Manage People, Not Userids

Jon Finke – Rensselaer Polytechnic Institute

ABSTRACT

Despite the title, this is not about managing people, but rather managing the enterprise data about the people, especially in defining the relationship between a person and the organisation and controlling functions based on that relationship, or what some people might refer to as identity management.

Single sign-on is an attractive goal for many organisations. When you include parking gates and badge readers on building entrances, the problem gets even more interesting. As we expand our deployment of wireless access points and publically accessible network jacks, the need to require authentication for access to our virtual world grows stronger. With the need for authentication, so grows the demands on the systems that provide authentication and authorisation, especially in the area of managing who gets access and revoking that access at the appropriate time. Concurrently, with the rising interest in physical security of our facilities, the need for authentication and controlling access to our physical world is also growing. This also requires tools and systems to manage the people and their status and privileges.

Both of these issues share many common attributes and can be well addressed by merging them into a single system to manage people information, and from that, access to the virtual (network) world as well as the physical world. By combining these projects, we are able to take advantage of the mandate (and administrative support) to identify all of the people on our campus to provide physical access control, and so, manage our virtual world. We will also attempt to define a somewhat generic or standard methodology for doing this with our particular business rules and requirements confined to a few limited and specific areas.

While the technical issues are challenging, the more daunting task comes with negotiating the institutional politics and getting adequate ‘‘buy in’’ from the appropriate departments to provide the people and resources willing to operate and use the eventual technical solutions. This paper discusses both the social and technical aspects of those solutions.

Introduction

There have been a number of systems developed over the years to manage user accounts, *Moira* [12] in 1988 (part of MIT’s Project Athena), to *Accountworks* [2] 10 years later and many others. In many ways, this process is pretty well understood, and many mechanisms exist for taking a user account and getting it the end system, be it Windows 2000 [7] and others [10, 11, 3]. In *A Retrospective on Twelve Years of LISA Proceedings* [1], the authors identified 23 papers dealing with account management. Under the section *Future Research Opportunities* they write:

> Surveying account creation practices would help identify why no tool has evolved as superior despite many papers on this subject. We believe this is because of unrecognised differences in the requirements at each site. With all of the requirements explicitly described, it should be possible to build a universal tool.

Like the organisations cited above, we here at RPI, have been developing our own system (Simon) for automatic creation and management of computer accounts. Since Simon’s inception in the early 1990s, it has evolved to maintain our telephone directory [5] and ID card and parking control systems, as well as providing a consistent data feed [9] of people and directory information to a number of other systems on campus. This paper is in part a case study on how we addressed both the political and technical issues.

The basic objective of all of these systems is to automatically create and maintain computer accounts for all students, employees, contractors, etc. as needed. Given a good data feed, this is basically a technical exercise, which has been documented in the references cited above and many other places. But in all of these cases, the differences come from that prerequisite; the good data feed (or feeds), ideally from the system of record for that data element. While every site is different, I feel that they all share some commonality and I hope to outline some approaches that may be applicable to your specific site. One of the challenges we face, is that for the most part, no one outside of the IT department really cares all that much about solving this problem. After all, it is IT’s problem to solve, and it doesn’t really impact people outside of the IT department at all. We found that by expanding the scope of our project from just computer accounts, to other ‘‘people’’ related activity such as the phone directory or the ID card system, we were able to get

the interest and assistance of some of the key players outside of the IT department.

From a technical standpoint, while I don't propose to describe the universal tool in this paper, I do hope to describe a close to universal methodology that can be applied to most educational or corporate[1] environments. As suggested in the above quote, the problem is to define a generic enough tool that it can handle the requirements of different installations, while still allowing the easy application of the appropriate business rules with minimal changes to the code base.

For the most part, dealing with data feeds from Human Resources (for employee data) and the Registrars office (for student data) is well understood. The challenge we face is dealing with the "none of the above" categories.[2] In an ideal world, we would find a way to push that maintenance load out on the departments and offices. From the IT standpoint, we don't really have the juice (we can't mandate cooperation) to get departments to maintain status information, but by merging the computer account management with the more general ID card and directory processes, we are able to get sufficient interest to maintain the status information.

In this paper I hope to first describe how we addressed the political and social issues for implementing this system. I will discuss some of the arrangements that were made, and some of the arguments that were used to convince people in other departments and divisions that they needed to be part of this project and provide the operational support to make it all work. In addition, I will describe some of the philosophy and guidelines we adopted dealing with sources and maintenance of information, and some of the pitfalls we encountered.

Moving from the human realm to the technical realm, I will next describe a general schema for managing "people" (and "department") information, and given that, then describe a general, adaptable system for managing and delegating all of the oddball categories of people, and provide code examples that do NOT have our business rules built in, but are easily adaptable for any site. From this, we derive a general mechanism to evaluate and integrate the "status" of a person, in an easily adaptable and flexible manner. This in turn drives the business needs of the enterprise, be it computer accounts, ID cards, directory entries, or whatever the access and authentication needs are.

Obtaining Buy In from Others

The fall of 2003 brought to the fore, the issue that would finally get serious administrative support behind the unified people status project, and possibly a hot button topic just about anywhere, Parking. In our case, it was the installation of parking gates. Now it really mattered if someone's ID card expired – they would be denied access to our campus. In point of fact, there were at least 40 different departments or units identified as "stake-holders"[3] in this project. The parking lot access project let us bring some focus to the project and get a reasonable number of key people into meetings.

Authoritative Data Sources

As part of an earlier data feed project, we found it useful to identify the authoritative data source or "system of record" for each type of data. Despite having both student and employee information in our central administrative database (SCT Banner),[4] it quickly became obvious that although most people were there, not all people could be found there. Thus after much discussion, it was decided that although Banner would remain the system of record for student and employee data, the official, campus-wide system of record for people data was our locally developed Simon system. As each additional group or type of person came along, we had to identify the "system of record" for that part of the data feed.

Identifying systems of record is an iterative process. Once you identify a data element, and who in the organisation owns and maintains it, you then need to get them to agree to supply you with an on-going feed. This may be complicated by then discovering that the authority doesn't really have all the data you need. For example, we were finally able to identify the authority for building names, by asking the University President at an open "town meeting". This approach may not endear you to the lucky selectee, but it does help the process move forward. We then discovered that although we could get the "official" names for buildings, they didn't have the more common names[5] or abbreviations for buildings. Abbreviations are important – we cut the size of our campus directory from 220 pages to 203 simply by using building abbreviations! We are still working on resolving this issue completely.

Our current strategy for picking up the smaller groups and data elements, is when someone asks for it, we ask them who is authoritative for that group. For example, when the Provosts office asked for a mailing list for emeritus faculty, I asked them who was in charge of that information. They admitted that they

[1] Although I am starting from a university environment, we are also a major employer, with an active HR department. I am not going to address the ISP world.

