Anaplan XL Reporting (AXL)

Getting the most out of AXL: Effective uses and practical recommendations

What Anaplan XL Reporting does best ____ *3*

The sweet spot: Where AXL delivers the most value ____ *4*

Areas of caution: When AXL may not be the ideal fit ____ *5*

Designing sustainable AXL reports ____ *7*

When to use grids, freeform, and formula reports ____ *9*

Understanding query behavior ____ *9*

Things to keep in mind ____ *10*

Planning for scale and complexity ____ *11*

Where a different tool might serve you better ____ *11*

Recommended guardrails ____ *12*

When to bring in extra help ____ *12*

A pre-deployment checklist ____ *13*

Final thought ____ *13*

Anaplan XL Reporting bridges the planning power of the Anaplan platform with the flexibility of Microsoft Excel. It lets you interact with live Anaplan data dynamically from your desktop. Used well, it eliminates a lot of repetitive, error-prone work. Used incorrectly, like any tool, it can create more problems than it solves. AXL is best suited for summarized, highly formatted reports and ad hoc reporting for power users, refreshed on demand.

These recommendations are intended to spark ideas and provide general guidance on how to use AXL with your Anaplan models. There's no single best approach for every team, model, or use case, so treat this as a framework rather than a rigid checklist. The ideal application of AXL often depends on the specific context of your models, data, and team's objectives. Finding the right balance is key.

Who this is for

This guide is written for the people who build and maintain AXL reports: financial analysts, operations teams, model builders, and other report owners who are connecting Anaplan data using the installed Excel Add-in. This guide focuses exclusively on Anaplan XL Reporting connected to Anaplan Models. It assumes AXL is already installed and in use at your organization. If you are new to Anaplan XL Reporting, see "[Start Here] Anaplan XL Reporting" under Additional Resources to get started first. Return here to learn about optimal uses of AXL and how to design reports that perform well and hold up over time, or when another part of the Anaplan platform is better suited.

Key topics covered

This guide walks you through:

- **Where AXL delivers the most value:** A summary of the situations where AXL works best, from on-the-fly analysis and executive-ready reporting to reusable, shared report templates.
- **When you should use caution:** A look at when a different solution is the safer choice, such as ETL (Extract, Transform & Load) work, transactional-level reporting, heavy writeback, and workflow-driven processes.
- **How to design sustainable reports:** Practical techniques for structuring source views and workbooks so reports stay fast, accurate, and easy to maintain.
- **When to use grids, freeform, or formula reports:** A side-by-side look at how each AXL report type trades off performance, scale, and customizability, so you can pick the right one for the job.
- **Performance and query behavior:** How queries work behind the scenes, what counts as one, and the everyday habits, refresh settings, slicers, and linking features that keep query volume under control.
- **What you should keep in mind:** The less obvious mechanics such as ALM (Application Lifecycle Management) promotions, cross-model hierarchy mapping, and writeback scope that you might miss until they cause a problem.
- **Planning for scale and complexity:** How growing user counts, data volumes, and report complexity change the calculus, and the signals that suggest a report has outgrown a single AXL workbook.
- **When a different tool is the better choice:** Guidance on when to step back to the Anaplan UX, Workflow, Polaris, or another part of the platform instead of AXL.
- **Recommended guardrails:** A consolidated set of practical thresholds, connection counts, writeback scope, refresh settings, and more, to use as sensible starting points when designing a new report.
- **When to bring in additional resources:** When to loop in IT, an internal champion, Professional Services, or a certified partner.
- **Pre-deployment checklis**t: A short list of questions to run through before sharing a new AXL report widely, covering source design, refresh settings, distribution, and writeback scope.

Quick takeaways

Before diving in, here are a few practical habits drawn from real day-to-day AXL usage.

1. **Watch how you store and share your file:** Avoid storing AXL workbooks on OneDrive or SharePoint and turn off AutoSave. Both can interfere with how AXL manages its connections and refreshes. Also, watch for auto-synced folders. Some corporate IT setups silently redirect the Desktop or Documents folder into OneDrive (Windows' "Known Folder Move"), so a workbook can end up cloud-synced even when not intentionally selected. Each user should have their own version of the file installed on their local computer.
2. **Avoid opening the same workbook from multiple devices or Excel sessions at once.** Having the same AXL file open by multiple people, or having multiple AXL files open at once, can also interfere with connections. Once the connections are “broken,” the file will need to be rebuilt; we recommend having a template copy for all critical workbooks.
3. **Split large reporting needs across workbooks.** Beyond keeping connections and grids to a reasonable number in a single file, consider splitting a large reporting need into a few focused workbooks rather than one file that tries to hold everything.
4. **Individual writeback, not concurrent planning.** AXL is built for a small, trained user making their own updates, not for multiple people writing to the same data simultaneously. Concurrent writeback is exactly where conflicts and errors show up.
5. **Reporting layer, not data warehouse.** For large-scale or multi-source data, use Polaris or the Anaplan UX for reporting rather than pulling everything into Excel via AXL.

Remember: Anaplan XL Reporting is a reporting tool. It was developed to make it easy to connect and report on your data right from Excel. First and foremost, when you think about how to use AXL, consider Excel's benefits and limitations.

