

Fermilab



| a new state of mind

AN ONLINE DATABASE FOR THE LINAC 201 MHz POWER AMPLIFIER VACUUM TUBE

by

DIRK KOSI FRANCIS

August 9th, 2007

LINAC 7835 Power Amplifiers: Online Database

Online Database			
<input checked="" type="checkbox"/> FNAL	<input checked="" type="checkbox"/> BNL	<input checked="" type="checkbox"/> ANL	<input checked="" type="checkbox"/> LNAL
<input type="radio"/> Show ALL Tubes			
<input type="radio"/> Show Tubes in SERVICE			
<input type="radio"/> Show Tubes in STANDBY			
<input type="radio"/> Show Tubes that are being REBUILT			
<input type="radio"/> Show Tubes that are in TESTING			
- LOGIN - SEARCH - SN+REBUILD SUMMARY - GRAPHICAL ANALYSIS			

SOUTH CAROLINA STATE UNIVERSITY
FERMILAB SIST PROGRAM

TABLE OF CONTENTS

Abstract.....	3
Introduction.....	4
Fermilab: A brief history	4
Outline: The Linac	5
Outline: The Linac 201 MHz power amplifier vacuum tube (model #7835)	6
The Project	9
Preparations and Materials.....	11
Tools	11
Implementation.....	16
The Database	16
Creating the Interface	18
Results	20
Discussion.....	22
Problems faced	22
Future Works	23
Acknowledgments.....	24
References.....	25
Appendix A.....	26
Screenshots	26
Appendix B.....	29
Sample Code	29

ABSTRACT

One of the most indispensable devices critical to the running of the LINAC, is the LINAC 201 MHz power amplifier vacuum tube, which provides the final amplification that drives the five 201-MHz resonant drift-tube cavities to 4.5 Megawatts (MW) in the low-energy half of the Fermilab Linac. Because of the importance of these tubes to the operation of the Linac, the Linac staff and particularly Larry Allen, has been accumulating 7835 tube data since operations began in 1969.

The purpose of my project is to create an online database of inventory and performance information on these tubes, whereby authorized users such as the Linac staff and other observers, will have the ability to manipulate data to monitor the lifespan and operation of the tubes. The interface will be designed at the database level by using MySQL and at the interface level using a combination of HTML and PHP and to a lesser degree JavaScript.

INTRODUCTION:

FERMILAB: A BRIEF HISTORY

Originally known as the National Accelerator Laboratory and renamed in 1974 in honor of 1938 Nobel Prize winner Enrico Fermi, Fermilab has always been a hub of scientific excellence and intellectual advancement. The mission of the lab is to advance the understanding of the fundamental nature matter and energy, and opportunities never cease to exist for scientist from all parts of the globe to conduct high-energy physics research. It was founded in 1967 and Robert Rathburn Wilson was named its very first director. Since then, Fermilab has been privileged to be under the directorship of more outstanding physicists namely, Dr. Leon M. Lederman, Dr. John Peoples, Dr. Michael S. Witherell and its current director, Dr. Piermaria Oddone.

Fermilab plays host to the Tevatron, the world's highest energy particle accelerator. It boasts of a cryogenic cooling system that was named an International historic landmark by the American Society of Engineers in 1993 and is also the largest low-temperature system in existence. The Tevatron also supports the operation of two collider detectors, the Collider Detector at Fermilab (CDF) and Dzero. One of the instruments most vital to the operation to the Tevatron is the Linac.

THE LINAC

The Fermilab Linac is a negative hydrogen ion, 400 Mega electronvolts (MeV) accelerator. It includes a 25 kilo electronvolts (keV) H-minus ion source, a 750 keV electrostatic accelerating column, a 116 MeV drift-tube (Alvarez) linac operating at 201.25 MHz, and a 401 MeV side-coupled cavity linac operating at 805 MHz.

The first stage in the acceleration process done at Fermilab, takes place in the Cockcroft-Walton generator, in which hydrogen gas is turned into H^- ions by introducing it into a container lined with cesium. An electric field is applied by the Cockcroft-Walton generator, and the ions are accelerated out of the container. This is where the linear accelerator comes into play, in that it accelerates the particles to 400 MeV, or about 70% of the speed of light, and injects into the Booster, the Main Injector and the Tevatron.

The Drift Tube Linac utilized by Fermilab in the process described above, is made entirely of drift tubes and was based on the Linear accelerator built by Luis Alvarez at Berkeley in 1947. Alvarez designed the first proton linear accelerator at the University of California at Berkeley where he was a Professor of Physics.

The Linac beam is used for injection into the Booster Synchrotron and also for cancer

therapy at the Neutron Therapy Facility¹, which is located about one-third of the way down the linac.

THE LINAC 201 MHz POWER AMPLIFIER VACUUM TUBE (Model #7835)

The main function of the LINAC 201 MHz power amplifier vacuum tube, is to provide the final amplification that drives the five 201-MHz resonant drift-tube cavities to 4.5 MW in the



Figure 1. A “7835” vacuum tube

1 <http://www.neutrontherapy.niu.edu/neutrontherapy/>

low-energy half of the Fermilab Linac.