[2] Such as contractors, vendors, adjunct faculty, visiting scholars, temporary employees, retires, conference guests, research collaborators, emeritus faculty, consultants, Ice Hockey officials, model railroad club members, etc.

[3] Stake holders included Parking and Transportation, Public Safety, Human Resources, Physical Plant, Contracts and Grants, the Student Union, the Registrar, the Provost, the Library, Sodexho (food service), Purchasing, and a number of others.

[4] SunGard SCT – http://www.sct.comhttp://www.sct.com .

[5] My office is located in the "Alan M Voorhees Computing Center," which everyone refers to as the "Voorhees Computing Center" or the "VCC".

were responsible, at which point I offered them a tool to maintain that list, that would also be able to give them mailing lists, interface with the campus mail room distribution system and feed into the ID card system. They agreed, we did some training and that part of the puzzle dropped into place. It was helpful to have a tool ready to go when they asked.

Sharp Edges

Even with a good data feed, you still have to watch out for some differing expectations between the users of the system and the mechanical reality. This issue was driven home recently at the end of the fiscal year. From a database standpoint, a data value is often a time/date value. If you have a time component, there is almost 24 hours between a termination date of 06/30/05 and a start date of 07/01/05. Thus, we had a bunch of folks who terminated on June 30th just after midnight and their new position did not start until the following day at midnight. As a result, about 40 people lost their parking access when they shouldn't have. We will be implementing a ''look ahead'' function that when someone's job status is terminated, we will look ahead to see if they are starting a new job within the next two weeks.

Empowering the Process

Although we had identified Banner as the system of record for employee data, and the Human Resources department as the maintainer of that data, we discovered that although Banner does a fine job tracking eligible employees and payroll, it wasn't ready to handle real time operations. From a database perspective, a potential employee is entered into Banner with some demographic information, a department affiliation, etc. This can take place weeks before their actual start date, and not everyone entered here actually starts as an employee. So rather than key off of this, we would wait until payroll actually entered them and assigned them a job and job classification. When all this was doing was driving the telephone directory, we could tell folks to wait a few days and the new folks would show up in the online directory. But when this entry was required to issue the person an ID card and a Parking transponder, waiting a few days was not acceptable.

Another requirement for employment, was the completion of the IRS I9 form. Historically HR had problems getting folks to come up to their office, which was located some distance from the main campus, to actually sign these forms. So we wrote a little application for HR staff, and made a deal with them. If they would use this application to mark when a new employee was ''OK'', (and indicate faculty or staff status), we would refuse to issue an ID card, or parking transponder or computer account until HR set the flag. With this in place, new employees had to visit Human Resources before they could really do much of anything. This also encourages the departments to follow the HR hiring process.

The other end of the employment process is employee separation. This impacts not only the IT world, but many other departments. When an employee leaves, the key shop needs to collect their keys; if the person was responsible for hazardous materials (gas cylinders, chemicals, etc.), someone else needs to take custody of it. In addition, separation will impact benefits such as pension and health care. One recurring problem we had seen, was when departments failed to renew employees on fixed term appointments, and essentially, their job ran out. In order to deal with this and other issues, Human Resources formed a ''Separation Process'' committee. One of the outcomes of that process was an automatic process to generate notification to departments of who was coming up for separation. By connecting with this process at the database level, we are able to synchronise our electronic world with the ''official'' employment status. Now if a department ignores these separation reports, a lot of things happen automatically. This process gave us a better employee data feed, and puts more teeth in the HR process.

Another example of cooperative projects is the management of Emeriti information mentioned above. The Provosts office agreed to maintain the lists, which feed into the ID card and directory systems. The tool they use also lets them get mailing lists back, and even the ID card photos.

One of our remaining challenges, is trying to eliminate or at least reduce the number of ''Director Specials''. Sometimes when a new person comes on board, ''helpful'' people high up in the food chain would try to grease the skids by arranging for the new person's email account to be ready when they show up. In the past, they would call their favourite director in the IT division, and get them to set up special computer account. This would sometimes result in a second account being created once the normal HR process went through, to make matters worse, this second account would be the official account, included in the department mailing lists, and so on. With wide scale use of card readers and parking passes, even if the person got their email account, they still couldn't get an ID card, get parking etc. While we might have to jump when the CIO says frog, the parking office doesn't! We are gradually convincing these directors and VPs, that the best way to get a new person on board, is to get the proper paperwork to HR, and from then on, the process is automatic and painless. We have on occasion walked the paper through HR, and the other systems, but even that extra effort is much less work than trying to unsnarl manual entries that entered the system in the middle.

Data Sources

One of our goals in the design and evolution of our system, was to let other people and departments do as much of the data entry and management as possible. Ideally, they would already be doing this work,

and we would just be able to tap into their systems. Although they may have been hesitant at first, the other departments seem to like this approach, if for no other reason, is that it makes them clearly the steward of their own data and individuals are encouraged to go to the "right place" to get their records and status cleaned up. Some obsolete fields have been omitted from the table descriptions that follow.

General People Schema

The key table for handling people in our system is aptly named PEOPLE(Figure 1). It has evolved over the years, with new fields being added and others being made obsolete, or on their way to being phased out. As we refine our data model, this table is moving back to it's intended role of identifying people, and other information is being moved to other tables.

The PEOPLE table shows it's heritage as the driver for creating computer accounts, and some of the data models of earlier systems used to feed it. One of the annoying lacking, is the the absence of a Middle_Name field. When upstream systems started providing a middle name as a unique field, it was automatically merged with the first name to be stored here. Later on, a Preferred_First_Name field was provided in another table, which can be set by the individual for use in directories and displays. There is a lot of code that know the old way of doing things, and fixing this will be a non trivial task (and is waiting for some other things to be rewritten).

Another problem we have, is with the Student and Employee flags. These are[6] used to control the creation and expiration of computer accounts. There have been special cases where someone needed an account, and one of these flags have been set via other means. However, we have run into cases where staff in the field see these flags set and think that the person in question is a current employee or student. I look forward to the day when these flags are gone and we rely on the status values.

The xxx_UID fields really don't belong here, and they do imply some policy, like restricting people to just one student and one employee account, but also to just one guest account. In practice, a student or employee should NEVER have a guest account, and there are cases where someone may need more than one guest account, or these accounts should not be reused, such as for a contractor who works for one department, leaves, and returns to work for another department.

In cases where the person in question comes from Banner (which is all students and employees), we include their "PIDM" – this is the primary key used by Banner. The general practice is that once a person is in Banner, all of their identifying information (name, ID Number, SSN, DOB and Gender) comes from Banner. The inclusion of the social security number is disturbing to some people, but it has proven very useful at the ID desk to identify people who are returning, and are already in the system from those who are truly new to Rensselaer.

One of the ongoing challenges of maintaining the data, is dealing with people who get entered into the

[6] We will be changing this, and using status values instead.

