Using Data Analysis for Better Decision Making

BY SANFORD HESS

Let's start with the bad news: There are no magic solutions here. Data connectivity is an aspirational goal, like operational efficiency. As defined in this article, "data connectivity" is the act of taking information from different computer systems and combining it to gain better insights. As you have probably seen already, that's harder than it sounds.

Why is data connectivity so difficult? It makes sense that computer systems should be able to talk to each other. And in fact, they can. There are endless ways for computer systems to share information, starting with the prosaic .csv file (.csv = comma-separated values) and going all the way through real-time direct access. The problem is the data.

You are probably familiar with one way in which systems talk: When one system imports or exports a file of information from or to a second system — an "interface," as we information technology (IT) people like to call them. Financial systems have lots of interfaces. They import files for posting to the general ledger and export formats like bank files or IRS layouts. (All the examples in this article will be for financial systems, but the same ideas would apply regardless of the data involved.)

The challenge of data connectivity is finding the commonality between systems. How do you make a financial system "speak" to a property database? Or police arrests? They usually have different transaction formats, reference codes, and even inconsistent code values to represent the same thing.

Data connectivity is slightly different from an interface; it's the idea of pooling data from two or more systems so you can ask questions about the combined information. Think about it as two different systems that are both exporting information to a third location, which is a reporting database, where people can run queries against it.

The challenge of data connectivity is finding the commonality between systems. How do you make a financial system "speak" to a property database? Or police arrests? They usually have different transaction formats, reference codes, and even inconsistent code values to represent the same thing. (There's an example later with organization codes.) The rest of this article explains how to approach this problem, although of course actual results will vary because every situation is different.

Before we begin, some terminology. Computer systems generally have two types of data: transactional data and reference codes. Transactional data are the detailed history of events, which are classified using the reference codes. Financial systems are full of transactional data: general ledger postings, invoices (paid or billed), budget requests, etc. Reference codes are the pre-defined values that group transactional data such as object codes, organization codes, vendor codes, funds, grants, etc.

STEP ONE — KNOW YOUR DATA

Connecting data starts with an understanding of what's there and how it's stored. Actually, let's back up — it *should* start with having a business question that's worth the effort. "Let's just mix all the data together and see what we find" is not a good approach. Before anyone expends significant effort connecting data, make sure that there's a clear statement of purpose and that there is a consensus from the people who understand the data that the information is available.

Notice the key phrase "people who understand the data" — that's the necessary ingredient for creating any meaningful data connectivity. This is especially true of financial systems, which have lots of transactions as well as data structures that are more complex than most. (Extracting the entire general ledger is useless if every query against it adds up to zero!) Understanding how to exclude accrual postings from a revenue query is harder than querying something concrete like the number of multi-family building permits.

For example, let's say someone wanted to combine business tax collections by address with other data about the location, such as crime incidents or 311 calls. You need to know how to find revenue transactions in the general ledger or a subsidiary file, depending on your system. To get the address you might need to combine the business address from a customer file ("reference data") with the tax collection amounts ("transaction data"). Finally, you'll need to know

if the dollar amounts are expected to be negative because they came from revenue postings.

It's not easy, which is why the first step of connecting the data is to know what you're dealing with.

> Connecting data starts with an understanding of what's there and how it's stored. Actually, let's back up — it *should* start with having a business question that's worth the effort.

STEP TWO — FIND THE COMMONALITIES

OK, let's assume that there's a worthwhile business question and the appropriate data have been extracted from each system. Now, you must find the common reference points that let you tie the sets of data together.

Let's go back to the example of business tax collections — we want to link collections to other geographic data about the business location. Street addresses are a widely used data points, but they also demonstrate a common problem: The same information can be stored in many ways. One system might have street numbers in a separate field, while another system might mix them in one field with the street name. Street directions are worse — even if they show up, they may be inconsistent — for example, "W Main" versus "W. Main" versus "West Main." (If the addresses are a total mess, then the best approach is to pass them all through a geo-coding process and let it try to match them all to geographic coordinates.)

Addresses are relatively easy, however, because at least there is a standard set of values that most people agree to. (Another example: purchasing commodity codes, although there are multiple standard sets.) That's not true for most data — so matching up values is more work.

Finance data experts should expect lots of these requests. Finance systems are chock-full of data, especially the money data that are part of many business questions. The finance department's chart of accounts are usually wonderfully precise and can slice data in many directions: objects, organizational structure, budget lines, and other elements. The problem is that other systems might not be structured the same way.

Here's a real-life example: The City of Urbana, Illinois, Public Works Department delivers services across many programs, including traffic signals, facilities, and forestry. Those appear as 16 organizational codes in the financial data. However, the work order system for Public Works was set up with a different organizational coding structure, with 12 divisions. Some of the codes match, but not always exactly, so matching work orders to the program budgets requires a crosswalk that includes more detail than the division codes — for example, the type of work performed. It's possible, but it isn't easy. The good news is that the operations manager came up with a good answer: Change the work order system to use the exact same organization codes as finance uses. (When two systems clash, the one that's bigger and harder to change often wins — good news for finance systems.)