Originally manufactured by RCA for military radar applications in the 1950's, the tubes were utilized by Luiz Alvarez, as a vital component of the drift tube linac structure. To be a bit more precise, the 7835 tubes are the fifth and final amplification, from a receiving 175kW to the



Figure 2. A '7835' vacuum tube in position at a linac station

required 4.5MW at a frequency of 201MHz. They are the only tube in existence capable of providing that level of amplification at that specific frequency. They are now manufactured by Burle Industries' Electron Tube Division who also manufacture a variety of power tubes including model number 7651, and the 4616 tubes which provide two of the other four stages of amplification. Burle also supplies three other national laboratories besides Fermilab, namely Brookhaven National Laboratory (BNL), Argonne National Laboratory (ANL), and Los Alamos National Laboratory(LNAL). Most operations of the tubes, including testing and building, are done at Burle facilities, with the exception of the final power test. Burle is the sole source for new and rebuilt tubes of this type.

The tubes are serialized in a specific format:

<Letter><number>R<number>

with <Letter><number> before 'R', representing the serial number e.g. **A27** and the <number> after 'R' representing the number of times the tube has been rebuilt. This would give an example such as **A27R6**.

The rebuild numbering starts from zero and new tubes upon initial use would not have a rebuild

number attached. Each tube has its assigned serial number stamped upon an attached label as shown in figure 1.

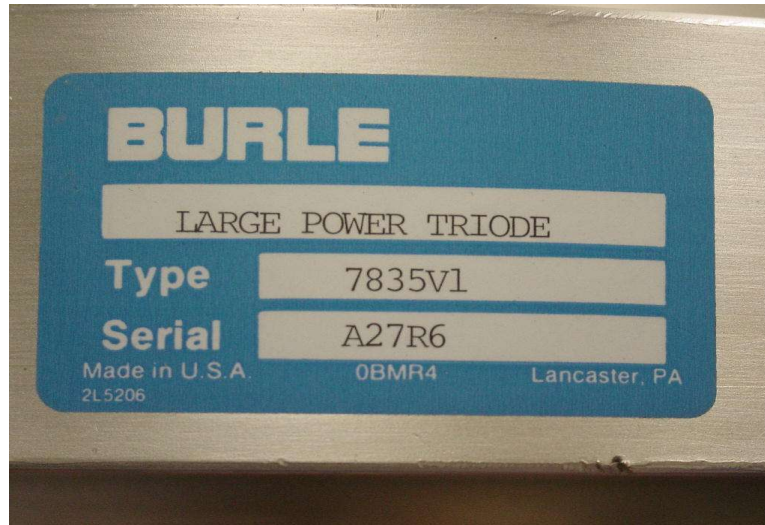


Figure 3. Serial Number Label of Burle Industries

When a tube is in an operational station, that date is recorded. Also, the date it was removed and the reason for the removal is recorded. There are 6 stations in operation at Fermilab. Five of those stations host active 7835 tubes while the sixth station (aptly named Station Seven) is that station designated to testing.

The lifespan of these vacuum tubes varies, but it has been determined that on average, they provide approximately one and one-third years of service in the Linac or 10, 500 hours of continuous operation. Tube assessment and power testing at Fermilab can take approximately two to three months. Tubes that fail in two particular ways: water leaking and sparking, can

sometimes be repaired at Fermilab. Natural failure of the tube, termed as low emission, requires the tube to be sent to Burle to be rebuilt, and this process takes approximately 4 months up to as long as a year. It has been observed from the inventory and performance data collected by the linac staff, that rebuilt tubes are indistinguishable from a brand new tube.

These tubes are vital to the functionality of the Linac, and as such, analysis and studies must be done to keep the tubes and the supply of the tubes running at maximum efficiency. Because of this, the Linac staff has collected relevant 7835 tube data since operations began in 1969. This data reveals the usage of the tubes, their status and installation history. The purpose of my project is to create an online database of inventory and performance information on these tubes that is accessible to Linac staff as well as other observers, and which can be utilized for the analysis of the 7835 tubes.

THE PROJECT:

The main functionality of the online database for the 7835 power amplifier tubes is to allow users to view and manipulate this database. The online database will be able to implement a wide variety of user inputs. The main operations are as followed:

- Display all tubes used by Fermilab, Brookhaven, Argonne and Los Alamos .

- Display tubes selected via operational status, namely In Service, In Standby, In Testing and Rebuilding
- Add Tubes, Rebuilds and Installations as well as the modification of new and existing entries
- Search feature: ability to use single and multiple search parameters to find specific tube entries
- Graphing functionalities

The database will be protected by a login validation page which limits the access of unauthorized users . All data and analysis of the tubes have been recorded via Microsoft Excel spreadsheets. This data in turn will be ported to the database but will continue to be maintained for some time. The purpose of this is to provide physical backup for the online database throughout development. There will be periodical backup of the database.

The database will be manipulated exclusively from a web interface and is a combination of web scripting languages and database applications:

- MySQL¹ - To manipulate and control the database
- PHP² - To process the data from the database before the page is generated
- HTML³ - To generate the actual interface
- JavaScript⁴ - To enhance the responsiveness

The database will be created and controlled through MySQL, while the interface to the database will be implemented using PHP. In between the snippets of HTML, there will be aspects of PHP used to manipulate the database and produced the desired results. JavaScript is also used to enhance the web pages and provide dynamic results and view where necessary.