ID	Number	Primary Key used to identify a person (or entity). Referenced by many other tables.
Lastname	Varchar2(60)	The family name or last name.
Firstnames	Varchar2(32)	First and Middle Names.
Prefix	Varchar2(8)	Prefix for a name – generally unused.
Suffix	Varchar2(8)	Formal suffix to name, such as "Jr", etc.
Student	Varchar2(1)	Indicates that person gets a student account – being phased out.
Employee	Varchar2(1)	Indicates that the person gets an employee account – being phased out.
Guest	Varchar2(1)	Indicates that the person gets a guest account – being phased out.
Student_Uid	Number	The unixuid of the student account, if any.
Employee_Uid	Number	Unixuid of the employee account, if any.
Guest_Uid	Number	Unixuid of the guest account, if any.
ISO_Number	Number	ISO Format ABA card number – ID card number.
FAIMS_PIDM	Number	Primary key for person identification in Banner.
Spriden_Id	Varchar2(9)	University ID number (Rensselaer ID Number – RIN) – assigned in Banner.
Spriden_Activity_Date	Date	Date of the last activity in the person ID table in Banner.
Budget_Str	Varchar2(32)	Default budget number for non student charges.
SSN_Str	Varchar2(9)	Person's social security number.
Birth_Date	Date	Date of Birth.
Gender	varchar2(1)	Gendor of person.
Clean_Lastname	Varchar2(64)	A "cleaned" version of the lastname (all lowercase, with spaces and punctuation removed) to aid in searching.
Clean_Firstnames	varchar2(32)	A cleaned version of the firstname.

Figure 1: People description.

system twice. We have a merge tool that helps us shuffle records between the different versions of a person, and marking the ''bad'' record so it is not re-used.

Students, Employees

For student data, we have a process that compares the Registrar's list of students with our own copy of the student list. Along with daily runs, we can update a specific student's records via a web application. We had the added challenge, in that we actually have (or had) two registrars, one for each of our campuses, and they had different practices in marking ''active'' students. This difference is demonstrated in the discussion of the Person_Status view below. Student processing is fully automatic now. Student with account, status or ID card issues are referred to the Registrar. The Registrar also notifies us when they recode large groups of students, so we don't panic when 1200 students drop out suddenly (due to graduation.)

In a similar manner, employee data is checked once a day with the Human Resources information in Banner. We can also do updates of a specific individual via a web tool. The Banner data for employees isn't as clean as we would like, which required a special tool for HR to use to indicate when employees really start. Fortunately, this process has provided some benefits to HR, that they are quite willing to use the tool (described previously). We did add a safety check that stops employee processing if more than 10 percent of the employees are marked as being changed.

Non-Traditional Sources

There are some groups of people, such as emeriti and retirees who are already in the system, but no longer active employees. Since we knew that these people were already in our system, all we needed to do was maintain a list of them. As part of our directory project, we had a tool available to manage mailing lists. Some minor changes gave us a version of the tool to maintain lists of emeriti and retirees. These mailing lists were then fed back into the system to provide additional status entries.

Guests

Although we had students and employee status well in hand, that still left ''none of the above.'' With the deployment of RFID cards and Parking transponders to provide access to campus buildings and parking facilities, the problem of maintaining status information for ''ID Guests'' became very real. Our original practice of creating a campus computing account (which as a side effect, would generate an ID card number for them) was not going to handle this.

One of the principles that we adopted, the 13th amendment to the Constitution notwithstanding, is that everyone had to be ''owned'' by an on campus person or department. For regular employees, Human Resources plays this role and for students, the Registrar handles it. But for everyone else, we needed to designate an ''owner''. In many cases we were able to use our existing departmental directory administrators to handle this role, and for other cases we created new ''departments'' and assigned administrators there. These administrators are assumed to know which guests are still with their department and can expire or renew them as needed. For some categories of visitors, they can also control if that person gets a computer account and if that person is to be included in the online directory.

The primary point of contact for guests, is our ID card office. The folks there have a tool that lets them make entries in the ID_People table (Figure 2). Entries here are automatically propagated into the People table, and are also used to provide status values. Some of the fields have been omitted for brevity including some personal ID fields (SSN, DOB, etc.), but the key ones are included.

Some entries in the ID_People table will have a Sponsor specified. The sponsor will point to another person, and this used used in the case of dependents or spouse, or a personal care worker; someone whose relationship with Rensselaer is a direct result of some other member of the community. This allows us to automatically ''expire'' dependents when their sponsor is no longer eligible to sponsor dependents.

If an entry doesn't have a sponsor, then it will have an Affiliation_Id. This points to a department. Originally, the department tree for the ID Guest system and the department tree for directory were different. We have since merged them and although we still have two tables, the tools keep them synchronised. One of the big benefits of merging these two trees into one, is that the directory departmental administrators can maintain ID guests as well.

Person_Id	Number	Primary key – matches People.Id
Lastname	Varchar2(64)	The last name.
Firstnames	Varchar2(32)	The first and middle names.
SSN	Number	The SSN or Rensselaer ID Number if available.
Entry_Type_Id	Number	Identifies different types of entries, see Id_Entry_Types.
Sponsor	Number	The People.Id of the person who ''sponsored'' this guest, if any.
Requestor	Number	The People.Id of the person who requested this entry.
Affiliation_Id	Number	An identifier of the department sponsoring this entry.
Expiration_Date	Date	Date this guest entry expires (can be renewed).
Dir_Flag	varchar2(1)	Flag to control inclusion in the campus directory.

Figure 2: ID_People description.

The ID Card Office staff use the tool to enter a new person into the system. The application attempts to locate the new person in the existing database, searching by name and by ID number. We really want to avoid duplicating people. We can still make the guest entry, but the Person_Id will match the original record. The next step is to select what type of guest they are, which then determines the additional questions they need to answer. The choices are contained in Id_Entry_Types (Figure 3). The xxx_Required flags tell the application if it needs to prompt for those fields. In the cases of sponsor and requestor, the person must be in the existing database, and in the case of a sponsor, the sponsor's status must be one that allows them to sponsor a guest. (How this is done will be discussed later). If the entry type has an affiliation id, that tells the application the root of the departmental tree it should use for the selection list. This ensures that Incubator staff must work for an Incubator company and so on. The department administration tool will expire all of the members of that department (or other organisation) is terminated. Our current list of entry types is in Figure 4.

Other flags will determine further processing. The directory flag can put someone into the directory automatically with no choice to the departmental administrator. For example, our ROTC staff go in – no choice. Other categories such as visiting researchers or Incubator staff can go in, but that can be controlled by the departmental administrators (''O'' optional, default include, ''P'' – optional, but default to not include) and the rest don't go into the directory at all. If there is an expiration delta, that number of days is added to the current date to get a default expiration date.

Entry Points, Delegation, Short Term Visitors

Not everyone on campus is there long enough to get their own picture ID card. While they still need one for building access and dining plans, the overhead of producing ID cards for someone who was going to be on campus for three days for a conference was simply not worth it. There are a few cases where an ID card is desirable as a keep-sake, for the most part, we have a collection of generic cards (guest1-Guest999).

