Cache Monitoring System

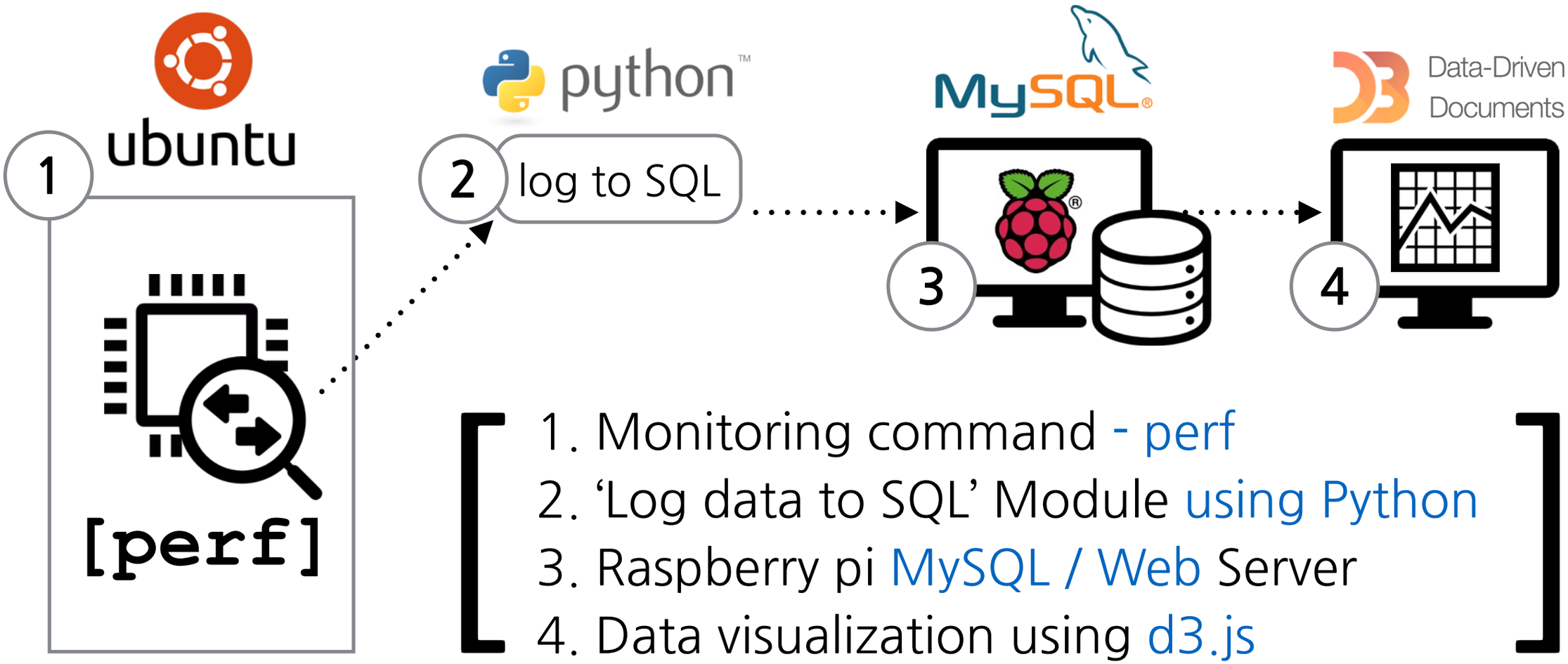
with Raspberry Pi & Data visualization

K M U C S
시스템최신기술
라즈베리사조

박성우 최승혁
이두나 허성실
최윤승

목 차

1. Cache monitoring System 1
- perf to d3.js



[
1. Monitoring command - perf
2. ‘Log data to SQL’ Module using Python
3. Raspberry pi MySQL / Web Server
4. Data visualization using d3.js
]

1.1. Monitoring Command
- perf

- **issue :** Raspbian OS가 설치된 Raspberry Pi에선 cache 모니터링을 지원하지 않는다.

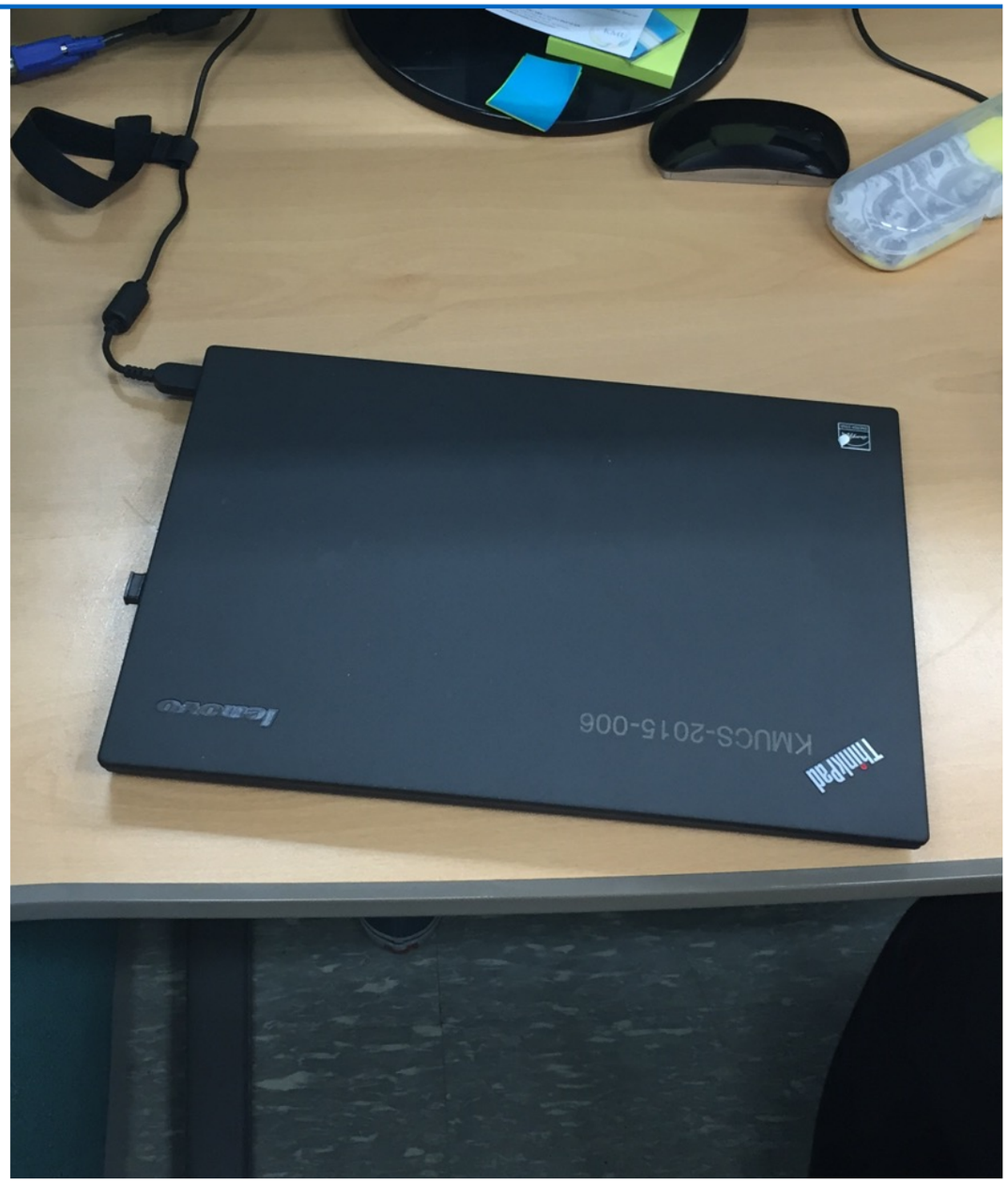
```
pi@raspberrypi ~ $ perf stat -B -e cache-references,cache-misses,cycles,instruct
ions,branches,faults,migrations sleep 5

 Performance counter stats for 'sleep 5':

   <not supported>      cache-references
   <not supported>      cache-misses
   <not supported>      cycles
   <not supported>      instructions
   <not supported>      branches
                48      faults
                 0

       5.007378743 se
```

```
pi@raspberrypi ~ $ perf stat -B -e L1-dcache-loads,L1-dcache-load-misses sleep 2

 Performance counter stats for 'sleep 2':

   <not supported>      L1-dcache-loads
   <not supported>      L1-dcache-load-misses

       2.007364033 seconds time elapsed
```