1 <http://www.mysql.com/>

2 <http://www.php.net/>

3 <http://www.w3schools.com/>

4 <http://www.w3schools.com/js/default.asp>

PREPARATION AND METHODS:

TOOLS:

1. SQL – Structured Queried Language

This is the first and most important language used in implementing a relational database. SQL is designed to retrieve and manage relational databases and to facilitate both database object access control management and database schema creation and modification. Designed in the early 1970s, it was originally a simple data manipulation language but has since evolved into the standard way to access and manipulate a relational database, courtesy many language extensions.

The most common elements in SQL used for this project are data manipulation language (DML) statements, which are used to retrieve, insert and modify data in a database. Data definition language (DDL) statements are used to allow users to define new tables and associated elements to relevant data. Data Control language statements handle validation and data authorization based on viewing access and data manipulation. Queries can be use to relay different displays of information based on user request as well as facilitate search functions, without altering the database in any way.

The implementation of SQL used for my project is MySQL. This is extremely useful and popular for web applications, and is closely integrated into the PHP: Hypertext Preprocessor (PHP) language which is discussed next.

2.PHP – PHP: HyperText Processor

PHP represents a robust open-source development language that provides the tools and flexibility to accomplish virtually any task. PHP is an embedded language within the Apache server. This means developers can jump between raw HTML code and PHP easily. Beyond its basic syntax, PHP also boasts a wide range of interfaces allowing it to communicate with everything from other web pages, to databases including ODBC, and other programming languages such as Java or COM.

PHP has all of the features of a complete programming language (control structures, repetitive tasks, and variables) but perhaps one of its most powerful features is database access. With PHP it is possible to access over 19 different types of databases and manipulate data within those databases based on input from the user via a web page. One can also send e-mail, work with newsgroups, and actually open a connection to another web site -- and get or send data with

it. If one is already familiar with ASP development, PHP can also communicate with other server-side languages such as Java and use COM interfaces.

3. *HTML*

Hyper Text Markup Language is a computer language devised to allow website creation. These websites can then be viewed by anyone else connected to the Internet. It is relatively easy to learn, with the basics being accessible to most people in one sitting; and quite powerful in the creation functionalities it allows to user to perform. It is constantly undergoing revision and evolution to meet the demands and requirements of the growing Internet audience under the direction of the W3C, the organization charged with designing and maintaining the language.

It provides a means to describe the structure of text-based information in a document — by denoting certain text as headings, paragraphs, lists, and so on — and to supplement that text with interactive forms, embedded images, and other objects. HTML is written in the form of labels (known as tags), surrounded by less-than (<) and greater-than signs (>). HTML can also describe, to some degree, the appearance and semantics of a document, and can include embedded scripting language code which can affect the behavior of web browsers and other HTML processors.

The definition of HTML is as follows:

- *Hyper Text* is the method by which you move around on the web — by clicking on special text called hyper-links which bring you to the next page. The fact that it is *hyper* just means it is not linear — i.e. you can go to any place on the Internet whenever you want by clicking on links — there is no set order to do things in.
- *Markup* is what HTML tags do to the text inside them. They mark it as a certain type of text (*italicized* text, for example).
- HTML is a *Language*, as it has code-words and syntax like any other language.

4. *JavaScript*

JavaScript is a scripting language most often used for client-side web development. Its standardized name is *ECMAScript*, though "JavaScript" is much more commonly used. "JavaScript" is actually Netscape Communications Corporation's (and now the Mozilla Foundation's) implementation of the *ECMAScript* standard. JavaScript is a dynamic, weakly typed, prototype-based language with first-class functions. JavaScript was influenced by many languages and was designed to have a similar look to Java, but be easier for non-programmers to

work with. The language is best known for its use in websites (as client-side JavaScript), but is also used to enable scripting access to objects embedded in other applications.

The primary use of JavaScript is to write functions that are embedded in or included from HTML pages and interact with the Document Object Model (DOM) of the page. Some simple examples of this usage are:

- Opening or popping up a new window with programmatic control over the size, position and 'look' of the new window (i.e. whether the menus, tool-bars, etc. are visible).
- Validation of web form input values to make sure that they will be accepted before they are submitted to the server.
- Changing images as the mouse cursor moves over them: This effect is often used to draw the user's attention to important links displayed as graphical elements.

Because JavaScript runs on the client rather than the server, it can respond to user actions quickly, making an application feel more responsive. Furthermore, JavaScript code can detect user actions which HTML alone cannot, such as individual keystrokes.

IMPLEMENTATION:

THE DATABASE

The central repository for the data in this project is a MySQL database. Three main tables for this project are shown in Figure 4.

