spendata

Spendata: Three Questions

What's the deliverable?

When the bar is low for a spend cube deliverable – say, a predefined set of dimensions in a read-only BI tool – then building a cube is easy. Any database will do for data prep, and even Excel will suffice. There have been hundreds of solutions constructed this way.

It's possible to do a lot better than this deliverable, and that's why Spendata scores so highly against its competition. Spendata enables analyses that are not possible to perform in a BI tool, such as real-time scenario modeling, qualitative segmentation, and more. See "How does Spendata differ?" for a list of some of these unique capabilities.

Even if your needs are simple, you'll find it much easier and less expensive to build cubes with Spendata than struggling with Excel or a relational database. With Spendata, updating a cube with new data, or making changes to an existing cube, is quick and easy. And, unlike home-grown solutions that publish to a read-only BI dashboard, users can map and family data at any time, link in new datasets, create new dimensions for new segmentations, and produce custom analyses without impacting your data team. With Spendata's inheritance feature, users can retain those custom analyses across refreshes of the data without having to redo any work.

What is the competitive landscape?

Some history is helpful. In 2001, ExpenseMap (Zeborg's offering – acquired by Emptoris, then IBM) was built with Access SQL queries and a custom front end.[1] Aside from the offline work of familying Vendors and mapping a Commodity hierarchy, the entirety of the ExpenseMap value proposition was the ability for end users to "slice and dice" in these and other dimensions. ExpenseMap was typical of the products of that period.

In 2004, BIQ introduced the ability for end users to map and family in pseudo-real time, and to build their own dimensions and workspaces. This enabled ad hoc analysis that is impossible with a read-only front end. BIQ also pioneered the ability to inject rolled-up data directly into existing spreadsheet models.

But post-BIQ, spend analysis technology has reverted to the older model – these days, just a data preparation service and a BI license. As in 2001, users cannot family or map on their own, build models, or create new data dimensions. The only technology advances have been in the area of offline mapping, with claims that "AI", specifically machine learning (ML), can produce better results than existing methodologies. In most cases it doesn't[2] – but for marketing purposes, the claim stands.

---

[1] BI tools were at the time not only expensive, but unfit for purpose.

[2] The poor quality of indirect spend classification by ML necessitates significant manual review. ML models trained on very specific categories of direct spending may be helpful.

So here we are 20 years later, with the majority of spend analysis solutions providing the same functionality as ExpenseMap did – a read-only "slice and dice" front end, with black-box offline mapping and familying processes performed by the vendor.

How does Spendata differ?

The differences between Spendata and the typical spend analysis deliverable are substantial and significant. Spendata is not just a data display tool; it is a ***platform for data analysis and modeling*** that happens to include a data display tool (a very powerful data display tool). Spendata's Views are integral, cooperative parts of other real-time analysis functionality, including derivations, many-to-many linking, and more. Users perform their analysis inside Spendata, where everything is quick, organized, and convenient, not outside the product in some third-party tool. Dumping data to Excel in order to analyze it was state-of-the-art 20 years ago, but it's an unacceptable answer today.

Here are some of the high points of Spendata's advanced functionality:

- **Multi-Level Derivation.** Every OLAP database "derives" dimensions – i.e., organizes them so that they can be rolled up by common keys. Unlike BI tools, Spendata derives dimensions in real time, as needed, whenever the derivation criteria change. Spendata has the ability to base a dimension derivation on one or more other derived dimensions, as well as on raw transactional data.

  In addition, derivations can refer to data assembled by previous derivation passes (script-only Columns), enabling the basis for a dimension to "span" transactions – in other words, to depend on multiple transactions separated widely in space/time. Derivations themselves are scriptable, so complex decisions can be made during the derivation process.

  Finally, Spendata allows dimensional measures – measures computed inside a rollup View – to be recursively added to Fact as a new column. Since other dimension derivations can depend on this recursive column, we can build models with Spendata that adapt in real time as the filtering of the basis column changes (see "Modeling and Dependencies," below).

- **Mapping and Familying.** It's naive to imagine that familying and mapping are limited to Vendor and Commodity, respectively – although most spend analysis vendors support only this. Spendata can map any dimension, including new ones created out of whole cloth, such as Preferred Vendor.

  All familying and mapping is done in real time, inside the tool, with a powerful UI. Spendata can reverse-engineer mappings done by previous tools, and convert them to Spendata rules that can be extended and maintained. Access to Spendata's auto-family and auto-mapping routines is readily available.

Finally, note that if dimension A's derivation depends on dimension B, and dimension B is mapped or familied, dimension A is automatically rebuilt. See "Modeling and Dependencies," below.

- **Indexing and Linking.** BI tools provide indexing capability in the form of many-to-one mappings between Fact and index. In addition, Spendata also provides many-to-many linking between datasets, with the ability to filter "across" datasets, showing either matches or non-matches to the filter criteria. Critically, Spendata does not link datasets at the transaction level; rather, it links them at the dimension level, which makes many-to-many mappings understandable. These high-level linkages are understandable and useful, as opposed to the largely useless transaction-level relationships provided by BI tools (which inherit a record-level mindset from the databases at their core).