1.1. Monitoring Command
- perf

[cache miss
monitoring 환경]

- lenovo T440
- intel CORE(TM) i7
- Ubuntu 14.04

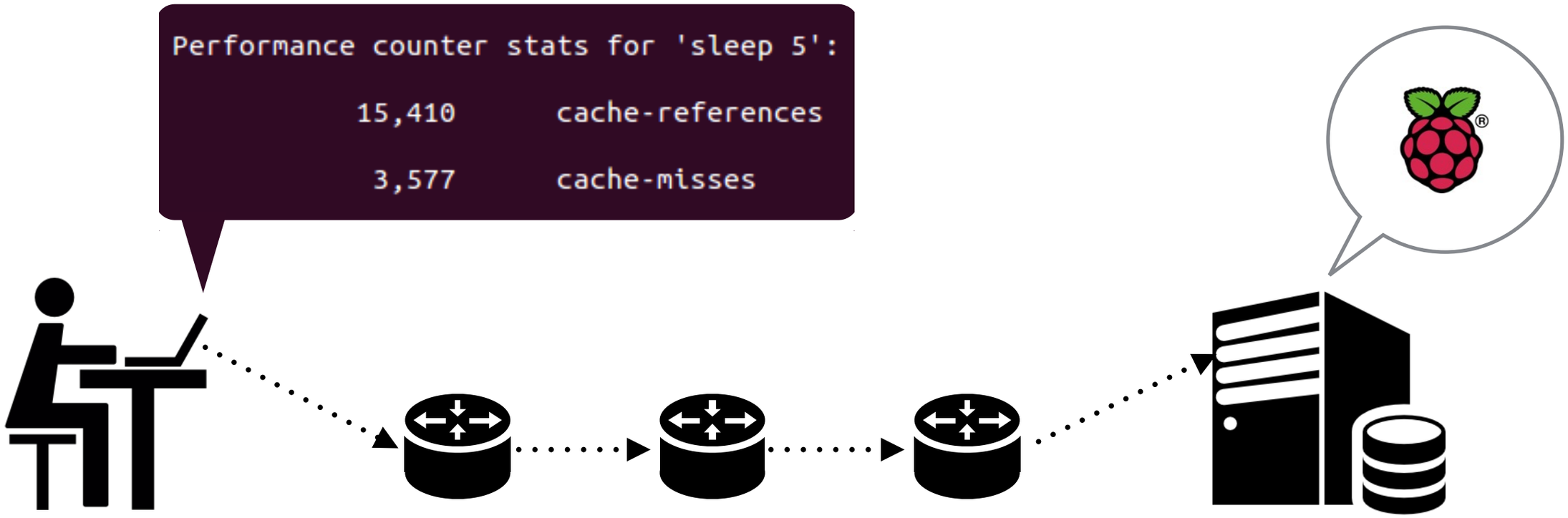


1.1. Monitoring Command - perf

```
dunam@dunam-ThinkPad-T440: ~
dunam@dunam-ThinkPad-T440:~$ perf stat -B -e cache-references,cache-misses sleep
 5

 Performance counter stats for 'sleep 5':

           15,410      cache-references
            3,577      cache-misses              #   23.212 % of all cache refs

       5.000597002 seconds time elapsed

dunam@dunam-ThinkPad-T440:~$
dunam@dunam-ThinkPad-T440:~$
```

- Cache 참조 횟수와 이에 따른 miss 발생 횟수를 알기 위해 다음 명령어를 입력한다.

```
$ perf stat -B -e cache-references, cache-misses sleep 5
```



이렇게 얻어낸 결과값을, 어떻게 서버로 보낼까?

1.2. ‘Log data to SQL’ Module using Python

< To do list >

1. 로그 데이터에 정규식을 사용,
   원하는 값을 얻어내자 !!

2. 얻어낸 값을 MySQL 서버에
   SQL 쿼리를 전송하자 !!