While that worked for the physical access, we still had the problem of visitors to campus who needed to authenticate to access our VPN to get on campus, or to access the wireless network. This would result in the request and creation of a full computer guest account, with email access, printing allocations, disk charges, and would take a business day or two to get set up, or someone in the department would ''loan'' their account out. While the first option was arduous,

Entry_Type	Varchar2(24)	A short name for this type of entry.
Sponsor_Required	Varchar2(1)	A flag indicating of a sponsor is required for this type of entry.
Requestor_Required	Varchar2(1)	A flag indicating of a requestor is required for this type of entry.
SSN_Required	varchar2(1)	A flag indicating if an ID number is required.
Affiliation_Id	Number	An identifier of the department sponsoring this entry.
Expiration_Delta	Number	Number of days from now for entries to expire.
Dir_Flag	varchar2(1)	Flag to help control inclusion in the campus directory.
Display_Rank	number(3)	A rank order to be used for display purposes.

Figure 3: ID_Entry_Type description.

Entry Type	Exp Delta	Sponsor Required	Requestor Required	Dir Flag	Display Rank	Entry Type Id
Special Programs	60	N	Y		100	91121096
Temporary Employee	90	N	Y	P	100	91318569
Conference Guest	60	N	Y	N	100	91114523
Off Campus Contractor	120	N	Y		100	91393994
On Campus Vendor	120	N	Y	P	100	91398702
Department Vendor	90	N	Y	P	100	91402506
RU Club Member	180	N	Y		100	91405098
Incubator	180	N	Y	P	100	91068656
Tech Park	180	N	Y	P	101	91080046
Dependent		Y	N	N	160	91449911
Spouse		Y	N	N	160	91449912
Domestic Partner		Y	N	N	160	91449913
Personal Aid		Y	N		200	91449917
Visiting Researcher	120	N	Y	O	300	91399849
Visiting Faculty	120	N	Y	O	300	91399850
ROTC Faculty	90	N	Y	Y	400	91397007
ROTC Staff	90	N	Y	Y	400	91397008

Figure 4: Current ID Entry types.

the second could result in the staff member who ''loaned'' the account being terminated.

Instead, we developed a tool that allows departmental administrators (and each department has at least one, see the next section) to allocate a temporary account that will allow someone to access the VPN to get in, or the wireless network to get out. When they allocate an account, they can specify an expiration time from 3 hours up to 2 days (accounts can be renewed), after which time, the password will be reset automatically. They can also specify a comment for that assignment that they can later review. Once allocated, these accounts ''belong'' to that department. Once they have expired, they are available for re-use by that department. If the department has no ''free'' accounts, a new one will be drawn from a pool. The pool is monitored to ensure a ready supply of new temporary accounts. All of this happens without any involvement by the IT staff – it is entirely in the hands of the department administrators.

This gives departmental administrators the ability to satisfy some of the needs of visitors to their department quickly and easily.

Departments

We really can't talk too much about managing people, without understanding what department they belong to. One of the key points of this system, is that there is someone responsible for maintaining the status information for everyone on campus. This may be indirect like with HR maintaining employee information, or the Registrar maintaining student information, or directly with departmental administrators maintaining guest information. Although there are a few categories such as dependents or spouses that are ''owned'' by another person, all of the rest of the guests need to be ''owned'' by a department. This requires a good list of departments, and possibly the relationship between departments (all of the engineering departments belong to the school of engineering, etc.).

As part of the telephone directory project, we had to get a pretty good handle on departments. When the University went to a fund accounting system with composite account numbers, one of the elements was the ORGN (Organisation). These were arranged in a hierarchy, with every orgn rolling up to it's parent and so on to the president. Our joy was short lived however. The primary purpose of the ORGN tree, was financial accounting. This resulted in lots of extra ''departments'' (like copy center), departments that were not ''data enterable'', so these were paired with an ''office of XYZ'' department so charges could be rolled up. To make matters worse, the department ''names'' were limited to 30 characters, and there was no place, nor interest from the finance office in maintaining abbreviations.

To get around this, we created our own superset of the ORGN tree, where we could add our own ''virtual'' organisations, specify alternate names for real organisations as well as abbreviations. An agreement was made that we would only create ORGN codes (the primary key) that started with ''V'', ''S'' or ''DH'', and the rest of the name space belonged to them. New ORGNs from the controller's office are automatically added to the directory (and the telecom staff double checks them – some are not included). But this gave us a place to expand the existing university departmental hierarchy, as well as build new trees for vendors, and many other special groups.

Another aspect of the department management, was to identify one or more administrators for each department. If a department did not have one, it could inherit administrators from it's parent. Although this was originally intended for managing the telephone directory, it gave is a ready made list of people who can take on additional responsibility for guests in their department. This was a key component in ensuring that status information for EVERY person on campus was the responsibility for an identified person (or department) to maintain.

People Status

Now that we have identified all of our different data feeds, it is now time to put everything together into a uniform object (Figure 5). The first step is to create an Oracle view, Person_Status (Figure 6) that combines information from Employees, Students, ID_Guests, Directory_Aux_Entries and Hartford_Raw_Dir

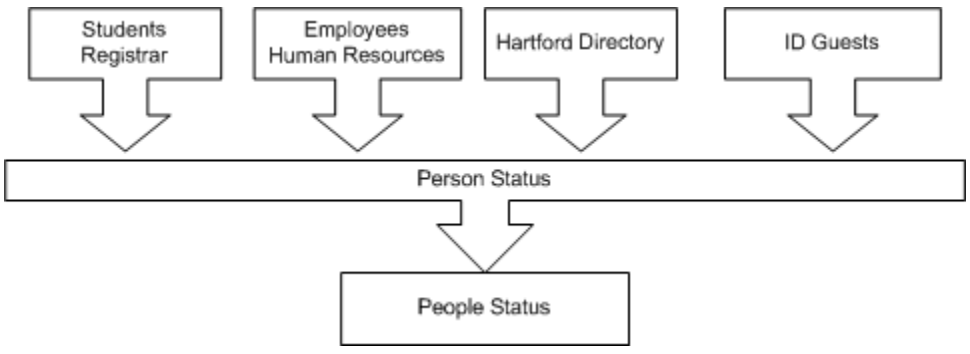

Figure 5: People Status data flow.

tables. We don't want to be querying this view all of the time, and we do want to know when someone's status changes, and maintain some history. To this end, we have a People_Status table (Figure 8). And finally, we feed this information to other systems.

Person_Status

The person status support was originally developed to provide a patron feed for our Library circulation system. It was later enhanced to feed our ID card system. It is now being generalised as people status management tool, however, some traces of the original library and ID card system remain in the column definitions. Most of this information is being moved to other tables that join with Person_Status based on the Status_ID column. The Person_Status view (Figure 6) is the key place where all the different inputs to the status process are brought together.