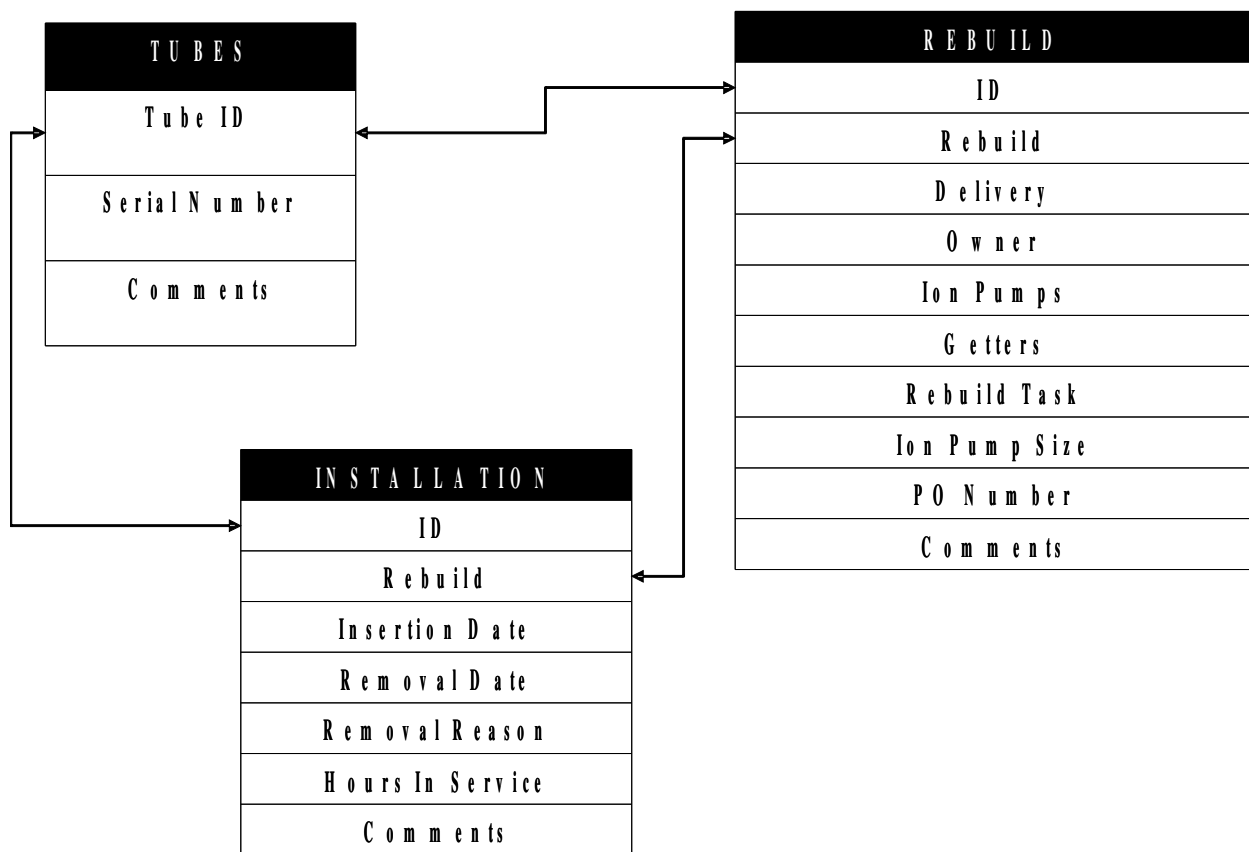


Figure 4. A visual representation of how the tables are connected in MySQL

Each tubes has an ID, serial number and comment fields, the ID being linked to the ID fields of the installation and rebuild tables. The Rebuild table contains fields for the information from each individual rebuild of the tube, such as the owner, the date that the tube was delivered and the number of times it has been rebuilt. Each rebuild is further subdivided into an Installation table. This shows the history and performance of each rebuild, like the station in which the tube was installed, the date it was installed and the reason the tube was removed.

The fields that are assigned keys are the links between the tables within the database. The ID fields from all the tables are connected as well as the rebuild fields in the Rebuild and Installation tables. These connectors are virtual - they don't actually exist within the database. However, by the use of queries and DDL statements, these linked tables form virtual tables so that all the data can be accessed easily.

The information was originally pushed into the tables from an SQL text file derived from the original Excel spreadsheet. This is done using this MySQL back up command:

mysql -u<username> -p <DatabaseName> < <name of text file>

In this case, our command would be: mysql -ulinac -p TubeData < 7835_Database, with linac being the username, TubeData being the name of the database and 7835_Database being the

name of the text file we are loading into the database. The data in the text file must be well formed SQL statements. This can and will be also used as a means of backing up the database.

CREATING THE INTERFACE:

With the database operational, the next step is to create the user interface on the web. The actual pages are created with a combination of HTML and PHP. All the tables and formatting are done through HTML coding. With the implementation of Cascading Style Sheets (CSS), pre-existing style formats such as table structure, colors and text formatting, are loading in conjunction with the actual HTML code to give the interface some degree of uniformity. Being an embedded language, PHP also seamlessly integrates into the HTML code to handle any requests that require more processing than usual. These request are actual database requests. As an example some sample code from one of the web pages that generates information from the Rebuild table can be seen in Appendix B.

Code that is highlighted in bold italics show SQL commands that are integrated into PHP code, which is itself integrated into HTML. This enables PHP to access the SQL database, execute the commands and process the data onto the formatted designed by the html code.

Some of the main PHP files used to construct the interface are listed in table 1 below.