1.2. ‘Log data to SQL’ Module using Python

final.py UNREGISTERED

final.py

```
1  import MySQLdb as db
2  import datetime
3  import string
4  import re
5  import time
6  import subprocess
7
8  while 1:
9      test = subprocess.Popen(["perf","stat","-B","-o", "data.txt", "-e", \
10         "cache-references,cache-misses", "sleep", "1"],     stdout=subprocess.PIPE)
11     test.communicate()[0]
12
13     f = open("data.txt", "r")
14     f.readline()
15     line = 1
16     i = 0
17     data = [0]*2
18
19     while line:
20
21         line = f.readline()
22         result = re.findall(r'\s([\d,]+)\s',line)
23         if result:
24         data[i] = result[0].replace(',', '')
25             data[i] = float(data[i])
26             i = i+1
27
28     tmpDB = db.connect('20                                          n')
29
30     cur = tmpDB.cursor()
31     current_time = datetime.datetime.now()
32
33     query = "INSERT INTO Cache(time, reference, miss, missrate)"\
34        "VALUES(%s,%s,%s,%s)"
35     data = [(current_time, data[0], data[1], data[1]/data[0])]
36     rslt = cur.executemany(query,data)
37     tmpDB.commit()
```

(ipaddr, usrname, passwd, schemaName)

Line 1, Column 1 Spaces: 4 Python

다음 과정을 반복

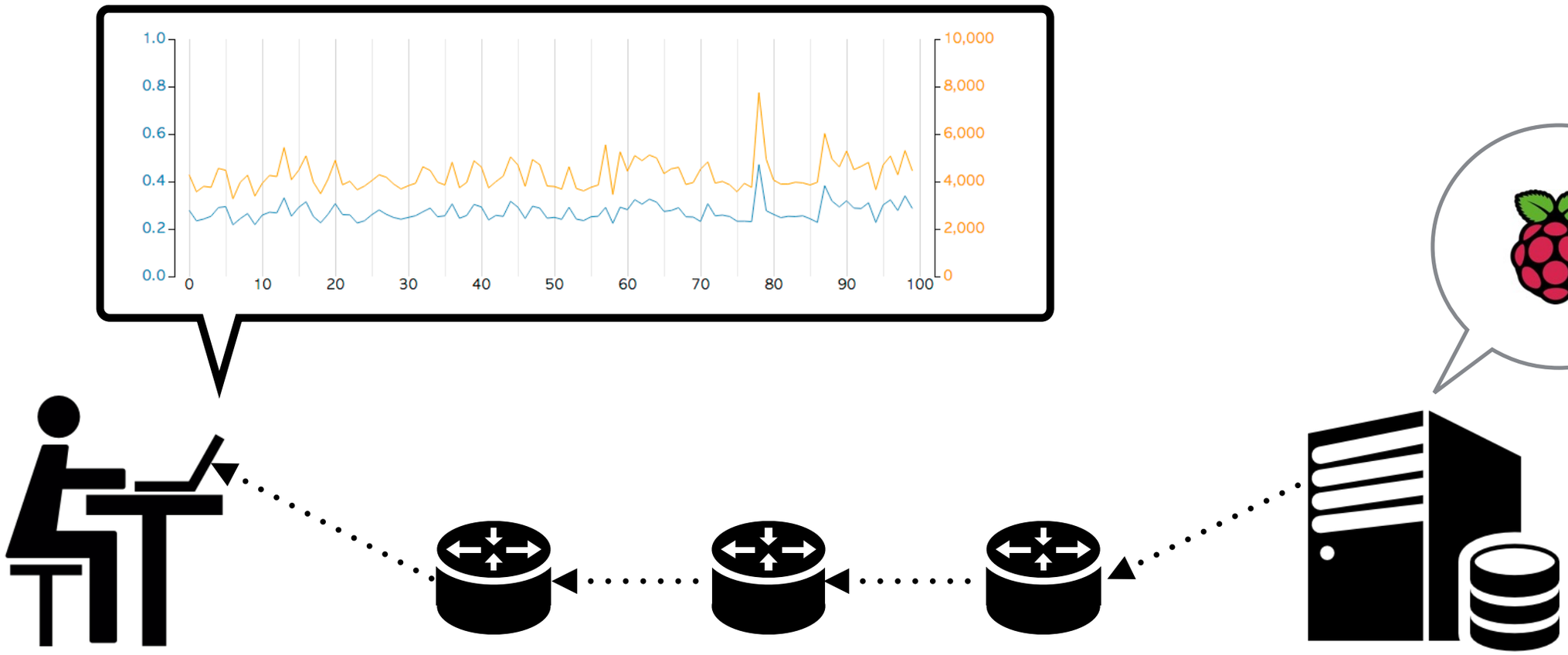
cache-references 와 cache-misses 저장

Cache monitoring 콘솔 명령 실행

정규식을 이용하여 원하는 결과값 저장

데이터베이스에 연결 요청

결과값을 DB에 저장한다.



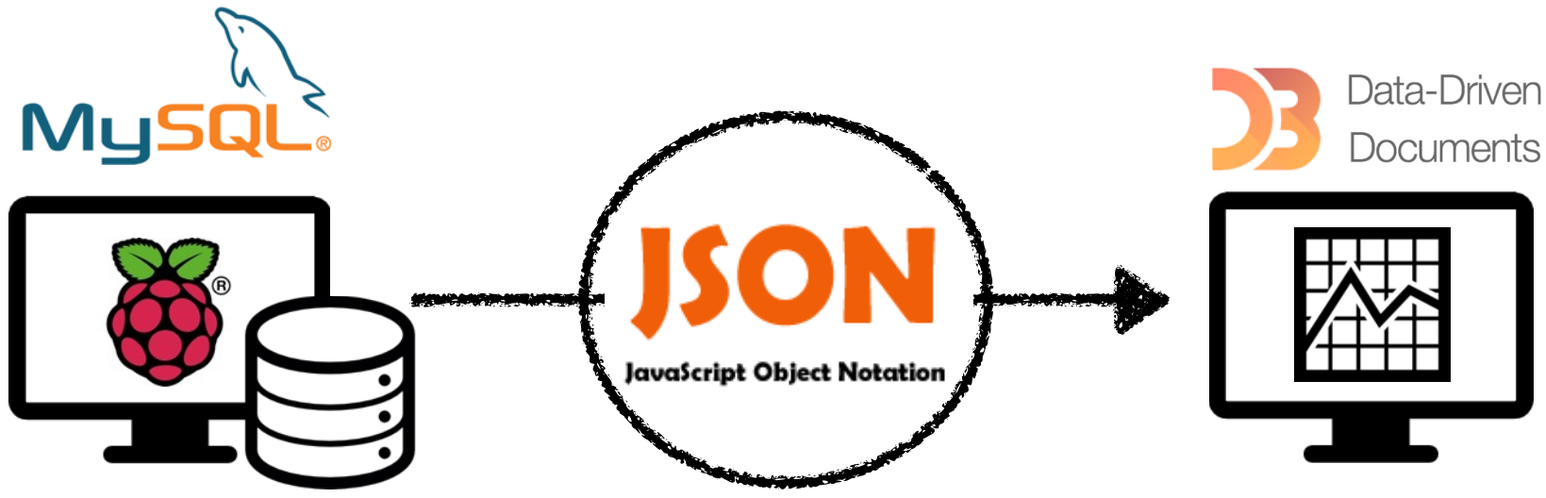
이렇게 저장된 결과값을, 어떻게 가시화 할까?