The Person_Status view is coupled tightly to People_Status_Types (Figure 7) to generate a snapshot of everyone's "status". The view and this table is where all of the site specific stuff lives. The view goes and roots around in a number of different database tables to come up with a standard set of status entries. The People_Status_Types table has columns for joining with the view to determine types, and some other columns that are used to feed the Library and some other systems. I have omitted this second set of columns, as they are gradually being phased out in favour of other tables. A subset of the entries in the People_Status_Types is available in the first appendix of the paper.

The Person_Status view is a series of joins with other tables and the People_Status_Types table, that are connected together with a SQL UNION directive. For ease of formatting, I have broken the view up into individual stanzas (Displays). This is where all the magic happens.

Get Employees

This first section looks in the employees table to get our current employees. The important distinction here is really the key 1 looking for "A" (active) employees and

Lib_Patron_Type	varchar2(3)	Patron Type for Library Circulation System
Status_Id	Number	Identifier for status type.
ID_Card_Status	varchar2(32)	"Status" entry for use on the ID card.
Person_Id	Number	Identifier for the person record for this entry.
Orgn_Name	Varchar2(32)	Name of the department, if applicable.
Orgn_Code	Varchar2(6)	Department Identification code. Used with Coas_Code to identify department.
Coas_Code	Varchar2(1)	Department Chart of Accounts identifier.
Type_Key	Number	Affiliation identifier for non departmental entries.
In_Dir	varchar2(1)	Flag indicating if person should be in the fac/staff directory.
End_Date	Date	Date when status expires, if known.

Figure 6: Person_Status description.

Status_Id	Number	Identifier for status type.
Rank	Number	Orders status values, highest value is returned.
ID_Card_Status	varchar2(32)	"Status" entry for use on the ID card.
Person_Id	Number	Identifier for the person record for this entry.
Status_Category	varchar2(8)	Rough category for status types.
Source_Table_Name	varchar2(32)	The name of the table we will match with.
Key_1	varchar2(16)	A key to match against, usage depends on category and table name.
Key_2	varchar2(8)	A second key (there is also a key_3 and key_4).
Nkey_1	number	A numeric key, usage depends on the status category and source table.
Comments	Varchar2(255)	A description of this entry.

Figure 7: People_Status_Types description.

```
select lib_patron_Type, status_id, id_card_status, person_id,
       substr(Ftvorgn_Title,1,32), pebempl_orgn_home,'9',
       to_number(null) TYPE_KEY, pst.in_dir, e.nbrjobs_end_date
  from employees E, people_status_types pst, fimsmgr.ftvorgn f
 where Status_Category='Emp' and key_1 = NBRJOBS_STATUS
   and PEBEMPL_ECLS_Code like nvl(key_2,PEBEMPL_ECLS_Code)
   and pebempl_orgn_home = f.ftvorgn_orgn_code
Union...
```

Display 1: Retrieving employees.

key 2 looking for the employee classification. It also dives into the Ftvorgn table to come up with the department name.[7] See Display 1.

Get New Employees

We dive back into the employees table again to look for new employees. The employee classification doesn't appear in the employees table until the next payroll cycle, and we really don't want to wait that long to start issuing ID cards and the like. So Human Resources has a tool to set the HR_Ok and HR_Status fields appropriately, and this stanza will pick those up. In the course of normal employee record processing, these fields will get cleared when the classification information comes through so we don't have people with two employee status values. But even if something goes wrong, and that isn't cleared, since the regular employee status types have a higher rank, that entry will override this entry. See Display 2.

Get Troy Students

Continuing with the view, we go after our local students. We are getting the same number of columns as we did in the first stanza (a requirement of a view), but most of the department related fields we are returning as null. We look for active students Sgbstdtn_Status_Code='AS', and those in the Troy campus (Key 1), and also breaking them down by College (school) and level (Grad/Undergrad). Although this level of granularity may be overkill for most applications, the Library wanted this fine grain distinction. See Display 3.

Get Hartford Students

We have a second campus in Hartford, and they code students somewhat differently. In Troy, the Registrar aggressively recodes students who are not active. But our Hartford campus does a lot of continuing education, where students just take a course or two, so their status is left as 'AS'. So to handle this case, we pull the same columns, but look for some other keys in the students table, Registered_This_Term. We actually have a third campus which is being phased out, so I will skip that stanza of the view. See Display 4.

Mailing List Entries

Our method of handling people who are already in the system, and need some special status, like what we did for the Emeriti faculty, is handled internally by adding them to a special "department". These tools were originally developed for the phone directory, and worked well here as well. In this case, an entry is made in the Directory_Aux_Entries table (which is listed as Dir_Aux_Ent in the appendix), and we pick up the appropriate entries.

[7] People well versed in Oracle and/or Banner will note that column aliases are missing, and additional conditions are needed to make this work. These have been removed to aid formatting. See the end of the paper for how to view the actual, working code.

```
select lib_patron_Type, status_id, id_card_status, person_id,
       nvl(substr(Ftvorgn_Title,1,32),'Unknown'), pebempl_orgn_home, '9',
       to_number(null), pst.in_dir, to_date(null)
  from employees E, people_status_types pst, ftvorgn_coas_9 f
 where Status_Category='NewEmp'
   and hr_ok = 'Y' and hr_status = key_1
   and nvl(pebempl_orgn_home,'XXXXXXX') = f.ftvorgn_orgn_code (+)
Union...
```

Display 2: Select new employees.

```
select lib_patron_Type, status_id, id_card_status, person_id,
       Sgbstdn_Majr_Code_1, Null, Null, to_number(null), Null, To_Date(Null)
  from banner_students bs, people_status_types pst
 where Status_Category='BStu'
   and Key_1 = SgbStdn_Campus_Code
   and ( key_2 = Sgbstdn_Coll_Code_1 or key_2 is null )
   and Key_3 = Sgbstdn_Level_Code
   and Sgbstdn_Status_Code = 'AS'
Union...
```

Display 3: Select Troy students.

```
select lib_patron_Type, status_id, id_card_status, person_id,
       Sgbstdn_Majr_Code_1, Null, Null, to_number(null), Null, To_Date(Null)
  from banner_students bs, people_status_types pst
 where Status_Category='HStu'
   and Key_1 = Sgbstdn_Level_Code
   and ( Hart_Reg_This_Term = 'Y' or Registered_This_Term = 'Y' )
   and Sgbstdn_Campus_Code = 'H'
   and Key_2 = Sgbstdn_Status_Code
Union...
```

Display 4: Select Hartford students.

We use a different table for the department name this time, this stanza was written later and I should revise the employee entries. See Display 5.

ID Guests

This is the place where we get all of our id guests. A number of the id entry types map directly to a status type. You can actually compare the Nkey_1 values in the appendix with the Entry_Type_Id values in the Entry Types listing above. We also make a check to ensure that the guest entry either does not have an expiration date, or that date is some time in the future. If there isn't one on the record, the current time will be used, but that is good enough! See Display 6.