  Additionally, Spendata provides the ability to manually edit Links, using its powerful Familying UI. This is useful when automation can't possibly infer a relationship, but a human can.

- **Modeling and Dependencies.** One can intuit this section from reading the previous entries, but dependencies are all over Spendata: mapping dependencies, linkage dependencies, derivation dependencies, user input dependencies, and View dependencies (see Multi-level Derivation," above). Creating and managing derivations, mappings, inputs, links, and views causes Spendata to automatically create and maintain an internal directed graph of the relationships, allowing it to automatically refresh downstream elements when an upstream element changes.

  This is "database as spreadsheet" functionality, and it means that Spendata can create spreadsheet-like models at database scale. Spendata is ideal for "what if" modeling: change an input or a mapping rule – or a filter that is the basis of recursive segmentation – and watch the entire workspace recalculate.

- **Publishing and Inheritance.** A key limitation to analysis flexibility is the refresh process, which wipes out user changes to the analysis model. Even if a user manages to obtain a read-write copy of the cube, and is able to augment it with custom analyses, the next refresh of the cube either overwrites those analyses, or the analyses become stand-alone and obsolete over time as reality changes. With Spendata's unique inheritance feature, custom analyses are fully preserved on refresh, and the underlying data are updated as usual.

- **Marking and Filtering.** BIQ was one of the first OLAP display products to introduce the idea of filtering all of the dimensions displayed on the screen at once, with one click. This was considered revolutionary at the time, especially since BIQ did this performantly on the relatively slow hardware of the day. But time moves on, and so do interface ideas.

  In practice, one quickly finds out that it's not at all convenient, in most cases, to have

every View filtered simultaneously, as BI products now do today. Rather, it's useful to have just one View filter, or just a constrained group of Views filter together. Furthermore, it's also useful to mark items independent of the filter that specified them, so those items can now be used in a different filter without the baggage of the specifying filter.

Spendata has reworked the standard BI interface to make both of these things easy and intuitive, and to provide filtering power unavailable with other tools.

- **Loading and Transforming Data.** Inconveniently for database and software designers, input data is almost never organized the same way as their ad hoc schema. Furthermore, input data usually varies in format over time, as users decide to include more or fewer fields, when personnel changes or software updates cause confusion in data extract procedures, or when new systems are added to the input -- or for a host of other reasons.

  A useful data analysis system like Spendata must not only adapt itself to input data, but also adapt itself to changes in the input data over time. This requires the following:
  - An easy way for users to "type" their data so that the system can catch and repair formatting errors
  - An easy way for users to understand whether data have been loaded correctly
  - An easy way for users to add data to the system when the data requires transformation because it may not match the format or semantics of previous data added to the system

  Click here and here for details.

- **API, Data Export, and Extensibility.** Most Spendata functions are available through its restful API, and those that aren't can be (trivially) added on request. Unlike BI tools that must limit what can be accomplished with scripting because of their payments model, Spendata imposes no restrictions at all. This includes data loading, filtering, reporting, data extraction, and so on. Automating any of Spendata's activities is therefore straightforward, using your favorite scripting language (bash, PowerShell, Perl, Python, node.js, etc.)

  Spendata's functionality is easily extended since derivations and dimensional measures and linkage functions are scriptable (in javascript) without limitation. Data can be:
  - extracted from Spendata from Views (into Excel, as WYSIWYG or in pivot table format),
  - extracted into a .csv file, or
  - injected into an existing Excel model.

  Transactional data can be extracted into Excel or into .csv, including both raw and derived columns, with the ability to extract only specific fields.

Spendata also provides an extract facility to a star schema, suitable for direct import to Power BI and other BI tools. Spendata ensures that related key names between Fact and indexes are unique, so that Power BI can automatically infer the schema structure without user input. And, because Spendata has organized the data as a star schema (rather than as a flat file), Power BI (and other BI tools) can function optimally in terms of filtering and roll-up performance.

- **Security and Data Sovereignty.** Spendata solves both of these problems by design: we simply do not move user data. User data stays on the user's machine, inside the corporate firewall, inside the country. All computation is performed locally.

  Spendata is a "software as a service" or SAS application served from the web. Unlike most SAS applications which depend on a server to perform the majority of the work, Spendata's server has no purpose other than to (1) download the Spendata application and (2) manage logins. The server doesn't "help" with computation in any way because the application resides entirely in the browser. There is no data flow from the browser to the Spendata server, with two exceptions:
    - Login requests
    - Requests for download of sample workspaces, sample data files, and Spendata's auto-family and auto-mapping knowledge bases

  This means that user data is never moved from the browser to the server. User data remains resident on the user machine at all times.

  It also means that Spendata has no access to your data, and cannot provide it to governments, litigants, or anyone else.