1.3. Raspberry pi
MySQL / Web Server

[MySQL Database &Web Server 환경]

- Raspberry Pi 2 Model B
- ARM Coretex-A7
- Raspbian OS
  (unofficial port of Debian Wheezy)

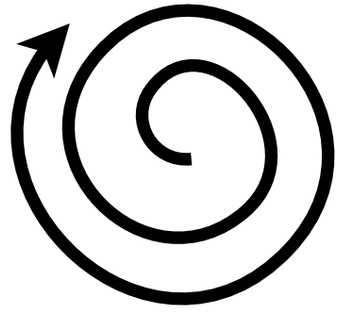
1.3. Raspberry pi
MySQL / Web Server



MySQL Server
v5.5.41

Python v3.4
simpleHTTPServer

1.3. Raspberry pi
MySQL / Web Server



Python 프로그램이
JSON을 실시간으로
최신화 한다!!

1.3. Raspberry pi
MySQL / Web Server

sql.py

File Edit Search Options Help

```
1  while 1 :
2          import MySQLdb as db
3          import time
4          tmpDB = db.connect('20                    on')
5
6          cur = tmpDB.cursor()
7
8          query = "SELECT * FROM Cache ORDER BY time DESC LIMIT 100"
9          cur.execute(query)
10
11         data=cur.fetchall()
12         #print data
13
14         f = open("d3.json", 'w')
15         f.write ("{\n\t\"record\": [\n")
16
17         for ent in data :
18                 f.write ("\t\t{\n")
19                 f.write ("\t\t\t\"time\": \""+ ent[0]+ "\",\n")
20                 f.write ("\t\t\t\"reference\": "+ str(ent[1])+ ",\n")
21                 f.write ("\t\t\t\"miss\": "+ str(ent[2])+ ",\n")
22                 f.write ("\t\t\t\"missrate\": "+ str(ent[3])+ "\n")
23                 f.write ("\t\t},\n")
24
25         f.seek(f.tell()-2)
26         f.write ("\n\t]\n}")
27         f.close()
28         time.sleep(1)
29
```

(ipaddr, usrname, passwd, schemaName)