People Living on Campus

We have a significant number of people who are living on campus after they graduate. One of the side effects of graduating, is losing your student status, which triggers a number of events, including cancellation of your ID card, which locks you out of your building! Although these people were no longer registered students, they did have housing contracts. We added some information about room assignments to our student table, and added this final stanza to the Person_Status view. During the normal school year, this entry will provide a second status value to all on campus students, but since it is of a lower rank, it is ignored. See Display 7.

People_Status

While the Person_Status view will give us a snapshot of the current status of everyone, it is not able to provide any historical information. A common question when checking someone with no status, is what was their status before it ended. There were also concerns about performance of this view – it is looking into a number of other database tables. For both of those reasons, we have a process that runs daily that checks the current status of everyone, with what we have saved in the People_Status table (Figure 8).[8] We can also invoke this processing for a specific person. This function is used for several administrative applications when they want to ensure someone's status record is current as of this instant. Refreshing a single person's status is almost instant, it takes longer to redraw the screen.

The People_Status table has the same columns as the Person_Status view, with the addition of a start and end date, as well as some numeric columns (When_Inserted and When_Marked_For_Delete) to assist with data propagation. Changes in a person's status are also reported in the Meta Change Queue subsystem, so other systems can easily watch for changes in a person's status.

[8] On a recent run, this process compared 19,000 records in just over a minute, a performance level I am comfortable with.

```
Select lib_patron_type, status_id, id_card_status, person_id,
       substr(nvl(orgn_common_name,orgn_name),1,32), dae.orgn_code,
       dae.coas_code, title_id, dd.def_include, to_date(null)
  from Directory_Aux_Entries DAE, people_status_types pst,
       directory_departments dd
 where Status_Category = 'DirAux'
   and pst.Key_1 = DAE.Orgn_Code and pst.Key_2 = DAE.Coas_Code
   and dae.orgn_code = dd.orgn_code and dae.coas_code = dd.coas_code
 Union..
```

Display 5: Select mailing list entries.

```
select lib_patron_Type, status_id, id_card_status, person_id,
       ia.name, ia.orgn_code, ia.coas_code,
       nvl(ip.affiliation_id,ip.sponsor), ip.dir_flag,
       ip.expiration_date
  from id_people ip, people_status_types pst, simon.id_affiliates ia
 where Source_Table_Name = 'Id_People'
   and Nkey_1 = Ip.ENTRY_TYPE_ID
   and nvl(ip.expiration_date,sysdate) >= sysdate
   and Key_1 = 'CURRENT'
   and ip.affiliation_id = ia.affiliation_id (+)
Union...
```

Display 6: Select ID guests.

```
select lib_patron_Type,status_id,id_card_status,person_id,
       S1rrasg_Bldg_Code || '-' || S1rrasg_Room_Number,
       Null, Null, to_number(null), Null, S1rrasg_End_Date
  from banner_students bs, people_status_types pst
 where Status_Category='ResLife'
   and S1rrasg_Active = 'Y'
   and ( key_1 is null or key_1 = s1rrasg_bldg_code )
```

Display 7: Select people living on campus.

Making Use of People Status

The original target of the people status support was to feed the circulation system for our Library. It was later expanded to feed the ID card system. It became obvious that extending the People_Status_Types table for every new application was the wrong approach. Instead, we added a set of People_Status_Aux... tables that allowed us to define new flags for each status type, and developed a tool to allow administrators of other systems to set their flags and not interfere with other consumers of the status information. We also use a person's status to trigger other processing.

People Status Auxiliary

Adding columns to tables, or creating new tables, and then writing interface routines and tools to manage them can get pretty tedious, and takes a lot of time that could be better spent on other tasks. I wanted to be able to "extend" status related function by making table entries in the database, rather than writing new code.

To start, we defined the table People_Status_Aux_Master (Figure 9) that define new "streams" of status information. Initially I was envisioning selecting people for data feeds (or streams) to other systems, and the name stuck. This defines the name of the stream and lists an Oracle role that will be used for access control by the administrative tool.

The next step was to define a prototype table, People_Status_Aux_Proto (Figure 10), to identify the possible fields, the field types and the default values. This is used by the administrative tool to automatically generate the appropriate columns and switches on the web page.

The final table holds the actual values for each stream, People_Status_Aux_Values (Figure 11) which stores a field value for each status, stream, field triple.

Once we have defined a new stream, assigned it an access role, and defined one or more fields for that stream, we are able to delegate the management of these values to the appropriate interested parties. At present we have the following streams defined and in use; see Figure 12.

Now that we have all of this information in the system, and control delegated to the appropriate offices, we need to get it out again. At the most basic level, we have a PL/SQL package with some routines, one that will give you a list of everyone with a particular Stream/Field/Value triple and another that will return true or false, if a given person has a particular

Person_Id	Number	Person identifier.
Status_Id	Number	Identifier for status type.
Start_Date	Date	Date when this status was initiated.
End_Date	Date	Date when this status will terminate if known or was terminated.
Orgn_Name	Varchar2(32)	Name of the department, if applicable.
Orgn_Code	Varchar2(6)	Department Identification code. Used with Coas_Code to identify department.
Coas_Code	Varchar2(1)	Department Chart of Accounts identifier.
Type_Key	Number	Affiliation identifier for non departmental entries.
In_Dir	varchar2(1)	Flag indicating if person should be in the fac/staff directory.
When_Inserted	Number	Sequence value when record was inserted.
When_Marked_For_Delete	Number	Sequence value when record was considered to be deleted.

Figure 8: People_Status description.

Stream_Name	varchar2(32)	The name of the stream.
Access_Role	varchar2(32)	An Oracle role that can manage this stream.

Figure 9: People_Status_Aux_Master description.

Stream_Name	varchar2(32)	The name of the stream. Must exist in the Master table.
Field_Name	varchar2(32)	The name of the field.
Field_Type	varchar2(32)	The data type – to assist the web tool in formatting and controlling.
Field_Length	varchar2(32)	The maximum length of the field where applicable.
Field_Default	varchar2(32)	The default value to use.
Field_Rank	Number	Rank order to display fields on the web tool.

Figure 10: People_Status_Aux_Proto description.

Status_Id	Number	The People_Status_Types.Status_Id of the status getting this value.
Stream_Name	varchar2(32)	The name of the stream.
Field_Name	varchar2(32)	The name of the field.
Field_Value	varchar2(32)	The value for this particular field.

Figure 11: People_Status_Aux_Values description.

Stream/Field/Value value. We have also set up Generate_File [6, 8] targets to extract list based on stream and field combinations. We have also written several ''wrapper'' packages to provide the appropriate streams to people and applications who need to connect directly to the database.

Other Uses

Our physical access control system (Card readers, parking gates), also uses status, or more specifically the lack of any status to automatically terminate all access to campus building, roadways and parking. We also use status to control if and how people are included in the different campus directories.

