



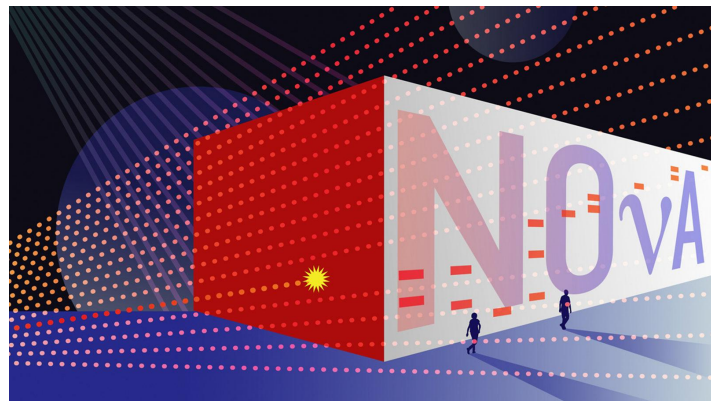
# Hardware DB Conceptual Overview

Stephen White

May 2022


# Hardware DB, A Bit of History

- HWDB was originally developed and created for NOvA about 2008-ish.00
  - By Dennis Box, Margherita Vittone and me. With guidance from Jon Paley.
- It was created to track a specific set of parts for NOvA and some tests done on them.



- About 2015 it started being adopted by other experiments including:
  - Mu2e, Icarus, SBND, Proto Dune, Ash River (currently under development)
- Known Issues with HWDB
  - Supporting a new experiment means creating a new schema from scratch.
  - Adding a new type of part/test requires developing and adding a new, distinct table or columns to the schema.
  - Good at tracking current state but little to no historical data was kept.
  - Very limited API allowing uploading data only.
  - Utilizes original, very early, web technology which will not last for the life of DUNE.

# Dune Expands the Requirements

- The full current state as well as past history must be available for each item.
  - Requires the complete test history for every item, not just the last one done on item or a history on a few items.
  - Robust html access for queries, inserts and updates.
- 
- Ability to see what an item is connected to or what is plugged into it.
    - Both current and past history of all connections.
  - Support for storage of associated documentation and or photographs.
  - Create a unique physical identifier (PID) for every item as well as provide for the generation of their labels. i.e. bar codes.
  - For a complete list of requirements: <https://docs.dunescience.org/cgi-bin/sso/ShowDocument?docid=23333>

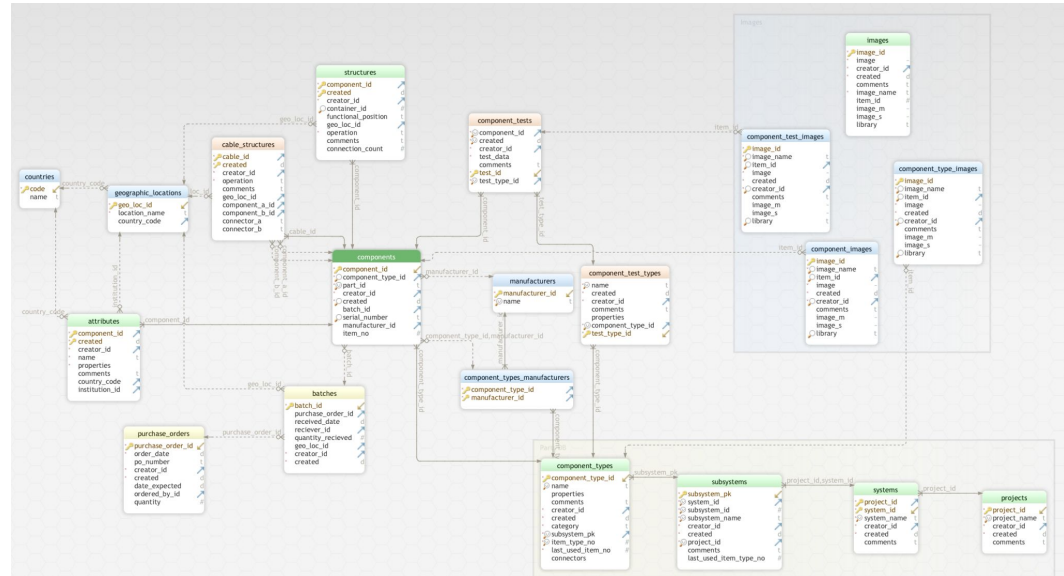
DUNE requires a very large set of discrete types of items to be tracked making creating individual tables for each item extremely difficult. Well, impossible.