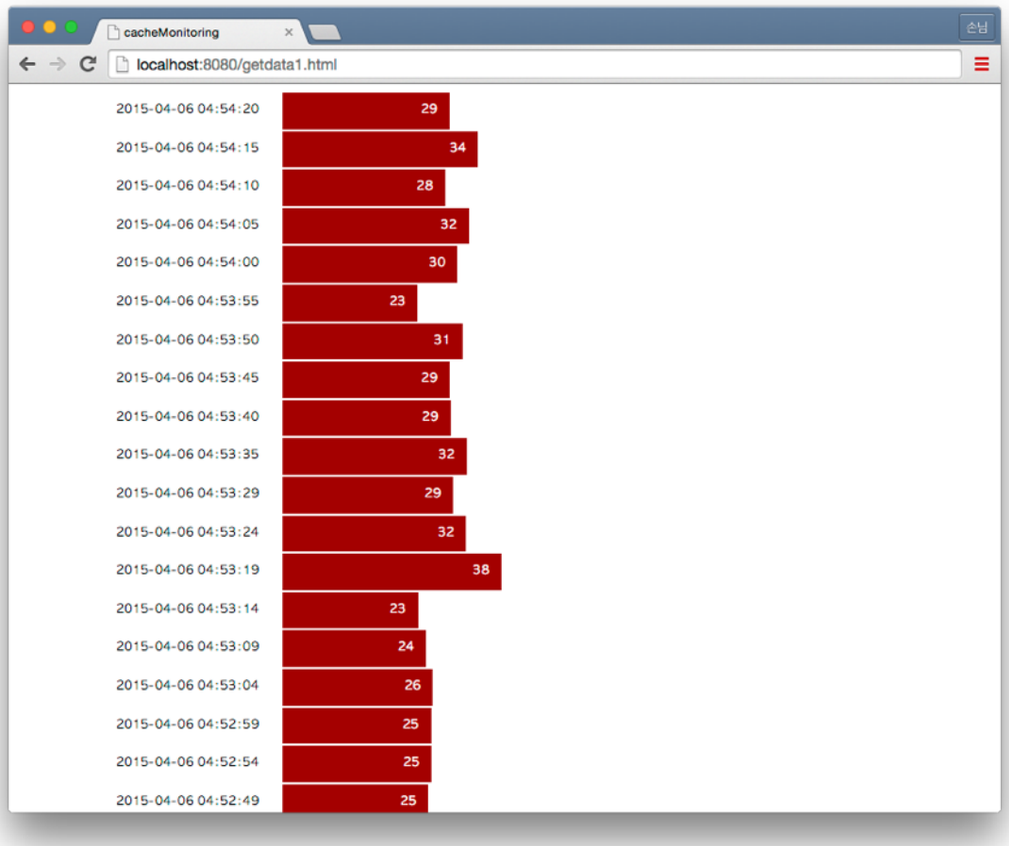
데이터베이스에 연결 요청

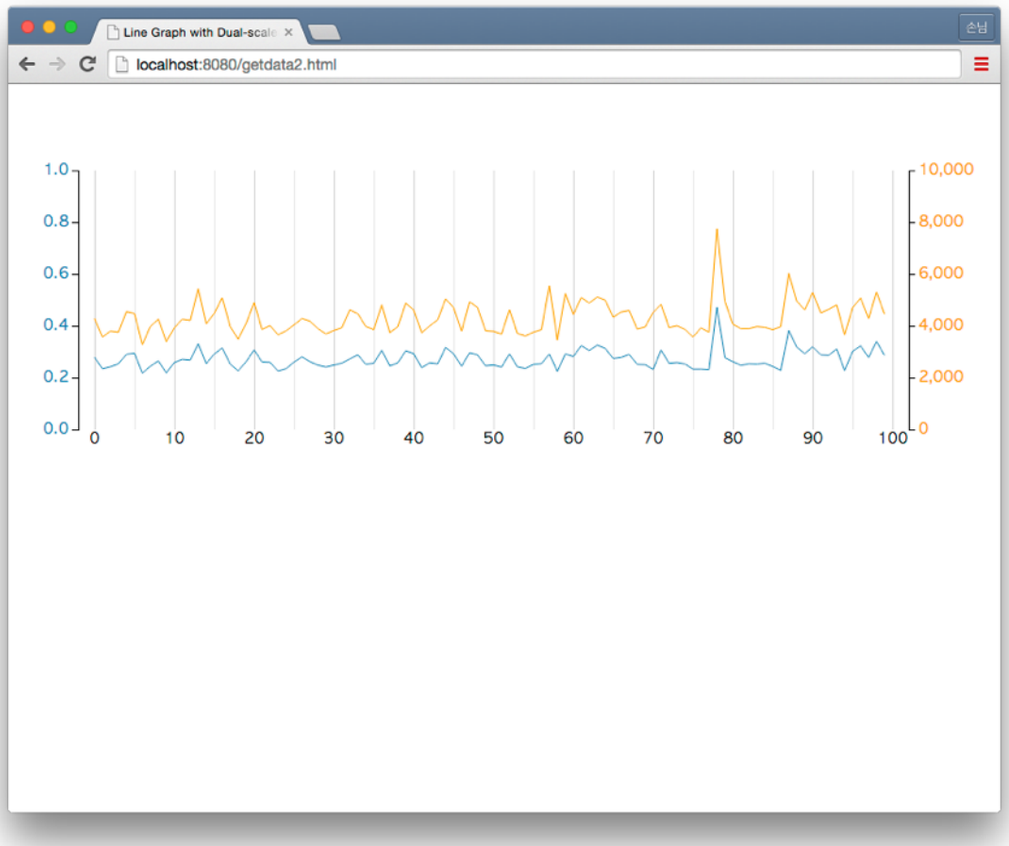
DB로 부터 최신의 cache-references와 cache-misses를 100개 가져온다.