Conclusions

The people status project has been evolving over a number of years here. During that time, some of the original data sources have been replaced, and new ones have been added. In some cases the direction of data flow has changed. At one time the faculty/staff directory data drove the status information; that has been reversed, status data now drives the faculty staff directory.

The People Status project at Rensselaer has reached critical mass. It impacts enough systems that matter to people (such as Parking!) that folks are willing to play by our rules. And it does make life simpler – students enter the system via the Registrar, employees get in via Human Resources and everyone else comes in via the ID card office. Once in, the assorted special cases get picked up by the departments who care about those people and things work well and with a minimum of human intervention.

Implementation Lessons

The people status project was not rolled out as a complete package starting at nothing, but rather was built on several smaller projects such as the directory and computer account management. These systems allowed us to identify and refine the data sources and procedures, while assembling the infrastructure. It also provides time and opportunity to develop the working relationships with key players in other departments.

There are two types of people I needed to work with to pull this project together, people who had data I could use, and people who needed data. By starting with some smaller projects (they didn't seem small at the time!) like the phone directory and computer account management, I was able to put together a comprehensive enough ''people'' database, that I could get people who needed feeds interested. By having tools and techniques readily available to enable them to control their feeds (Generate File, Meta Change Queue, etc.), I was able to get a number of ''clients'' for ''my data''. These clients in turn made it more attractive for the owners of the data to work with me, since it would leverage their work.

I was also able to target groups that were outside of the mainstream, and so didn't have quite the level of IT support that they may have wanted. I help them, and they are willing to modify their procedures and processes to better accommodate what I need. I also find it very helpful to visit my ''clients'', and see what they are doing.[9] I have been able to provide them with a tool that greatly simplifies their operation (often times, it was an existing tool that needed slight modification) and I strengthen a person to person relationship as a result.

One important lesson, is that your source data is often not quite what you need – It may be too early or too late (like with HR) requiring some extra tools to make it useable, or intended for a different purpose and it will need to be adapted and cleaned up (like with the departmental tree). You have to work with your sources to make it work. The solution may be with people and process and not technology.

Systems and Processes Impacted By This Project

A number of systems and business processes (see Figure 13) has been impacted and improved by this project. The most common improvement is the automatic removal (or at least notification) when people leave or their status changes.

[9] Is this an obvious part of customer service – Yes! Is it often overlooked, sadly, Yes.

EBS List	For important campus announcements, we have email lists like ''All Faculty'', ''All Staff'', ''All Students'' that are maintained based on current status. Our Postmaster keeps these mappings in place.
Hostmaster	For our host database, there are requirements as to who can ''own'' or ''administrate'' machines on our campus network. Our Hostmaster makes this determination.
ID Access Control	We have several buildings that are open to the campus ''community'' via card readers. This enables our Access control staff to control that definition of community.
ID Card Admin	There are two controls here, one to indicate if that person's information is maintained in Banner (if not, the ID desk can make the updates directly), and also if that person is eligible to have a dependent.
Package Tracking	Our campus mail room wants an address feed, but not of everything. They pick and choose.
PC Store	Our Campus Computer store is limited in who they can sell to (due to contracts with vendors). This helps them identify who is ok, and who is eligible to charge their purchases to a student account.

Figure 12: Current auxiliary streams in use.

A Note on Table Definitions

There are a number of Oracle table definitions included in this paper. In order to save space and assist with formatting, I did not include a number of columns that are in most of the Simon tables. A recurring set of columns that we see in many of the Simon tables are When_Inserted, When_Updated and When_Marked_For_Delete columns. These are filled with an ever growing sequence, and are used to identify records that have changed since some previous point in time, and to propagate those changes to other tables and systems [4]. Many tables also have a Clerk or Clerk_Id field to indicate who touched that record last, and some also have an Activity_Date column to indicate in a more human form, of when that change took place. I have also removed references to those columns in some of the source code examples included here. The full definitions of both table and source code can be obtained via the Web. See the following section for details.

Futures

We continue to identify new groups and categories of people that need status. Adding these new types has become pretty trivial, with the tools and techniques being quickly adaptable to new situations. One offshoot of this project deals with physical access control. We mapped the people status auxiliary values into a more general demographic mapping module (along with departmental affiliation, course registration, etc.) into consumers of group information such as the access control system, and windows protection groups. Now when a student registers for a particular class, not only do they get access (via their ID card), to the room with the lab equipment, but access (via their computer account) to restricted course files and directories on Windows and AFS file servers. There might even be another paper on this topic for next year.

References and Availability

This is not a comprehensive, stand alone, product that we can package up easily for distribution. Despite efforts over the years to move our business rules out of the code, and into data table and views, there is a lot of site specific stuff in here. That being said, I feel that there is a lot here that can be used by other sites. You will have to do some code development of your own to address your own site's specific issues and requirements. Some of the existing packages we have developed may serve as models for your own work. The web tools we have developed use a pretty much standard Oracle web environment with our own custom front end to handle the authentication.

All of the PL/SQL source code for the Simon system as well as the full table and view descriptions are available via the web at http://www.rpi.edu/campus/rpi/simon/misc/Tables/simon.Index.html and http://www.rpi.edu/campus/rpi/simon/misc/Tables/SIS-index.html . If you ask nicely, I will try to answer questions and might be able to dig out some of the C and JAVA code that makes up other parts of the system.

Acknowledgements

I would like to thank Tom Perrin for his shepherding of this paper with me. I also want to thank Rob Kolstad for his excellent (as usual) job of typesetting this paper.

Author Biography

Jon Finke graduated from Rensselaer in 1983 with a BS-ECSE. After stints doing communications programming for PCs and later general networking development on the mainframe, he then inherited the Simon project, which has been his primary focus for the past 14 years. He is currently a Senior Systems Programmer in the Communication and Collaboration Technology department at Rensselaer, where he continues integrating Simon with the rest of the Institute information systems. In addition to the Simon project, Jon is also involved with the support of the Telecommunications billing system,[10] and providing data and interfaces for Unity Voice Messaging and CISCO VOIP deployment projects at Rensselaer. When not playing with computers, you can often find him merging a pair of adjacent row houses into one, or inventing new methods of double entry accounting as treasurer for Habitat for Humanity of Rensselaer County. Reach him via USMail at RPI; VCC 319; 110 8th St; Troy, NY 12180-3590. Reach him electronically at finkej@rpi.edu. Find out more via http://www.rpi.edu/~finkej .

[10] AXIS – Pinnacle CMS by Paetec.

Library	Circulation and Patron database – drives borrowing limits, aids with contacting via address feeds.
Access Control	Automatic revocation of access when a person leaves, automatic access for community members.
Computer Accounts	Automatic creation and expiration.
Human Resources	HR controls when a person will be issued access, email, etc. – ensuring compliance with employment rules.
ID Card Office	Single point of contact for guests, identification of departmental contacts, refinement and documentation of policies and procedures.
Directory	Status drives inclusion in the online and printed directories.