# Hardware Database for DUNE

Hardware Database supports the complete life cycle of each item in the DB for the experiment as a whole.

- Manufacturing / Procurement
  - Manufacturer created a component
  - Where it was created
- You describe what items is to be stored in the DB, the data to be stored for it. As well as what items are allowed to be attached to it.
  - No longer need a developer to add a table for each new type of item.
  - You create a definition of what that item is, a pattern if you will.
  - Item data is entered according to the pattern.

Simplified view of Schema. (Sorry for the fuzziness.)



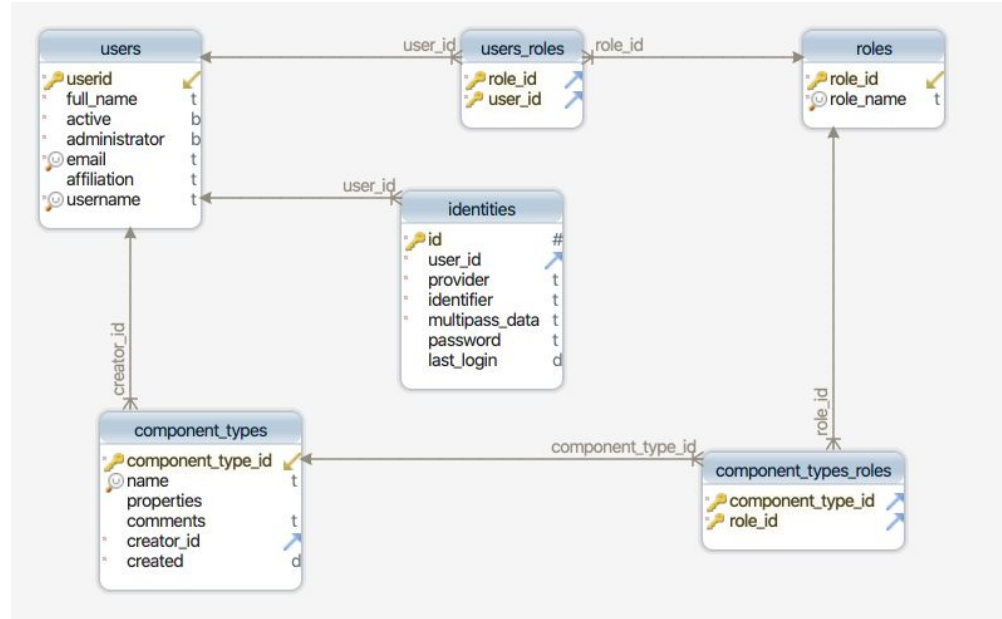
Actual readable schema:

<https://cdcvs.fnal.gov/redmine/projects/components-db/wiki>

# Hardware Database for DUNE

- Versioning is fully supported with a history of all changes.
  - Displays the item according to the version in effect at the item's creation or last update.
  - The entire history of “patterns” and each item is available.
- Testing and Quality Control
  - Create any number of tests for each type of item
  - Run any test and store its data multiple times. There is no limit.
  - View the entire test history.
- Support for Documentation, Photographs, URLs.
  - Can be tied to the Patterns, Items and Tests.
- A complete, secure, REST API is available for most of what the forms do.

## Security Tables



# Accessing the System

- The page the production link is on: <https://dbweb0.fnal.gov/>.
  - You can also access the development system
- All DUNE analysis experimenters have read only access.
  - Requires a FNAL services account
  - If you do not do analysis, you may need to be manually added.
- Login using your Fermilab Services account/password.
  - We support the lab's Single Sign On (SSO).
  - Non-FNAL accounts are not allowed.
- Security is provided by
  - Creating a role for one or more component types
  - Adding users to roles.
- Data can be entered through web forms or a REST API.
  - The API requires a CILogin certificate for security.



## Components DB

Home

Batches

Cable Structures

Component Types

Components

Geographic Locations

Images

Manufacturers

Purchase Orders

Structures

Admin <

Logout

### Overview

The components database tracks parts and what other, if any, part is attached to it. The database is designed around physicists describing what each part is and how those parts are fit together. First one must create a "type" which defines what a part is and the data collected for it. Then the data is entered for a specific "type". Tests may also be defined for a specific product "type", and data entered for each physical piece. Each physical part tracked must have some external id, which is unique across the experiment, entered with that part.