가져온 결과를 JSON 파일로 저장

모니터링 콘솔 명령을 ‘sleep 1’로 하였기에 1초간 휴식해도 된다

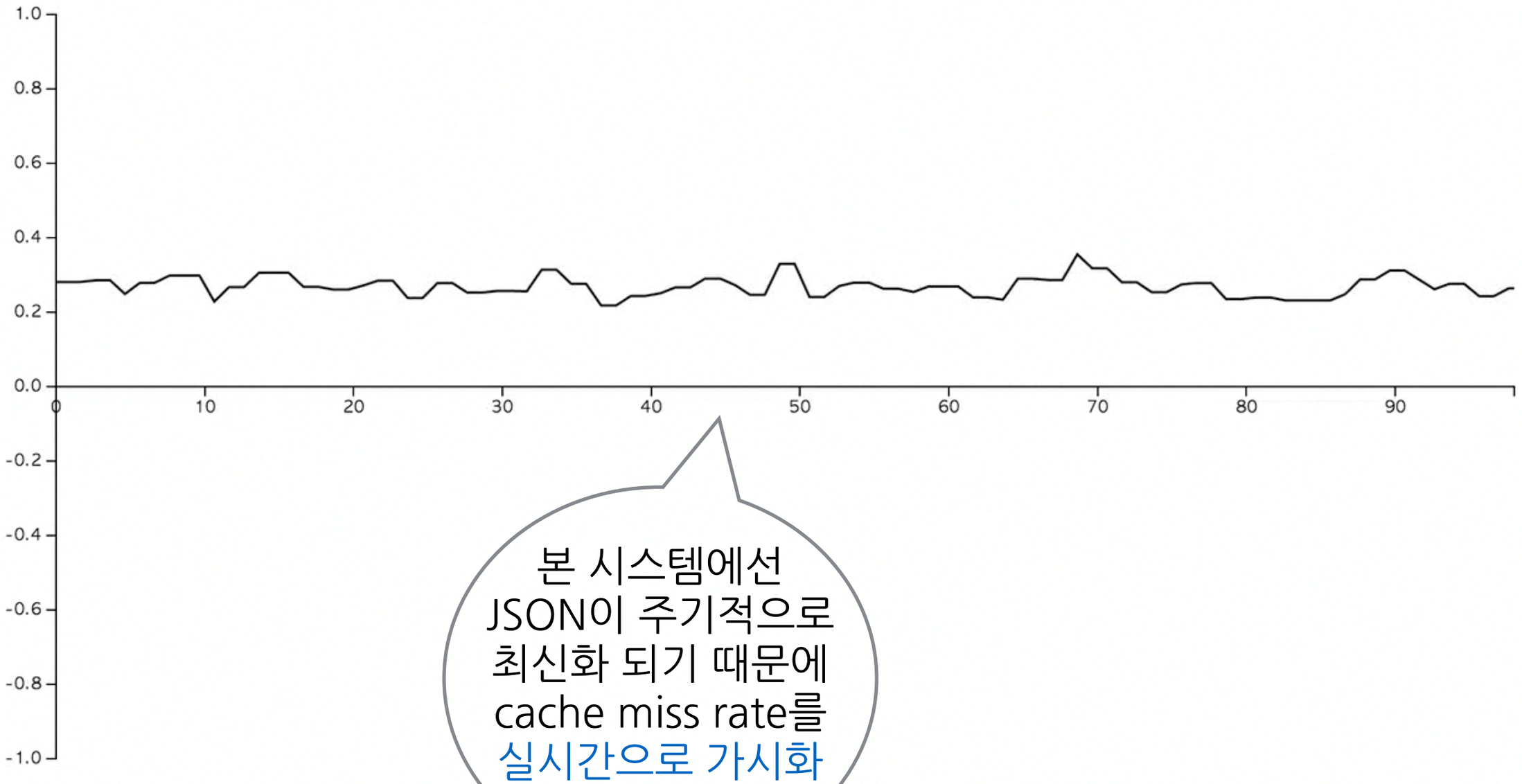
1.4. Data visualization using d3.js





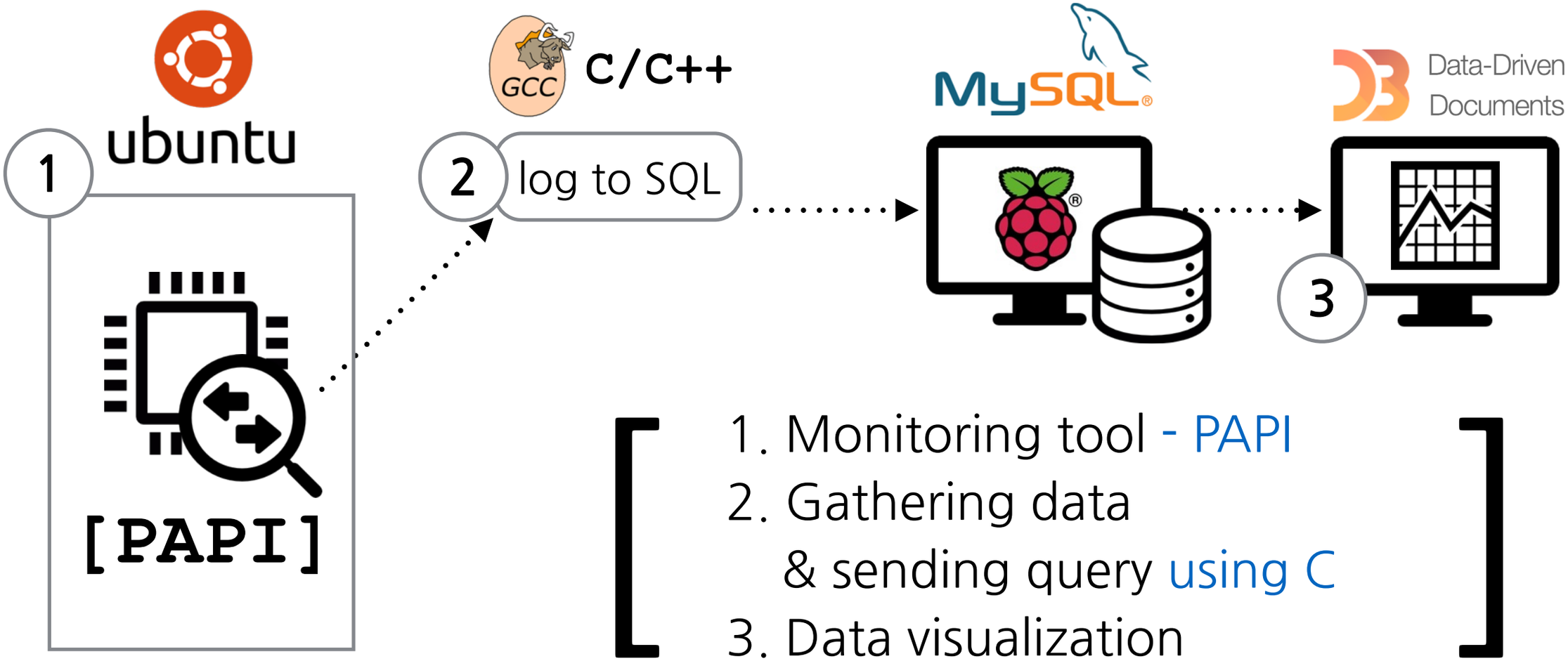
d3.js 를 이용하면, 이렇게 다양하게 가시화가 가능하다.

1.4. Data visualization using d3.js



본 시스템에선
JSON이 주기적으로
최신화 되기 때문에
cache miss rate를
실시간으로 가시화
할 수 있다.

2. Cache monitoring System 2
- PAPI with C



[
1. Monitoring tool - PAPI
2. Gathering data
   & sending query using C
3. Data visualization
]

* 실험 환경은 이전과 동일.

