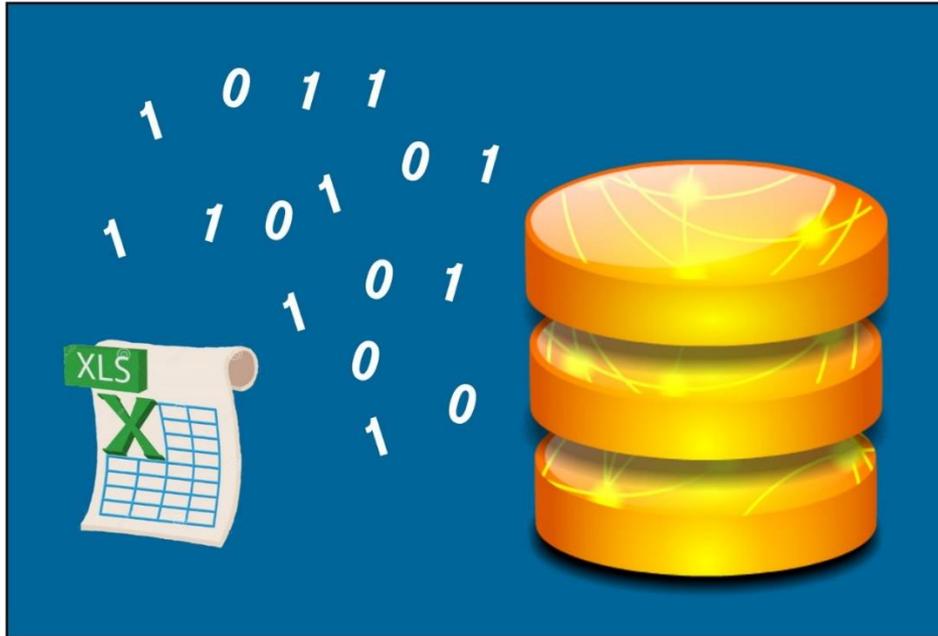# Refreshable Excel Reports: A Real Quick And Dirty Solution to Business Reporting

Shôn Ellerton, January 28, 2017

Towards the run-up to Christmas in 2016, I caught up with an ex-colleague and friend who had recently joined a major national telecommunications company as a delivery manager. Not very long into the conversation, he mentioned that the one thing he missed the most working in his new environment were *refreshable Excel reports*. Now you may be thinking out aloud, 'Refreshable Excel reports? Seriously?'. And yes, if you deduced that *refreshable Excel reports* are simply Excel spreadsheets linked to a data source via an ODBC connection, you are most assuredly correct! But allow me to let you on a little secret: refreshable reports are an extremely cheap and effective way to provide users data that they need to run their day-to-day tasks. So before you enter through the doors of the semantics reporting bazaar where you will run through a gauntlet of highly enthusiastic vendors wishing to sell you a product claiming that it will end all your business's reporting problems, see how far you can go with the below first.

So why am I writing about *refreshable Excel reports*?
Is there anything new about it? **No**.
Is it vaguely interesting? **Not very**.
Does it break new barriers in technology? **Definitely, no**.
However, is it innovative? **Well… read on**.

I had introduced *refreshable Excel reports* within a major national project providing easy accessibility to centralised data to hundreds of its personnel. It was meant to be a 'stop-gap' solution until a more advanced system would come into being. Half-way through the project, a more advanced system did get implemented (and with success I must add); however, did it kill off *refreshable Excel reports*? Not at all. In fact, *more and more* users wanted access to these reports. In one occasion, I read an email containing a reference to an *ASDR Report*, a report named with an acronym I was not familiar with. Much to my surprise, it meant *All Sites Data Report*, a fairly unimaginative name I gave to one of the first refreshable reports at the very beginning of the project knowing that this would only be a temporary measure until systems became more mature and advanced.


What is a refreshable report?

Just what is a refreshable Excel report? Remembering a time when Lotus 123 was still popular, I offer many humble apologies for those out there who may be using spreadsheet software other than Microsoft Excel; therefore, I will refer to these reports as refreshable reports from now on.