### Vincit qui se vincit.

He conquers who conquers himself.

### REST API

A REST API is available for retrieving and adding data to the database. Using the API requires a certificate. It also requires that you be recognized as a member of DUNE. Be aware that FERMI only accepts CILogon certificates.

[REST API](#)  
[REST Tutorial](#)

### Database Schema

This product is created over a PostgreSQL database. Two views of the schema are available. One which shows all the tables. There is also a second simpler view of the schema which does not show the security tables and their connections.

[Full Schema](#)  
[Simplified Schema](#)  
[Database Conceptual Overview](#)



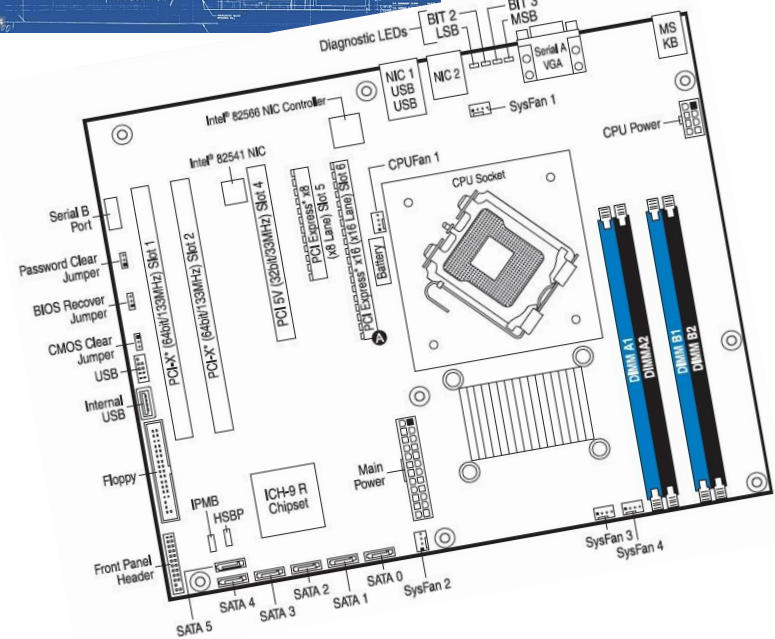
# The Big Three

This system is built around the concept of

- Component Types
- Items
- PartIDs.
- This is integral to all web forms, APIs and database tables.

# Component Type

- Component Type is simply a PATTERN, where you define what DATA will be collected for a type of item.
  - This is a virtual construct.
  - It is used to display a web form for specific to that type of item. It is also used by the APIs.



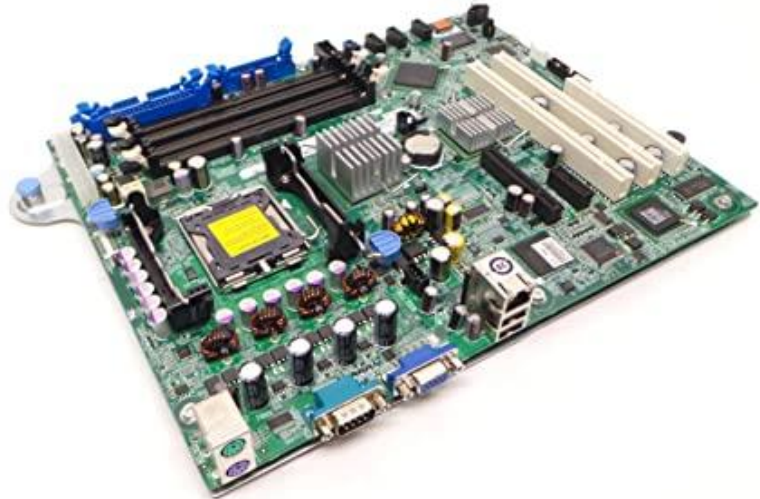


# Items and PartIDs

- An item is simply a physical piece of real world equipment.
- Every Item must be barcoded with a PartID.
- A PartID is a unique identifier defined according to DUNE specifications.

- <https://edms.cern.ch/document/2505353/3>

- Every item must have a PartID attached via a barcoded label.



# Moving Forward...

Jim Stewart will be giving an introduction to the Parts Identifier.

Hajime Muramatsu will provide training on setting up Component Type definitions.

Experience has shown that the success of the Hardware Database always depends on the willingness of the physicists to enter the data. Once a physicist leaves the experiment all unentered historical data is lost forever.