You might not find any common reference points at all. If that's the case, those data probably aren't going to connect. Not everything does. This is why you have the conversations before you spend the effort building anything.

Sometimes the result of this effort is the realization that you need to collect additional data fields in one or both systems. That's an entirely acceptable result, so long as you

start the effort now to capture it. It will take time to get the data field added, let alone accumulate enough data to be meaningful.

Here's another real-life example from the Urbana pubic works department. Urbana has a landscape recycling center that recycles yard waste into mulch and other products that it sells. In the financial system, the recycling center is an enterprise fund and uses the citywide object codes to track expenditures. Sales are entered in a cash register, which produces a nightly extract file that is transformed and loaded into the finance system's billing module to post revenue, with different billing codes assigned for every product sold. This gives the finance department a clear picture of overall costs and detailed revenue by item. However, Public Works wants to know the profitability of each item, to determine if any should be discontinued (or re-priced). The problem is that the finance data only include the in aggregate, not per item. A great question was asked: Can the work order system data be combined with the finance revenue to calculate the cost of product?

It turned out that the answer was "No, we're not tracking work at that level of detail." This led to discussions about how that work could be tracked in the work order system, although there is no silver bullet solution here because the workers do many tasks all day and it's not a great environment for carrying around tablets for remote entry. (A soapbox comment: A common dilemma is demonstrated here, between capturing better data and letting people get their jobs done. When this happens, it's important to find the right balance, and that requires conversations with everyone involved. It's a sadly ironic situation if the goal of gathering data to analyze efficiency creates more inefficiency in daily work.)

STEP THREE — VERIFY THE RESULTS

If you've been through IT projects, you know that testing is an important part of any computer system — and data analysis is no different. It's especially important to verify the data if you expect to share them publicly, because once data are out there the information is very difficult to retract.

If you've been through IT projects, you know that testing is an important part of any computer system — and data analysis is no different. It's especially important to verify the data if you expect to share them publicly.

There are lots of ways for something to go wrong when creating extracts and transforming the data. "Transform" is the catch-all term IT uses for making changes to the data after they're extracted. This could range from reformatting a date to adding new calculated fields, but it's all programmed by humans — so mistakes are inevitable.

The best way to test extracted data is to run queries on the data in their new home and run the same query on the original data. You should get the same results. In addition to summary totals, you should also spot check a few records field by field to verify the data mappings. So, in the business tax revenue example, you would run monthly collection totals to verify the amounts extracted and choose a few records to verify that the address and reference codes are correct, compared to the originals.

Another time to verify the data is if someone finds surprising results. Don't forget to be suspicious of the data! Here's an example. Urbana's open data included police records pulled down from the source system as Excel files, then transformed

and loaded to the public portal. A data expert in the police department noticed a pattern of periodic drop-offs in incidents that no one in IT had spotted. The culprit? The Excel files were an older format that limited sheets to 65,536 rows — so exported records after those were being lost. (Excel is very useful, but it has some nasty habits that make it bad for data transformations. Leading zeroes can be lost — a problem when you have reference codes like "0102" — and Excel also likes to change parcel numbers to scientific notation. Not to mention Excel Dates — really, Excel, you need your own date format? Really?)

Even 10 years ago, mashing up data from different systems was too expensive and complex for most government organizations, but now anyone can find open-source tools they can use to extract, merge, and host datasets.

CONCLUSIONS

This article started with the bad news that there is no magical fix for connecting data, but we can end on a more positive note about the possibilities that lie ahead. Even 10 years ago, mashing up data from different systems was too expensive and complex for most government organizations, but now anyone can find open-source (i.e., free) tools they can use to extract, merge, and host datasets. We need these tools because demand for connected data is growing — from internal managers, elected officials, and our citizens. As people gain experience with data analysis, it's only natural that they will start asking broader data questions.

You might be one of those people receiving those questions. Even if you're not getting questions yourself, you might know a lot about the financial system, the chart of accounts, or other software packages run in your organization. This puts you in a position to help those requesters when the questions come.

Your goal is to help the requesters clarify their business problems based on the available data and to determine what data should be used and how to interpret the information. The steps above can give a structure to the approach, but every situation is different — and the best place to start is always getting the experts and requesters in the same room to have a discussion. (Whiteboards are optional, but useful.)

If you want to learn more about your data, learning the reporting tools is a great approach. Try to create a query on your own and gain some experience about how the data are stored for reporting. (The good news here is that most commercial software does a great job of making it accessible.) Most data extracts can be addressed through basic queries and an export to a commonly used data format.

Let's hope that the trend towards better use of data for decision making will continue. Data analysis can be a wonderful tool for informing and guiding problem solving, and it can provide unexpected insights. Data analysis can also be abused or manipulated; it can also be incorrect. Our goal as data experts is to encourage the informed use of accurate data — an aspirational goal, indeed.

SANFORD HESS is the information technology director for the City of Urbana, Illinois.