A refreshable report is a spreadsheet which is connected directly to a database to return a defined dataset. Strictly speaking, it should not be called a refreshable *report* as raw data is being merely extracted from a connected database to populate the rows and columns of the spreadsheet. However, semantics aside,

these spreadsheets are often created and customised by users to serve their *reporting* needs.
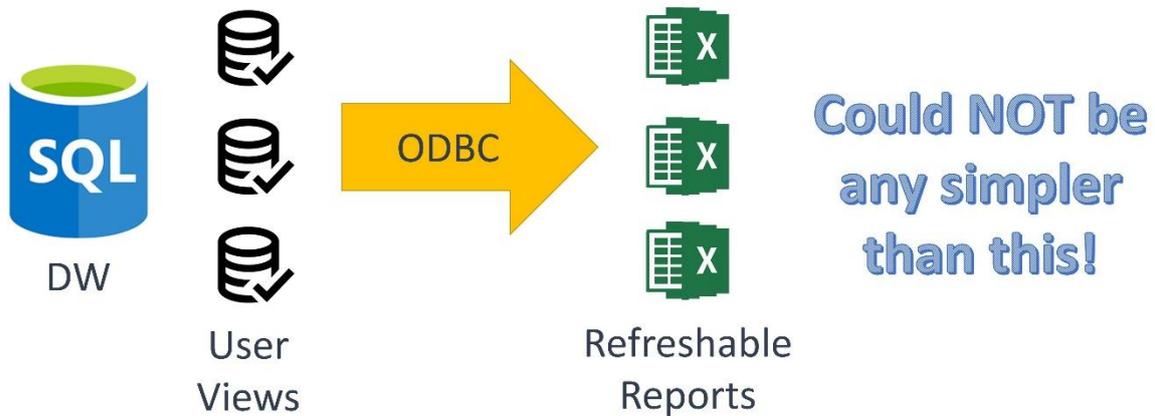
The spreadsheet returns columns and rows of data when the user 'refreshes' the data; either by clicking on the **Refresh Data** button or automatically if the spreadsheet was configured to do so. The user can customise their spreadsheets to return any number of rows and columns based on writing their own SQL queries, or alternatively, in the case of Excel, using the handy in-built query builder (Microsoft Query or Power Query), in which you can drag and drop columns and provide filtering on the dataset to be returned.

In addition to being fast and flexible, what makes refreshable reports particularly attractive is the fact that users can add their own computed columns either adjacent to the returned dataset or elsewhere in the workbook being derived from the returned data. As of writing, I have not come across another attractive solution whereby the user has that ability to have this much flexibility in extracting data. I have tried a wide variety of third-party BI reporting solutions (too many to

mention) but in each case, they became very 'clunky and sticky' to use when dealing with large column sets. Some of them also commanded quite high licence fees which proved to be unattractive from the project purse's perspective. I remember I once tried to revoke refreshable reports during the more mature stages of the project on the grounds of encouraging users to only use predefined reports generated by frontend applications. (It was an idea initiated by the IT department to reduce the number of direct user accesses to the underlying databases). This idea was quashed very quickly by those many project users who were using business-critical refreshable reports on a regular basis. In fear of having a lynch mob descend on me (and the IT department), refreshable reports were given a permanent place alongside the other more advanced reporting platforms which were developed during the project.

When to consider using refreshable reports

Refreshable reports are ideal for projects which are constantly evolving and are especially tolerant of changing processes, governance and methodology throughout a project's lifecycle. Also, substantial savings can be made by relegating the creation and publication of reports to project personnel rather than through report developers (a position traditionally hired under corporate IT).

However, there is no such thing as a 'free lunch'. Refreshable reports will probably not serve all your needs, especially for those projects requiring complex data derived from very large datasets. Refreshable reports are also not particularly good at presenting data visually in an interpretive way. Other tools, for example, Tableau, are extremely powerful and easy to use to 'slice and dice' data and to create those magnificent dashboard-type displays that are often presented in the boardroom. Reports which require complicated data output, for example, a report intertwined with subsets of data or groupings, may not work very well using refreshable reports. Other BI tools have been developed for this and are best suited on these occasions.
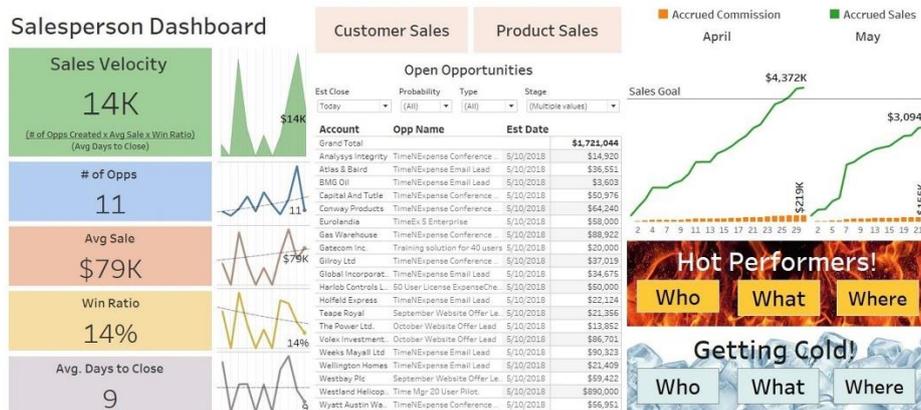
Security can be a concern with refreshable reports; however, this can be bolstered by ensuring that your reports can only access certain views, stored procedures or cubes as read-only in the underlying database or data warehouse. User authentication via active directory containers is far more preferable than using static database usernames and passwords as these are often exposed in the connection string in the data connection. Knowing how your database security works is essential and can be a safeguard from being exposed to hidden dangers; for example, I once saw a potentially-catastrophic loophole where synonyms via use of linked servers allowed read-only clients to have full access to a database on another server!