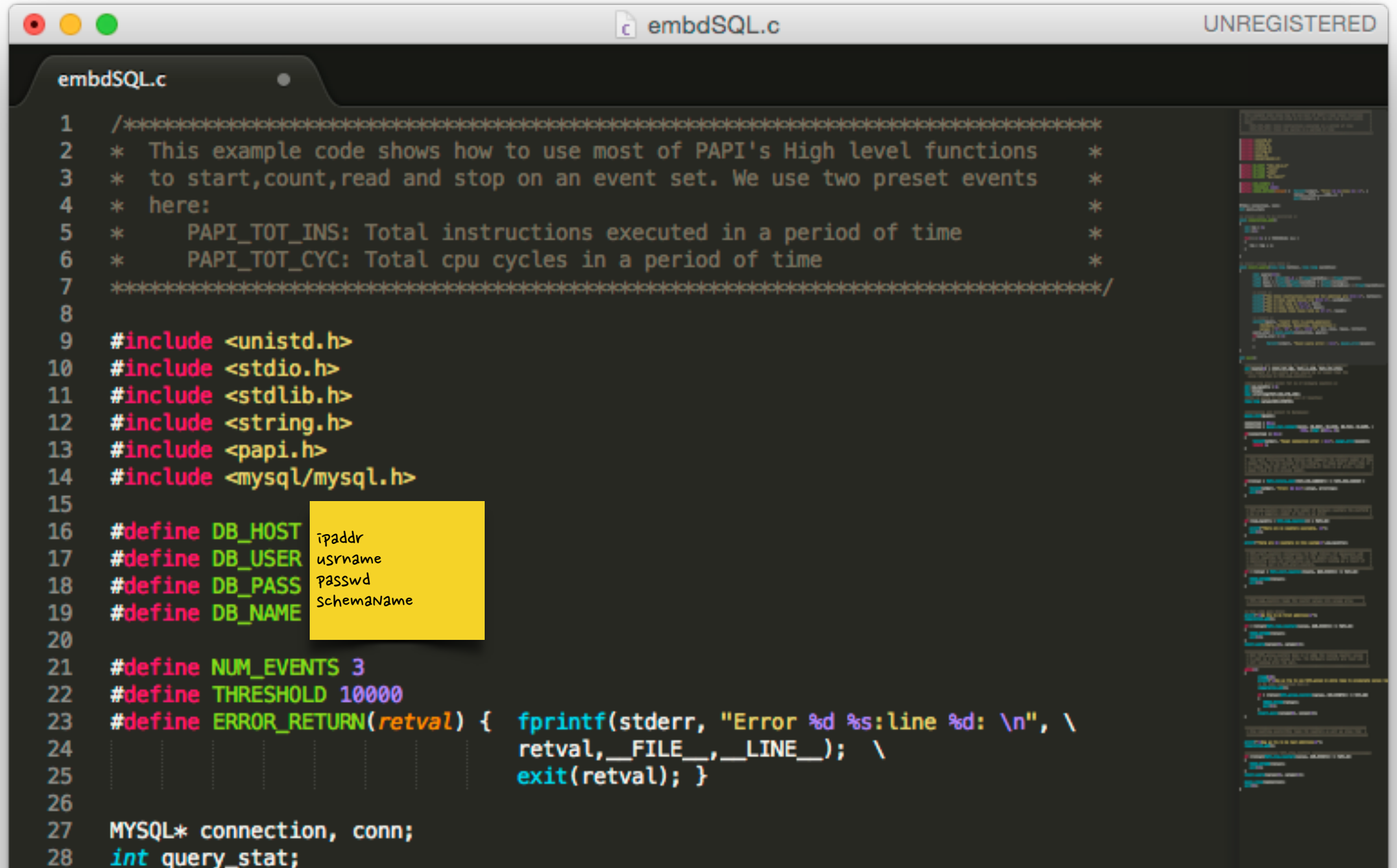
2.1. Monitoring tool
- PAPI

- PAPI provides the tool designer and application engineer with a consistent interface and methodology for use of the performance counter hardware found in most major microprocessors.

- In addition Component PAPI provides access to a collection of components that expose performance measurement opportunities across the hardware and software stack.

