# 6/20/16

**See how and how the database was created (user)**

Show create table users;

**Show the databases, use a database and show the tables contained in the database.**

Show databases;

Use PS3Robot;

Show tables;

**Show the contents of the table called command.**

select \* from command;

**Install / create a database/user**

<http://www.raspberry-projects.com/pi/software_utilities/web-servers/mysql>

-To have the database on a RAM disk do this:

1. Allocate ram space.

<https://www.domoticz.com/wiki/Setting_up_a_RAM_drive_on_Raspberry_Pi>

2.Move mysql directory to the new space.

[http://tecadmin.net/change-default-mysql-data-directory-in-linux/#](http://tecadmin.net/change-default-mysql-data-directory-in-linux/)

**Re-install mysql**

sudo apt-get --purge remove mysql-server -y && sudo apt-get update -y && sudo apt-get install mysql-server –y

# 6/21/16

Two tables: one for robot status and one for control (command requests)

**status**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| IP | Name | cdate | ctime | Enable/Disable  Motors (emotors) | PWM  Left | PWM  Right | PWM  Aux#1 | PWM  Aux#2 | Report  Status |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |

**Control**

**Should have:**

-A motor speed multiplier (whole number or fractional) (double)

-Each command to have an enable/disable field to check game settings.

-

Something like that ?

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| IP | Name | cdate | ctime | Speed Multiplier | Speed Multiplier Enable | Speed Multiplier Left | Speed Multiplier Right | One sided speed enable | Aux multiplier | Aux multiplier enable | Aux enable |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |

-So as far as control goes, there are multiple ways to command the robots. Do we want to have specific control directly from the server. ie: server can set all fields of the robot status to whatever it wants.

-We can also have a simple code. ie: 1 = can drive, 2 = speed boost for 20 seconds, 3 = stop robot.

- I think we could use the report status field with numerical codes, this way each number that is not ‘-1’ would correspond to a request from the robot. ie: rfid tag detected. A process can monitor the report status field and send data (from the control database) for which command and data needs to be sent to the robot.

**Using sqlite:**

Quite different, can’t get a table going yet..

# 6/22/16

**Optimizing sqlite on raspi**

<https://spin.atomicobject.com/2013/11/14/sqlite-raspberry-pi/>

**Fetch data from mysql database**

<https://scriptingmysql.wordpress.com/2011/09/09/retrieving-data-from-mysql-via-python/>

<https://kushaldas.in/posts/fetching-row-by-row-from-mysql-in-python.html>

**Delete data from a table in MySQL**

<http://www.mysqltutorial.org/mysql-delete-statement.aspx>

<http://dev.mysql.com/doc/refman/5.7/en/delete.html>

# 6/30/16

**On server side, grant access to controller database to robot1 on any IP robot 1 may use.**

mysql>grant all privileges on Controller.\* to "robot1"@"%" identified by "therobot1";

**grant privileges on robot on robot**

mysql> grant all privileges on \* to "robot1"@"131.202.14.160";

note: make sure to do “mysql>use ‘databasename’; before this line

In short :

1. Create a user on the robot (robot1, robot2, etc)
2. Grand privileges locally on robot
3. Grant privileges on server for all ip(for easier setup during competition)
4. Use script to modify database.

**Trial run when inserting to remote database directly from PS3\_client main process.**

Regardless of update frequency (counter of 50 or 10 loops), the robot stays stuck on the current driving mode for 2-3 seconds for every 10-15 seconds. Not ideal.

I think the best would be to send the data using Zmq to another process which does the insertion to the server. This process would be the one to listen for server inputs, to request the data from the main process and then to send it to server once main process has delivered data.

**Show all users**

SELECT User FROM mysql.user;

**Trial run while using zmq to send data to a local process where the process inserts the data into a remote database.**

I am excited that this method works in a non-blocking fashion. The PS3 controls are neither delayed nor unresponsive at any time.

**MySQL Query format(this helped)**

<http://stackoverflow.com/questions/775296/python-mysql-parameterized-queries>

**other**

<http://zguide.zeromq.org/page:all>

# 7/4/16

**Non-blocking subscriber recv()**

<http://grokbase.com/t/zeromq/zeromq-dev/124368ntw8/how-to-make-recv-non-blocking>

<http://stackoverflow.com/questions/26012132/zero-mq-socket-recv-call-is-blocking>

Note: subscribers need to be active when the publisher starts running. Otherwise there will be no messages sent.

**Using POLLING**

<http://bytersplace.blogspot.ca/2013/12/zmq-and-non-blocking-socket.html>

**Redirecting STDOUT to /dev/null:**

sudo python PS3\_client6.py > /dev/null 2>&1

**Kill a process by port number it is using**

Anyway, here's how to kill a process that owns a particular port:

sudo netstat -ap | grep :<port\_number>

That will output the line corresponding to the process holding port . Then, look in the last column, you'll see /. Then execute this:

kill <pid>

If that doesn't work (you can check by re-running the netstat command). Do this:

kill -9 <pid>

In general, it's better to avoid sending SIGKILL if you can. This is why I tell you to try kill before kill -9. Just using kill sends the gentler SIGTERM.

Like I said, it will still take a few minutes for the port to re-open if you do this. I don't know a way to speed this up. If someone else does, I'd love to hear it.

# 7/26/16

Architecture 2 does not seem to require a database controller. The process commits properly when using the non-blocking zmq receive function (to receive from the robot). From documentation, the non-block initializes a buffer as well which seems to make sure we are not missing any messages.

**Change column variable type**

To change column a from [INTEGER](http://dev.mysql.com/doc/refman/5.7/en/integer-types.html) to TINYINT NOT NULL (leaving the name the same), and to change column b from CHAR(10) toCHAR(20) as well as renaming it from b to c:

ALTER TABLE t2 MODIFY a TINYINT NOT NULL, CHANGE b c CHAR(20);

\*Finally got the DB insert figured out. There was a simple syntax error.

**RECV() buffer**

From documentation online (a few different sources), the recv() function of zmq creates a buffer by default. It is also possible to reference a buffer -> recv(\*buf) where you have to initialize your own string array as a buffer.

# 7/28/16

Controller to robot data encapsulation:

Request:

1 = reset command values(ID,drive,aux, etc) to none.

2 = report your status to the server

3 = robot must comply with “drive” variable only (on/off)

4 = robot must comply with drive,aux,special1 and special2.

5 = for expanded functionality

The robot will remain in the last received command state until another command is received.

Implement a key press sequence to override this in case the server goes down. (so we don’t have to reboot robots)

GOT START/STOP TO WORK !!!

# 7/29/16