Accessibility to refreshable reports may be limiting factor as accessibility is usually contained to those who are working within the corporate network unless the database is cloud-based, which poses other questions on security, performance and sovereignty (where the data sits geographically). Users who do need to access refreshable data externally would normally use the services of a VPN tunnel.

Performance will be a massive issue if your users are allowed to extract huge sets of data from views based on multiple joins of underlying base tables. If these base tables are shared by an online data-entry client application and the refreshable report invokes a complicated query which returns large quantities of data from different tables, then it is quite likely that the online client application performance will suffer. The content of most reports generated by refreshable reports does not necessarily have to be *live at that minute*, therefore, it makes more sense to extract data from a data warehouse rather than the live transactional database. The data warehouse, in most cases, will automatically refresh its own data from the live transactional database on a periodic basis (daily, hourly, or whatever suits the occasion). Extracting data from multidimensional data cubes can foster a further increase in performance if they are available.
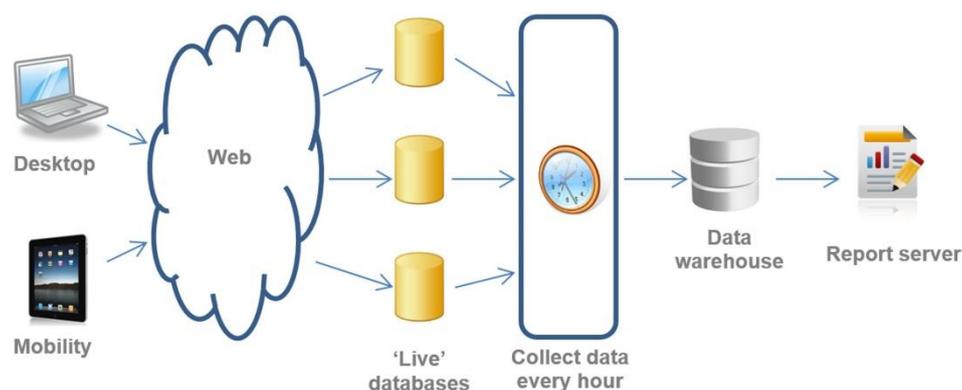
When implemented with the right safeguards, refreshable reports have proved to be an important ally in the provision of data to users in the business and are very popular with the 'data warriors' within the business, who specifically customise these reports by combining their data query skills along with their business-as-usual skills they need to drive the project.

Refreshable Report Requirements

Setting up refreshable reports is technically not that difficult; however, there are number of requirements which must or should be met.

Access to the database

This seems blatantly obvious but it is all too often that corporations deny direct access to the database. My experience here is that this little hurdle has become more of a problem with each year that passes by. I recall back in the early 2000s, that T-Mobile UK had a corporate database with a great many of the tables and views being open to the business via direct ODBC connections. It was a very successful model and minimised the need for IT to create custom reports for the hundreds of project groups within the business. The databases and enterprise frontend client applications were administered by the IT department. This was the *de facto* standard of data access in those days; however, the paradigm is gradually shifting towards using corporate-sanctioned reporting tools and denying direct database access to users. Security and consistency of reporting is undoubtedly better; however, the real downside is that you need those in the IT department to provide continuous support for your project's reporting needs, which usually means inter-departmental expenses incurred on the project.