Figure 13: Systems and business processes impacted by this system.

Bibliography

[1] Anderson, Eric and Dave Patterson, ''A retrospective on twelve years of LISA proceedings,'' *13th Administration Conference (LISA 1999)*, pp. 95-107, USENIX, November 1999.

[2] Arnold, Bob, ''Accountworks: User create account on SQL, Notes, NT and UNIX,'' *The Twelfth Systems Administration Conference (LISA 98) Proceedings*, pp. 49-61, USENIX, December, 1998.

[3] Cooper, Michael A., ''Spm: System for password management,'' *The 9th Systems Administration Conference (LISA IX) Proceedings*, pp. 149-170, USENIX, September, 1995.

[4] Finke, Jon, ''Data propagation between oracle tables,'' *Proceedings of Community Workshop '92*, Troy, NY, June, 1992.

[5] Finke, Jon, ''Institute white pages as a system administration problem,'' *The Tenth Systems Administration Conference (LISA 96) Proceedings*, pp. 233-240, USENIX, October, 1996.

[6] Finke, Jon, ''An improved approach to generating configuration files from a database,'' *The Fourteenth Systems Administration Conference (LISA 2000)*, pages 29-38, USENIX, December, 2000.

[7] Finke, Jon, ''Embracing and extending Windows 2000,'' *The Sixteenth Systems Administration Conference (LISA 2002)*, USENIX, November, 2002.

[8] Finke, Jon, ''Generating configuration files: The director's cut,'' *The Seventeenth Systems Administration Conference (LISA 2003)*, pp. 195-204, USENIX, October, 2003.

[9] Finke, Jon, ''Meta change queue: Tracking changes to people, places and things,'' *The Eighteenth Large Installation Systems Administration Conference (LISA 2004)*, pp. 231-239, USENIX, November, 2004.

[10] Harlander, Dr. Magnus, ''Central system administration in a heterogeneous unix environment: Genuadmin,'' *USENIX Systems Administration (LISA VIII) Conference Proceedings*, pp. 1-8, USENIX, September, 1994.

[11] Hughes, Doug, ''User-centric account management with heterogeneous password changing,'' *The Fourteenth Systems Administration Conference (LISA 2000)*, pp. 67-76, USENIX, December, 2000.

[12] Rosenstein, Mark A., Daniel E. Geer, Jr., and Peter J. Levine, ''The Athena service management system,'' *USENIX Conference Proceedings*, pages 203-211, USENIX, Winter, 1988.

Appendix A: Selected People_Status_Types

RNK	ID Card Status	Status Category	Source Table	Key 1	Key 2	Nkey 1	Description
750	Employee	Emp	Employees	A	E%		Exempt Employees
750	Employee	Emp	Employees	A	X%		Executives
750	Employee	Emp	Employees	A	N%		Non Exempt employees
750	Faculty	Emp	Employees	A	F%		Faculty
740	Employee	NewEmp	Employees	Staff			Newly hired staff
740	Hartford Emp	HartEmp	HartfordRawDir				Hartford Employees
740	Faculty	NewEmp	Employees	Faculty			Newly hired faculty
730	ROTC Staff	N/A	id_people	CURRENT		91397008	US Military personal assigned to the ROTC detachments in a support (non teaching) assignment.
730	ROTC Faculty	N/A	id_people	CURRENT		91397007	US Military personal assigned to a ROTC detachment who will be teaching ROTC and other courses.
730	On Leave Employee	Emp	Employees	F			On Leave with full benefits
730	Employee (PT)	Emp	Employees	P			On Leave w/ Partial Pay and Benefits
650	Visiting Researcher	N/A	id_people	CURRENT		91399849	A person doing research (but not being paid by RPI)
650	Research Professor	N/A	id_people	CURRENT		91399850	Someone who is teaching, but is not paid by RPI. Will have a memo from the Provost's office or the Dean's office.
640	Undergrad	BStu	Banner_students	T	A		Undergrad, in Architecture at Troy
640	Graduate	BStu	banner_students	T	S		Graduate, in Science at Troy
640	Graduate	BStu	banner_students	T	E		Graduate, in Engineering at Troy
640	PDE Grad	DStu	banner_students	GR			Graduate Student with PDE
640	HartGrad	HStu	banner_students	GR	AS		Graduate Student at Hartford
640	Undergrad	BStu	banner_students	T	E		Undergrad, in Engineering at Troy
640	Undergrad	BStu	banner_students	T	S		Undergrad, in Science at Troy
640	Graduate	BStu	banner_students	T	A		Graduate, in Architecture at Troy
560	Incubator	Incubatr	id_people	CURRENT		91068656	A person affiliated with a company in the incubator center.
550	Dependent/Spouse	Dependnt	id_people	CURRENT		91067788	Dependent/Spouse of existing RPI person
550	Personal Services	N/A	id_people	CURRENT		91449917	A personal aid, generally for health care of a disabled member of the Rensselaer Community.
550	Family	N/A	id_people	CURRENT		91449912	A spouse of a member of the RPI community
550	Family	N/A	id_people	CURRENT		91449913	A domestic partner of a member of the RPI Community.
550	Family	N/A	id_people	CURRENT		91449911	A dependent, spouse or domestic partner
550	Temp Employee	TempEmp	id_people	CURRENT		91318569	A temporary employee, not on payroll – JJ Young, Manpower, etc.
500	Emeritus Faculty	DirAux	Dir_Aux_Ent	MR6190	9		School of Engineering Emeritus Faculty
500	Emeritus Faculty	DirAux	Dir_Aux_Ent	MR6210	9		Emeritus Faculty from the School of Science
500	Emeritus Faculty	DirAux	Dir_Aux_Ent	MR6200	9		Emeritus Faculty in Architecture
300	Vendor	Vendor	id_people	CURRENT		91067997	A vendor
300	Retiree	Retiree	id_people	CURRENT		91080042	A retiree
300	Vendor	N/A	id_people	CURRENT		91398702	A person affiliated with an on campus vendor
300	Vendor	N/A	id_people	CURRENT		91402506	A vendor working with a specific department on campus.
300	Vendor	N/A	id_people	CURRENT		91393994	Somone affiliated with an off campus vendor
300	Tech Park	TechPark	id_people	CURRENT		91080046	An employee at a tech park company
260	RU Club Member	N/A	id_people	CURRENT		91405098	A member of a Rensselaer Union sponsored club or organization.
250	Special Programs	Spec Prg	id_people	CURRENT		91121096	Special Program
200	Special Access	Spec Acc	id_people	CURRENT		91068269	Special Access Card
200	Conference Card	ConfCard	id_people	CURRENT		91114523	A Generic Conference ID card.
150	Residence Hall Occupant	ResLife	banner_students				A person with an active housing contract. May not be a student nor employee.
55	Retired Faculty	DirAux	Dir_Aux_Ent	S6160	9		A retired faculty member. This status is maintained by the Provost's office.