1  fi  local val="$def"  if [ "${!var:-}" ]; then  val="${!var}"  elif [ "${!fileVar:-}" ]; then  val="$(< "${!fileVar}")"  fi  export "$var"="$val"  unset "$fileVar"  }  \_check\_config() {  toRun=( "$@" --verbose --help --log-bin-index="$(mktemp -u)" )  if ! errors="$("${toRun[@]}" 2>&1 >/dev/null)"; then  cat >&2 <<-EOM  ERROR: mysqld failed while attempting to check config  command was: "${toRun[\*]}"  $errors  EOM  exit 1  fi  }  # Fetch value from server config  # We use mysqld --verbose --help instead of my\_print\_defaults because the  # latter only show values present in config files, and not server defaults  \_get\_config() {  local conf="$1"; shift  "$@" --verbose --help --log-bin-index="$(mktemp -u)" 2>/dev/null | awk '$1 == "'"$conf"'" { print $2; exit }'  }  # allow the container to be started with `--user`  if [ "$1" = 'mysqld' -a -z "$wantHelp" -a "$(id -u)" = '0' ]; then  \_check\_config "$@"  DATADIR="$(\_get\_config 'datadir' "$@")"  mkdir -p "$DATADIR"  chown -R mysql:mysql "$DATADIR"  exec gosu mysql "$BASH\_SOURCE" "$@"  fi  if [ "$1" = 'mysqld' -a -z "$wantHelp" ]; then  # still need to check config, container may have started with --user  \_check\_config "$@"  # Get config  DATADIR="$(\_get\_config 'datadir' "$@")"  if [ ! -d "$DATADIR/mysql" ]; then  file\_env 'MYSQL\_ROOT\_PASSWORD'  if [ -z "$MYSQL\_ROOT\_PASSWORD" -a -z "$MYSQL\_ALLOW\_EMPTY\_PASSWORD" -a -z "$MYSQL\_RANDOM\_ROOT\_PASSWORD" ]; then  echo >&2 'error: database is uninitialized and password option is not specified '  echo >&2 ' You need to specify one of MYSQL\_ROOT\_PASSWORD, MYSQL\_ALLOW\_EMPTY\_PASSWORD and MYSQL\_RANDOM\_ROOT\_PASSWORD'  exit 1  fi  mkdir -p "$DATADIR"  echo 'Initializing database'  mysql\_install\_db --datadir="$DATADIR" --rpm --keep-my-cnf  echo 'Database initialized'  SOCKET="$(\_get\_config 'socket' "$@")"  "$@" --skip-networking --socket="${SOCKET}" &  pid="$!"  mysql=( mysql --protocol=socket -uroot -hlocalhost --socket="${SOCKET}" )  for i in {30..0}; do  if echo 'SELECT 1' | "${mysql[@]}" &> /dev/null; then  break  fi  echo 'MySQL init process in progress...'  sleep 1  done  if [ "$i" = 0 ]; then  echo >&2 'MySQL init process failed.'  exit 1  fi  if [ -z "$MYSQL\_INITDB\_SKIP\_TZINFO" ]; then  # sed is for https://bugs.mysql.com/bug.php?id=20545  mysql\_tzinfo\_to\_sql /usr/share/zoneinfo | sed 's/Local time zone must be set--see zic manual page/FCTY/' | "${mysql[@]}" mysql  fi  if [ ! -z "$MYSQL\_RANDOM\_ROOT\_PASSWORD" ]; then  export MYSQL\_ROOT\_PASSWORD="$(pwgen -1 32)"  echo "GENERATED ROOT PASSWORD: $MYSQL\_ROOT\_PASSWORD"  fi  rootCreate=  # default root to listen for connections from anywhere  file\_env 'MYSQL\_ROOT\_HOST' '%'  if [ ! -z "$MYSQL\_ROOT\_HOST" -a "$MYSQL\_ROOT\_HOST" != 'localhost' ]; then  # no, we don't care if read finds a terminating character in this heredoc  # https://unix.stackexchange.com/questions/265149/why-is-set-o-errexit-breaking-this-read-heredoc-expression/265151#265151  read -r -d '' rootCreate <<-EOSQL || true  CREATE USER 'root'@'${MYSQL\_ROOT\_HOST}' IDENTIFIED BY '${MYSQL\_ROOT\_PASSWORD}' ;  GRANT ALL ON \*.\* TO 'root'@'${MYSQL\_ROOT\_HOST}' WITH GRANT OPTION ;  EOSQL  fi  "${mysql[@]}" <<-EOSQL  -- What's done in this file shouldn't be replicated  -- or products like mysql-fabric won't work  SET @@SESSION.SQL\_LOG\_BIN=0;  DELETE FROM mysql.user WHERE user NOT IN ('mysql.sys', 'mysqlxsys', 'root') OR host NOT IN ('localhost') ;  SET PASSWORD FOR 'root'@'localhost'=PASSWORD('${MYSQL\_ROOT\_PASSWORD}') ;  GRANT ALL ON \*.\* TO 'root'@'localhost' WITH GRANT OPTION ;  ${rootCreate}  DROP DATABASE IF EXISTS test ;  FLUSH PRIVILEGES ;  EOSQL  if [ ! -z "$MYSQL\_ROOT\_PASSWORD" ]; then  mysql+=( -p"${MYSQL\_ROOT\_PASSWORD}" )  fi  file\_env 'MYSQL\_DATABASE'  if [ "$MYSQL\_DATABASE" ]; then  echo "CREATE DATABASE IF NOT EXISTS \`$MYSQL\_DATABASE\` ;" | "${mysql[@]}"  mysql+=( "$MYSQL\_DATABASE" )  fi  file\_env 'MYSQL\_USER'  file\_env 'MYSQL\_PASSWORD'  if [ "$MYSQL\_USER" -a "$MYSQL\_PASSWORD" ]; then  echo "CREATE USER '$MYSQL\_USER'@'%' IDENTIFIED BY '$MYSQL\_PASSWORD' ;" | "${mysql[@]}"  if [ "$MYSQL\_DATABASE" ]; then  echo "GRANT ALL ON \`$MYSQL\_DATABASE\`.\* TO '$MYSQL\_USER'@'%' ;" | "${mysql[@]}"  fi  echo 'FLUSH PRIVILEGES ;' | "${mysql[@]}"  fi  echo  for f in /docker-entrypoint-initdb.d/\*; do  case "$f" in  \*.sh) echo "$0: running $f"; . "$f" ;;  \*.sql) echo "$0: running $f"; "${mysql[@]}" < "$f"; echo ;;  \*.sql.gz) echo "$0: running $f"; gunzip -c "$f" | "${mysql[@]}"; echo ;;  \*) echo "$0: ignoring $f" ;;  esac  echo  done  if [ ! -z "$MYSQL\_ONETIME\_PASSWORD" ]; then  "${mysql[@]}" <<-EOSQL  ALTER USER 'root'@'%' PASSWORD EXPIRE;  EOSQL  fi  if ! kill -s TERM "$pid" || ! wait "$pid"; then  echo >&2 'MySQL init process failed.'  exit 1  fi  echo  echo 'MySQL init process done. Ready for start up.'  echo  fi  fi  exec "$@" |