PHP	PURPOSE
index.php	top level access into 7835 online database
login.php	allows user login for database manipulation
addRebuild.php	allows user to create new rebuild entry
addTube.php	allows user to create new tube entry
rebuild_info.php	displays information from Rebuild table
installations.php	displays information from Installation table
selection_output.php	displays the search results generated

Table 1. List of main PHP files used

While I didn't get the opportunity to use it very much, JavaScript was also somewhat used in the the pages. Although it could have a tremendous effect on the overall responsiveness of the interface, the full features and potential of the language was not fully explored. However, it did help to make the overall feel of the interface more dynamic.

The navigational bar of the interface shown in Figure 3, is an example of JavaScript implementation, whereby the user is able to control table generation onto the page by making the relevant selections using the checkboxes.

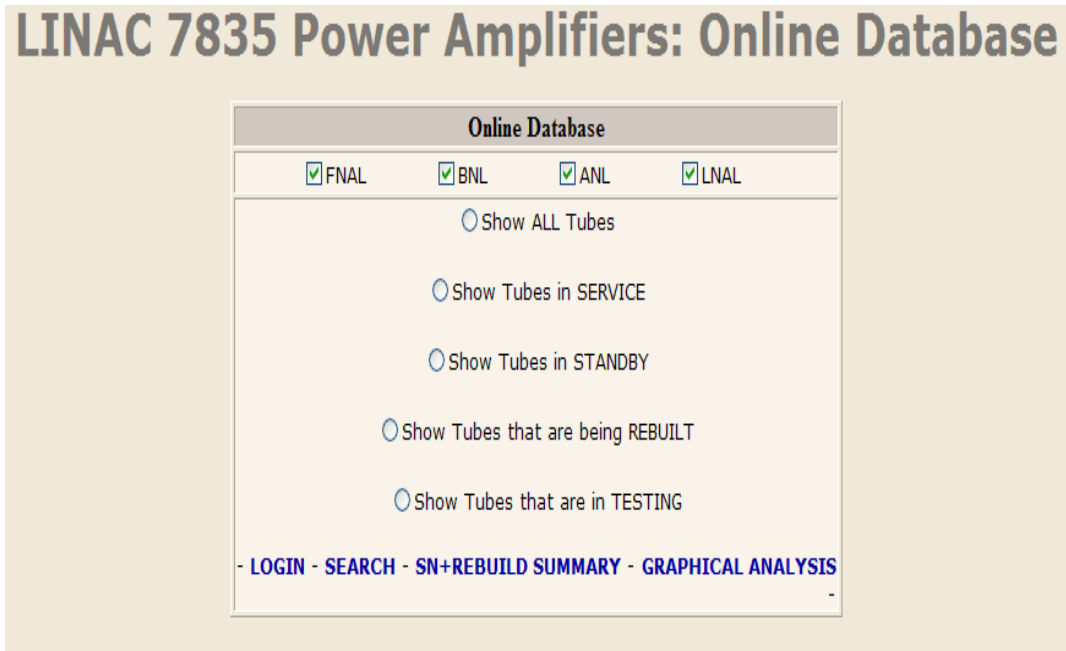


Figure 5. Navigational bar of the interface

RESULTS

The main functionalities of the online database and web interface have been completed.

The interface can perform the following functions:

- Display a list of tubes available to the user based on the status of the tubes
- Display each of the tables within the database and the fields allocated to them

- Display a variety of summary information on the tubes:
 - Tubes installed and the hours of each tubes
 - A list of serial number and rebuild based on the installation station
 - A list of serial number and rebuild based on the timespan

- Creation functionalities. It is possible to input:
 - new tubes and serial numbers
 - rebuilds (in case of a new rebuild the rebuild number is automatically assigned '0' and the rebuild task is automatically 'new')
 - installations

- Graphical capabilities. User is able to scatter plot graph based on their selection:
 - Delivery date of a serial number and rebuild vs number of hours in service
 - Delivery date of a serial number and rebuild vs number of rebuild per tube

Screenshots of the actual interface can be seen in Appendix A.

DISCUSSION

PROBLEMS FACED:

Being unfamiliar with all the languages posed a bit of a challenge but they became very useful with some practice. The main problem arose with trying to execute some of the SQL commands using PHP. I would run the command perfectly within MySQL, but when integrated into PHP, many errors would occur. This was particular prevalent in the search page. Solving the single search was fairly easy, but the real problems arose in trying to get them to work together to form a multiple search. Because there were many different combinations implemented in the multiple search, there existed the difficult task of creating just as many SQL queries to retrieve every result individually. This would have led to a lot of unnecessary coding. In order to avoid this problem, I found a way to use only one SQL command but attach SQL extensions using conditional statements based on the user's selection.

Understanding the graphical plotting functions was also a bit of a challenge, in that the SQL and PHP commands had to be perfectly in sync and then implemented into an external graphical program (GnuPlot). The actual difficulty lies in actually getting the information that the user wants drafted from database. If plotting a graph with the where one axis was the date, after pulling the date information from the database, it still had to be converted via PHP, to a

format that GnuPlot could understand.

FUTURE WORKS:

The web interface still needs some minor cleaning before testing.