Database and infrastructure design

The subject of a well-designed database requires an article by itself. Like any properly-designed database, it should be scalable and should be consistent in terms of schema and nomenclature. Security and performance depends on which objects the users have access to. Moreover, the infrastructure supporting the database and its network connectivity should be sufficient to cater for your users' needs. The IT department can assess and perform (or, at least, suggest) upgrades to the infrastructure if necessary; however, this can be a protracted and weary exercise within some of the larger corporations.
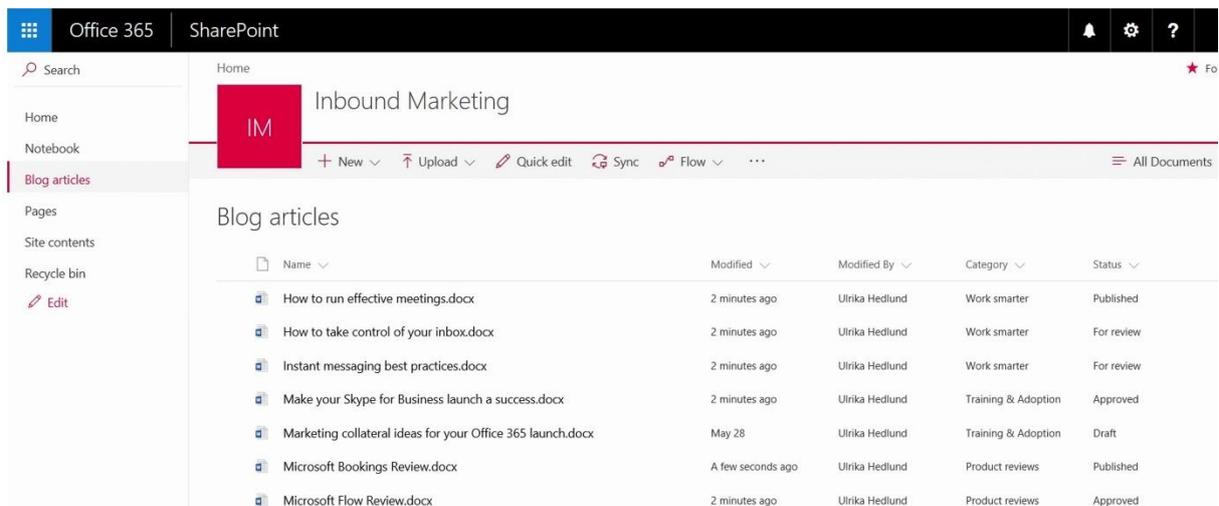
User Documentation

Creating custom refreshable reports does require some degree of skill; therefore, it is important to ensure that there are instructional step-by-step guides available

for users in order to create a report from scratch. The other important documentation that should be available are *data dictionaries* of the underlying databases. This is a godsend for those who need to understand what the data is, for example, definitions, data types, field lengths, relationships, keys etc. Do not assume that you need to be in the IT field to understand data dictionaries. It is surprising how many project users not working in the IT space have extensive knowledge of IT data skills. From my own personal experience, I have suffered the evils of pigeonholing in which employers discount candidates for IT-related jobs if they do not either have a degree in computer science or business computing. My degree was in civil engineering; thank you for asking!

A repository of framework or 'skeleton' refreshable reports

Why not create a selection of half-completed refreshable reports and save them to a common place where users can access it? For example, if you have a Sharepoint, or any other Document Management System, save them there. The user can then download the report and then customise it to their own specific wishes.

Summing it up

The Good News about refreshable reports

- Refreshable reports require very little maintenance if due diligence and thought went into the design of the data structure that sits behind it;
- Adapts to quickly-evolving projects whereby they can be created practically 'on the fly' with little or no lag time for users to consume the data;
- They 'kill off' the myriad of spreadsheets with differing data by extracting data from one centralised source; thus, data integrity is ensured;
- They offer the user almost complete flexibility with what data to return; very important in a business with a wide variation of different reporting requirements;
- Extremely cheap to implement not requiring software engineers and application deployment managers to rollout sophisticated frontend applications. High licencing fees by using third-party reporting tools could be avoided as well;
- Spreadsheet can be shipped to any user in its native format. The user can remove the datalink if a static copy is kept and archived;
- Cheap;
- And most importantly, the user can 'build the spreadsheet' around the refreshable data to suit their own reporting needs.

And now, the not-so-good news:

- Can induce performance issues to your database if your database is not designed properly;
- Changes to underlying database can seriously disrupt existing refreshable reports if changes are not communicated to the business;
- Not suited to complex datasets;
- Security can be an issue if permissions in the underlying database are not set properly;
- Your IT department may not like it!