Typical ingest processes

back up to How to plan data ingest for VIVO

previous topic: Ingest tools: home brew or off the shelf?

Note: these discussions reflect primarily the approaches and workflow that have been used at Cornell. Other approaches are used at other sites, and please update or annotate as appropriate to point out different requirements and/or solutions.

Introduction

Data ingest is a very broad term that refers to first-time loading of data but must also encompass processes to correct errors and reflect additions and deletions. Data are rarely static, and a general model for data ingest needs to include the context of where data are managed and where the resources to maintain data can be found. Data quality can be addressed at five different points in the workflow -- before it leaves the source system (whatever that may be), as a data file before it's brought into VIVO, during data ingest processes, after it's been loaded into VIVO, and finally as a reporting phase back to the source.

Alternative approaches

Some VIVO sites do not allow manual editing by users, but reflect data from one or more other systems of record with VIVO being a point of integration and for syndicating integrated data to other websites or reporting tools. This can simplify data management after it's in VIVO but still very likely requires data alignment unless all the sources of data are internally consistent and share common unique identifiers.

When data in VIVO have been created or augmented by interactive editing, and when users can edit their own pages (typically called self-editing), there are more complexities to plan for.

- If data in VIVO come from ingested sources but are also edited directly, whether by end users editing their own profiles or by a limited number of data curators or student hires, then there are more complexities to deal with.
    - First, if data are corrected in VIVO (a misspelled name, for example) but not in the source, the next input from that source will likely overwrite the correction and revert to the misspelled name.
        - If possible, build workflows to feed changes back to the source for correction, and propagate the next fix to VIVO through the next scheduled ingest of that source.
        - Sometimes a manual edit will still have to be made in VIVO to assuage an unhappy user, but if the identical change is made in the original source there should not be a concern about overwriting the correct version.

Ideally ingest processes are made repeatable and incremental so that changes do not require removing and then adding large amounts of data, but sometimes a source is only updated annually or the source system goes through changes that require large batch changes.

- In this case a separate VIVO instance can be used for ingest.
    - This instance is populated from the nightly backup of the production instance.
    - The use of a separate VIVO means that the production instance is not loaded down by the ingest process.
    - Ingest processes run at night.
        - Since ingested data is largely separate from editable data, it is not likely that there would be conflicts, except for the load on the system.
        - Ingest processes are run that compare the new data to the data in VIVO.
        - They generate RDF triples that must be added or removed from VIVO to represent the new data.
        - Because we are not apply these triples immediately, we can inspect them for correctness before committing them.
        - The RDF triples are applied to the production VIVO system.
            - These processes are ad hoc, and idiosyncratic to Cornell's data sources and ontology extensions. They are constantly being changed, and are not packaged for release.

Designing a repeatable ingest process that tests new data against what is already in VIVO

Concepts

The process of developing a data ingest plan for VIVO often focuses on each different data source independently, but in fact there may be some overlap among sources, whether those sources represent different types of data or different sources of the same type of data.

For example, people will probably come first from a human resource database – employees, departments, and the positions that connect employees and departments. But a grants ingest process will also bring in new people, as there may be investigators from other organizations listed. And when publications are ingested, a large institution may find there are tens of thousands of people records to keep straight.

In some future world that organizations like ORCID are working achieve, every researcher will have a unique international identifier, and this identifier will help disambiguate whether the John Doe that co-authored with a researcher at your institution is the same John H. Doe serving as an investigator on a grant. For now, the mechanisms of identifiers and the heuristics of disambiguation are important to recognize but not to solve – it's primarily important in planning your ingest processes to recognize that these questions are out there.

Addressing identity

We don't recommend using a person's name as part of their URI for the simple reason that their name may change. In fact, many data architects recommend always using completely randomized, meaningless identifiers within URIs (for the part after the last / in the URI, known as the local name).

When performing extract, transform, and load (ETL) tasks to get data into VIVO for the first time it will be necessary to create URIs for each new person, organization, and other type of data ingested. These URIs can be created by VIVO's ingest processes or generated by the ETL process itself and loaded into VIVO. The ETL process can create any arbitrary URI as long as the local name begins with a letter – some RDF processors are not happy with URIs having localnames beginning with a number or other symbol. The ETL process can also create a URI based on an institutional or other identifier, which has the advantage of being predictable and repeatable. However, you need to be sure that the identifier is unique and will not be re-used in the future should the person leave the institution or an organization identifier be recycled.

The goal with subsequent ingest – either new types of data or updates to existing sources -- is to match new incoming data against the existing URIs and their contents to avoid creating duplicates. This means having some way of checking new data against existing data.

Creating nightly accumulators

At Cornell, we have found it advantageous to run a nightly process that extracts a list of all people and all instances of several other types of entities along with their URIs and key identifying properties such as name parts, email addresses, and so on. These lists serve as source against which to match incoming data to avoid having to query our production VIVO instance every time we encounter a co-author's name, a journal, or an organization name. We call the lists accumulators, and store them in an XML format because our largest source of updates about researcher activities comes from an XML web service.

These accumulator lists help assure that new data are matched against existing data, reducing but not eliminating all possible false positives or false negatives. We will discuss disambiguation in more detail further along in the process in connection with How to manage data cleanup in VIVO.

---

next topic: Challenges for data ingest