Several enhancements to the plotting capabilities are imagined. As it stands, plots are predetermined and fixed. There should exist separate options so that the user can select any two values or fields and the interface should be able to generate and plot this graph.

The responsiveness can and should be improved through a more complete utilization of JavaScript.

Upon completion of these improvements, it would be beneficial for current 7835 tube data to be put into the database. Also, thorough testing by experts at all four labs needs to happen.

ACKNOWLEDGEMENTS:

First and foremost, I would like to thank God for granting me this wonderful opportunity to work in such a productive and pro-active environment.

I would like to thank my supervisor, Dr. Elliott McCrory, who from day one challenged me to think outside the box and learn things that i didn't know before. I was unfamiliar with HTML, PHP, MySQL, and JavaScript before I came to Fermilab and upon completion of this program, I believe that I have learned enough to initiate personal projects of my own using the knowledge of the languages that I gained here. My assistant supervisors, Timofei Bolshakov and Vaia Papdimitriou also lend both technically and interpersonal assistance in making sure that everything went well when Elliott wasn't around to do so.

I would like to thank the program director Ms. Dianne Engram for making sure that everything was up to standard both from the living aspect of our internship right down to the educational and professional aspect. To our mentors, Jamieson Olsen and Mayling Wong-Squires who played tremendous roles in assisting us interns during our time at Fermilab as well as Dr. Davenport for his support and tutelage, the members of the LHCROC team for their extended support, the Department of Energy for the opportunity to be a part of this program and also my fellow interns and Fermilab staff, I would like to say thank you.

REFERENCES:

Oral Communication Elliott McCrory, Applied Scientist II Integration Dept., Accelerator Div.

Meyer, Eric A. (2001). Cascading Style Sheets 2.0 California: McGraw-Hill

Williams, Hugh E., & Lane, David (2002) Web Database Applications California: O'Reilly

Moncur, Michael (2000) Sams teach yourself JavaScript in 24 hours New York: Sams

W3Schools., Online Tutorials on the World Wide Web: <http://w3schools.com>

Wikipedia., <http://www.wikipedia.com>

The PHP Website., <http://php.net>

APPENDIX A

SCREENSHOTS:

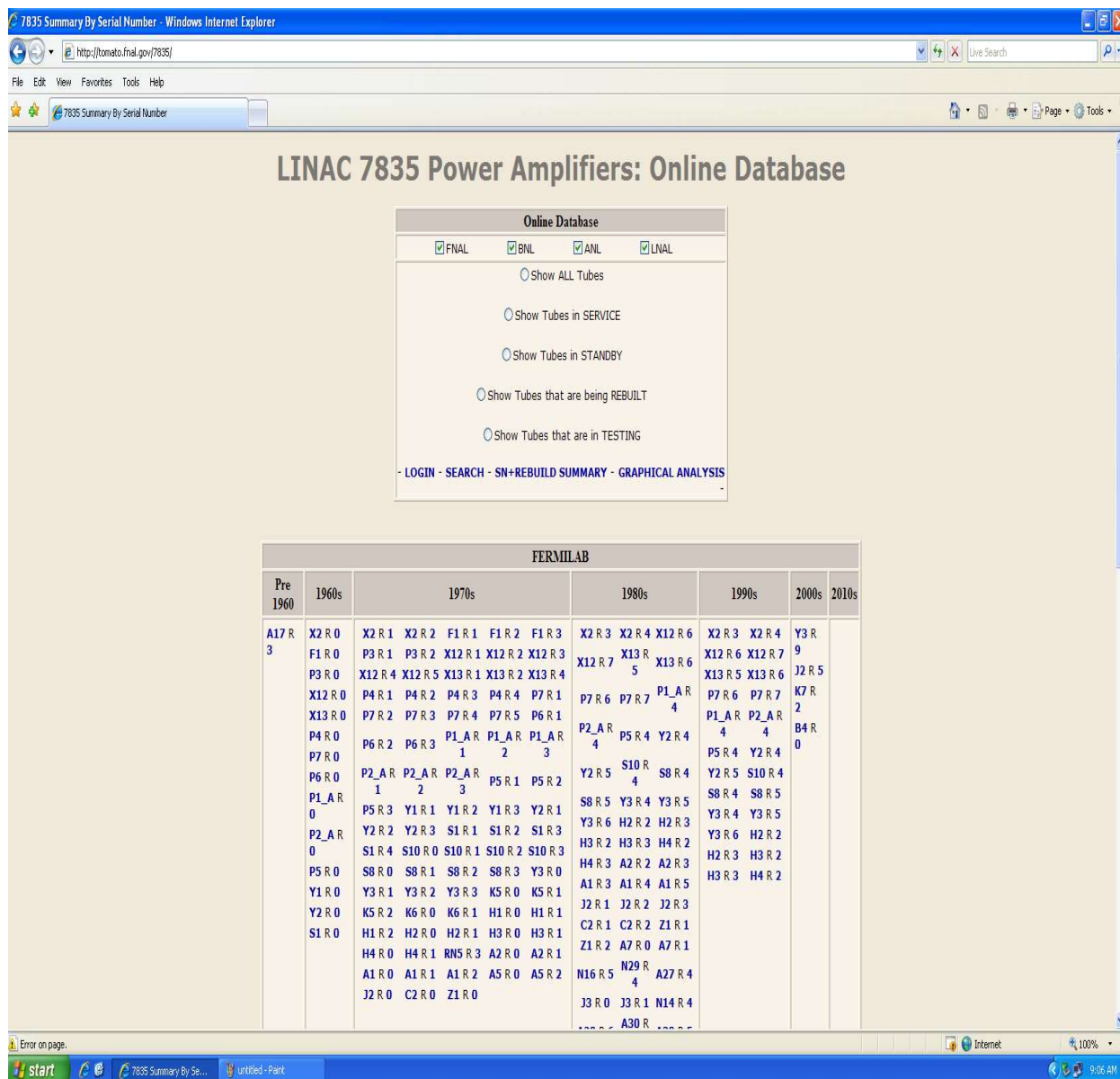


Figure 6. The home page of the 7835 online database

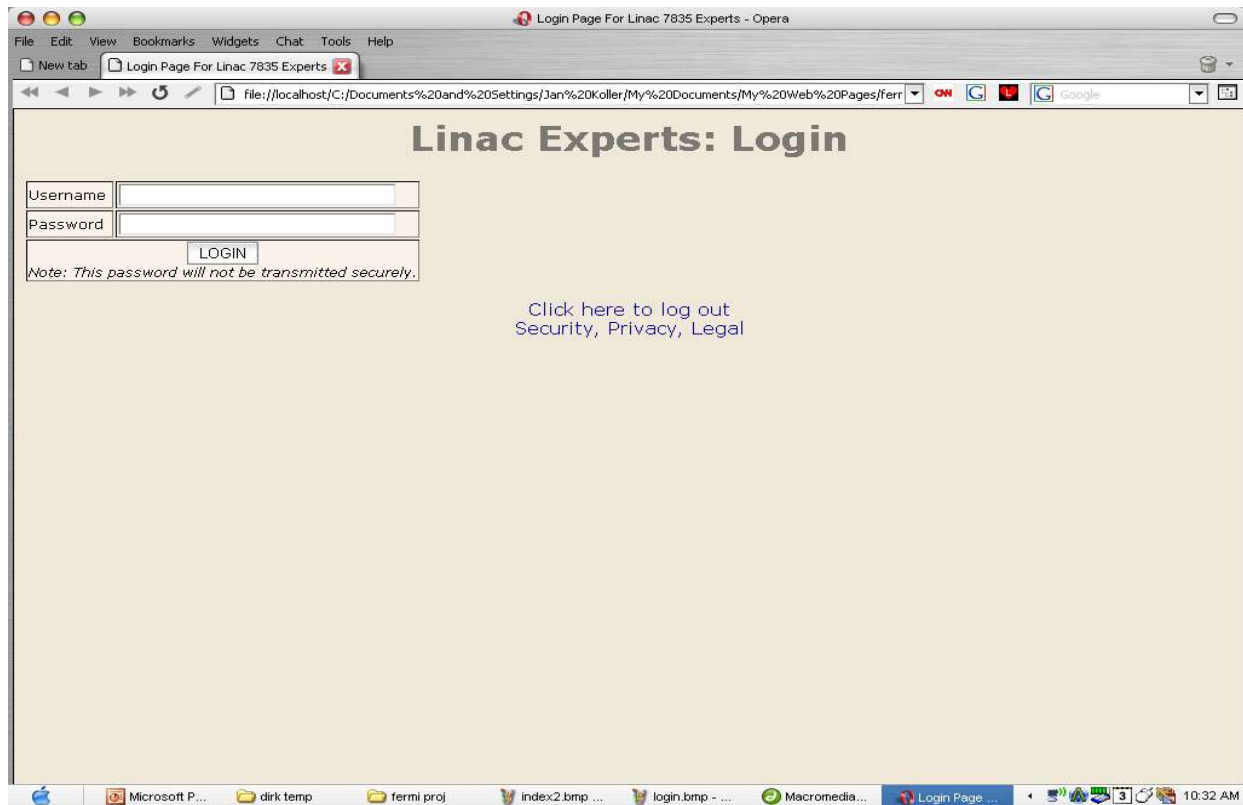


Figure 7. The login page



Figure 8. The Search Bar

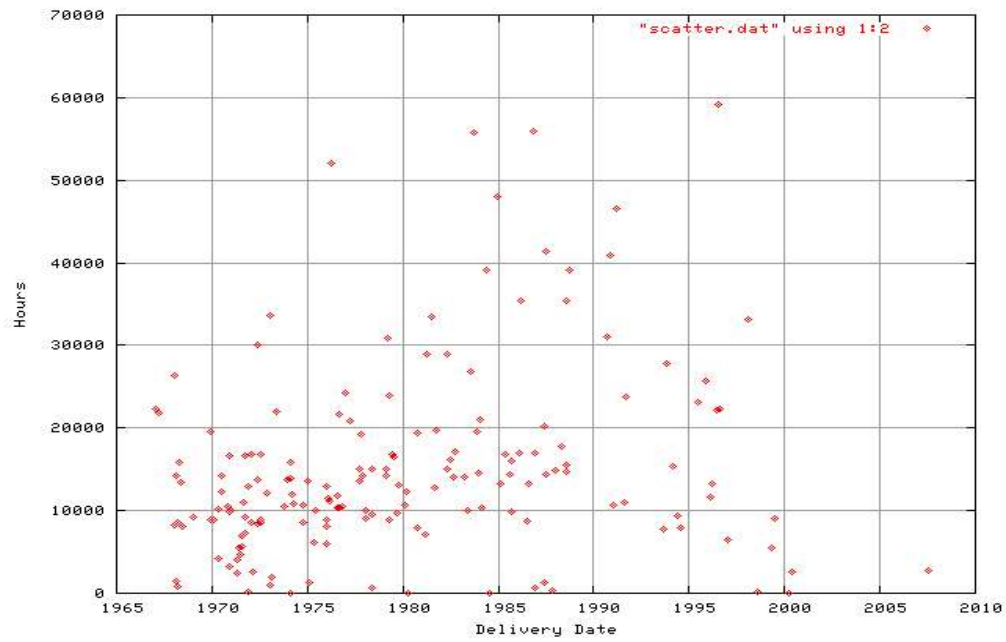


Figure 9. Plot of Tube Delivery Date versus the number of hours in service

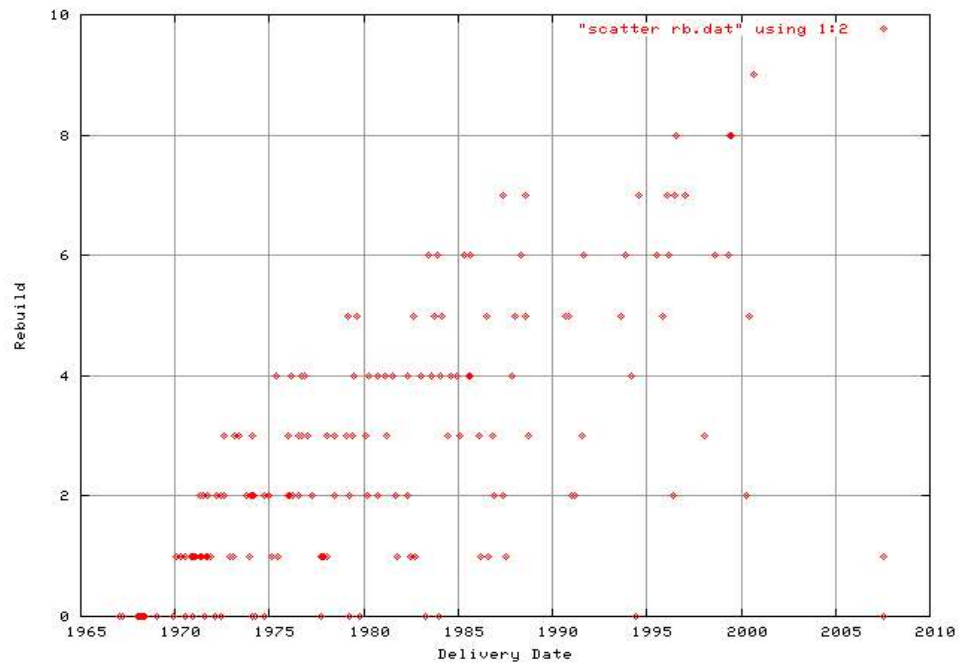


Figure 10. Plot of Tube Delivery Date versus the number of rebuilds

APPENDIX B

SAMPLE CODE (from rebuild_info.php):

```
<?php
include "functions.php";
session_start();
openMyDatabase();

$serial= "";
if(isset($_GET['SerialNumber'])) {
    $serial=$_GET['SerialNumber'];
}

if(isset($_GET['Rebuild'])) {
    $get_rebuild=$_GET['Rebuild'];
}

$username = $_SESSION['username'];

if ( strcmp($username, "tubeguy") != 0 ) {
    header("location:
login.php?nextPage=rebuild_info.php?SerialNumber=" . $serial .
"");
}

$count = "SELECT max(Rebuild), count(*) FROM Tubes, Rebuild
WHERE Tubes.id = Rebuild.id AND SerialNumber='" . $serial . "'";
$cd = mysql_query($count) or die("Select in ID Failed!" .
$count);
$row_count = mysql_fetch_array($cd) or die("Select in count
failed");
$rebuild_number = $row_count['max(Rebuild)'] + 0;

if($row_count['count(*)'] == 0){
    $tube_query = "SELECT id from Tubes WHERE SerialNumber='" .
$serial . "'";
    $tube_data = mysql_query($tube_query) or die("Selection from
ThePeople Failed!","Error:". mysql_error());
    $tube_rowID = mysql_fetch_array($tube_data);
    $id = $tube_rowID['id'];
```

```

    //$owner = "Fermilab";
}else{
    $tube_query = "SELECT id from Tubes WHERE SerialNumber='" .
    $serial . "'";
    $tube_data = mysql_query($tube_query) or die("Selection from
ThePeople Failed!".$_,Error: ". mysql_error());
    $tube_rowID = mysql_fetch_array($tube_data);
    $id = $tube_rowID['id'];
    //$owner = "Fermilab";
    $rebuild_number++;
}
?>

```