*PAPI, http://icl.cs.utk.edu/papi/index.html

2.2. Gathering data & sending query using C



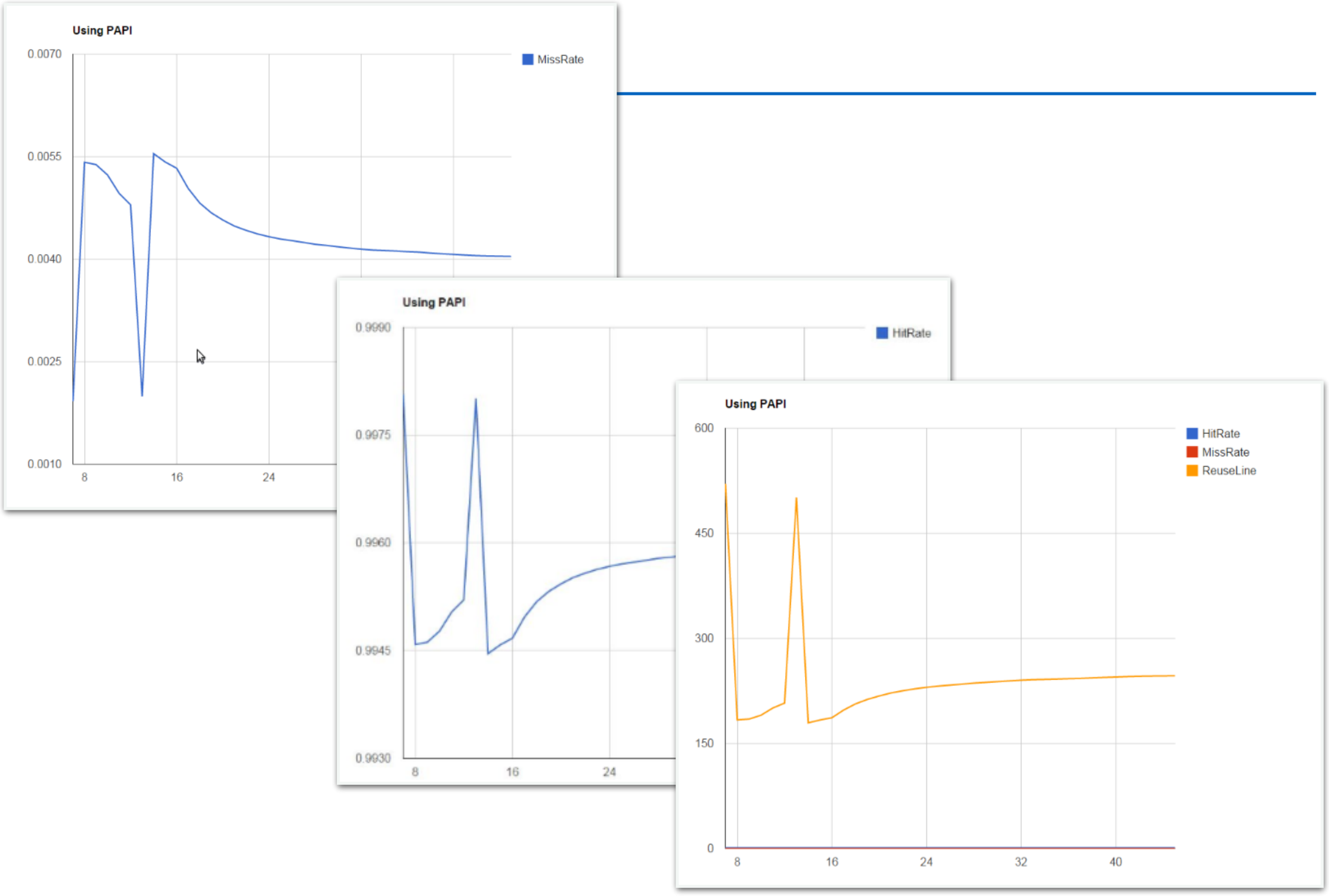
```
25                                  exit(retval); }
26
27  MYSQL* connection, conn;
28  int query_stat;
29
30  /* stupid codes to be monitored */
31  void computation_add()
32  {
33      int tmp = 0;
34      int i=0;
35
36      for( i = 0; i < THRESHOLD; i++ )
37      {
38          tmp = tmp + i;
39      }
40
41  }
42
43  /* insert values into table */
44  void insert_query(long long totInst, long long cacheMiss)
45  {
46          char query[255];
47          float hit =  (float)(1.0 - ((float)cacheMiss / (float)totInst));
48          float miss = (float)((float)cacheMiss / (float)totInst);
49          float reuse = (float)((float)(totInst - (float)cacheMiss) / (float)cacheMiss);
50
51          /* print */
52          printf("The total instructions executed for addition are %lld \n", totInst);
53          printf("The L1 data cache misses are %lld \n", cacheMiss);
54          printf("The L1 hit rate is %f \n", hit);
55          printf("The L1 miss rate is %f \n", miss);
56          printf("The L1 cache line reuse rate is %f \n", reuse);
57
58          /* insert */
59          sprintf(query, "insert into l1_cache_analysis\
60              (HitRate, MissRate, ReuseLine, Instruction) \
61              values ('%f', '%f', '%f', '%lld')", hit, miss, reuse, totInst);
62          query_stat = mysql_query(connection, query);
63          if(query_stat != 0)
64          {
65                  fprintf(stderr, "Mysql query error : %s\n", mysql_error(&conn));
66          }
67  }
68
69  int main()
70  {
```

```c
int main()
{
    /*Declaring and initializing the event set with the presets*/
    int Events[3] = {PAPI_TOT_INS, PAPI_L1_DCM, PAPI_TOT_CYC};
    /*The length of the events array should be no longer than the
      value returned by PAPI_num_counters.*/

    /*declaring place holder for no of hardware counters */
    int num_hwcntrs = 0;
    int retval;
    char errstring[PAPI_MAX_STR_LEN];
    /*This is going to store our list of results*/
    long long values[NUM_EVENTS];


    /*Initialize and Connect to Database*/
    mysql_init(&conn);

    connection = NULL;
    connection = mysql_real_connect(&conn, DB_HOST, DB_USER, DB_PASS, DB_NAME, \
                                    3306, (char *)NULL, 0);
    if(connection == NULL)
    {
        fprintf(stderr, "Mysql connection error : %s\n", mysql_error(&conn));
        return 1;
    }

/*******************************************************************
*  This part initializes the library and compares the version number of the*
* header file, to the version of the library, if these don't match then it *
* is likely that PAPI won't work correctly.If there is an error, retval    *
* keeps track of the version number.                                       *
*******************************************************************/

if((retval = PAPI_library_init(PAPI_VER_CURRENT)) != PAPI_VER_CURRENT )
{
    fprintf(stderr, "Error: %d %s\n",retval, errstring);
    exit(1);
}


/*******************************************************************
 * PAPI_num_counters returns the number of hardware counters the platform *
 * has or a negative number if there is an error                          *
 *******************************************************************/
if ((num_hwcntrs = PAPI_num_counters()) < PAPI_OK)
```

```c

    /*****************************************************************
     * PAPI_num_counters returns the number of hardware counters the platform *
     * has or a negative number if there is an error                 *
     *****************************************************************/
    if ((num_hwcntrs = PAPI_num_counters()) < PAPI_OK)
    {
        printf("There are no counters available. \n");
        exit(1);
    }

    printf("There are %d counters in this system\n",num_hwcntrs);

    /*****************************************************************
     * PAPI_start_counters initializes the PAPI library (if necessary) and    *
     * starts counting the events named in the events array. This function    *
     * implicitly stops and initializes any counters running as a result of   *
     * a previous call to PAPI_start_counters.                       *
     *****************************************************************/
    if ( (retval = PAPI_start_counters(Events, NUM_EVENTS)) != PAPI_OK)
    {
        ERROR_RETURN(retval);
        exit(1);
    }


    /*****************************************************************
     * PAPI_read_counters reads the counter values into values array      *
     *****************************************************************/

    /* Your code goes here*/
    printf("\nWe try to do first additions\n");
    computation_add();

    if ( (retval=PAPI_read_counters(values, NUM_EVENTS)) != PAPI_OK)
    {
        ERROR_RETURN(retval);
        exit(1);
    }
    insert_query(values[0], values[1]);

    /*****************************************************************
     * What PAPI_accum_counters does is it adds the running counter values  *
     * to what is in the values array. The hardware counters are reset and  *
     * left running after the call.                                  *
     *****************************************************************/
```

```c
/*****************************************************************
 * What PAPI_accum_counters does is it adds the running counter values  *
 * to what is in the values array. The hardware counters are reset and  *
 * left running after the call.                                        *
 *****************************************************************/
while(1)
{
        sleep(10);
        printf("\nNow we try to use PAPI_accum in while loop to accumulate values repe
        /* Do some computation here */
        computation_add();

        if ( (retval=PAPI_accum_counters(values, NUM_EVENTS)) != PAPI_OK)
        {
            ERROR_RETURN(retval);
            exit(1);
        }
        insert_query(values[0], values[1]);
}


/*****************************************************************
 * Stop counting events(this reads the counters as well as stops them  *
 *****************************************************************/

printf("\nNow we try to do last additions\n");
computation_add();

/********************* PAPI_stop_counters *************************/
if ((retval=PAPI_stop_counters(values, NUM_EVENTS)) != PAPI_OK)
{
    ERROR_RETURN(retval);
    exit(1);
}
insert_query(values[0], values[1]);

mysql_close(connection);
exit(0);
}

```

2.3.Data visualization



3. 데모 & QnA

1. Cache Monitor 실행
2. SQL to JSON 프로그램 실행
3. Python HTTP Server 실행
4. 웹 브라우저를 통해 가시화된 Cache miss rate 결과 확인

감사합니다.