What Anaplan XL Reporting does best

AXL was built for Excel-native reporting and analysis: an Excel desktop add-in connected to governed Anaplan data. It delivers the most value for Excel power users, operations or analyst teams, Anaplan model builders, and trained report builders who need to:

- Leverage native Excel formatting, formulas, and workbook structure.
- Refresh data directly from Anaplan without manual copy-pasting.

Think of AXL as a personal productivity tool for specific, high-value jobs: producing the controlled, beautifully formatted reports and the rich ad hoc analysis that reporting teams own. It complements the Anaplan user experience (UX) rather than replacing it. It removes the manual export-and-rebuild cycle that consumes so much analyst time.

The sweet spot: Where AXL delivers the most value

AXL is built for flexible, custom formatting, and on-demand interaction with Anaplan data. It tends to deliver the most value in situations such as these.

Use AXL for	What it looks like
Ad hoc, one-off analysis	Pull data from an Anaplan model into Excel to validate logic, explore scenarios, or determine the need for a new report before investing time in building a permanent dashboard in the Anaplan UX.
Executive-ready, highly styled reporting	Leverage Excel's capabilities to build custom reports and visualizations, from pixel-perfect executive board books and financial statements to specific layouts such as butterfly reports or incorporating connected charts.
Reporting across models	Create a single workbook that pulls from multiple Anaplan models (for example, Finance and HR) while respecting each user's underlying permissions. These would be independent grids, freeform, or formula reports that live within the same workbook or even the same worksheet. ***Note**: Blending sources or connections into a single output report is not recommended.*
Data Validation	Compare Anaplan data against existing Excel data for spot-checking. ***Note**: This is not meant for heavy reconciliations or full data validation processes.
Data discovery	Slice, filter, and combine live, governed data to explore the business from new angles without cluttering the core model with one-off exploratory reports.
Personalized reporting	Let teams build their own version of a shared template, their own calculations, and their own views. They can do this without changing the version other teams use or changing the underlying Anaplan modules.
Summarized customized grouping	Use AXL to create customized groups of list members to summarize your data in new ways for novel insights. ***Note**: This should not be used for heavy transformation, cross-hierarchy levels, or transformations that will require ongoing maintenance. Customized groupings are best for experienced AXL power users with strong knowledge of the connected Anaplan models. It is not recommended for distributed reports OR any uses involving writeback.*
Reusable reporting	Build a single master template distributed to multiple users. When the user opens their individual workbook, they only see data that aligns with their specific permissions. For smaller lists, a feature such as Propagate can generate a separate workbook with tabs per cost center or business unit from the template workbook.
Small-team-owned report packs that complement the UX	Summarized Excel output that a small, trained team refreshes and maintains, sitting alongside your UX dashboards rather than replacing them.

Real-world examples

- A finance team produces a monthly board pack using Anaplan data.
- An FP&A analyst refreshes a highly formatted Excel report for management review.
- A reporting team builds summarized Excel reports that complement UX dashboards.
- Power users perform impromptu analysis on Anaplan data directly in Excel.
- A small team prepares report exhibits that are manually added into PowerPoint or PDF packs.

Areas of caution: When AXL may not be the ideal fit

While powerful, AXL isn't the correct solution for every task. Using it for the wrong purpose can create inefficiencies and governance challenges. We recommend careful consideration in these areas:

AXL isn't ideal for	Why it's not recommended
ELT (Extract, Load, Transform) Processes, Data Transformations	Avoid using AX for heavy data transformation. Not only does this create a parallel "**source of truth**", but it also adds unnecessary data and calculations to the Excel workbook, impacting the performance of the AXL workbook and the source system. Over time, maintaining the rules and mapping tables will become increasingly challenging. For data governance, we recommend that all data transformation logic happen in ADO, a Data Hub, a data management system, or a dedicated ETL tool to ensure transparency and control. Otherwise, you end up with a second, ungoverned data store.
Transactional-level or dataset reporting	Excel has inherent size limitations and struggles with large datasets. It was not designed to handle millions of rows of raw, transactional data. For this reason, AXL shouldn't be used to pull massive datasets into a workbook for detailed review or as a data store. Instead use AXL to bring in summarized, dimensional data, and then use the pivoting, slicing, and filtering features to report on the exact level of detail needed to make decisions. Excel was never built to be a data store. You will experience poor performance long before you hit any technical limit.
Complex or high-volume writebacks	AXL is well-suited for light data entry, such as updating high-level assumptions or making minor adjustments. However, it shouldn't be used for large-scale or heavily dimensionalized writebacks. This approach has many problems: it lacks a clear audit trail, relies on users all using the correct workbook template, and lacks native collaboration features. For these reasons, the Anaplan UX is the more robust, straightforward, and traceable solution for heavy data entry.
Complicated, in-depth data analysis	When a report starts comparing Anaplan data against data kept locally in Excel or leans on a deep chain of custom formulas to reconstruct logic that isn't in the model, the workbook has effectively become its own parallel use case. That logic lives outside Anaplan's governance, is not visible to anyone else, and is easy to diverge over time. If the analysis is valuable, it belongs back in the model, not permanently embedded in one person's Excel file. Finally, the more complicated the formula, the more drag this creates on the file.
The "**one-stop-shop"** workbook	A single tab, or workbook, trying to hold everything, sometimes 300 grids or more, refreshes slowly, is difficult to troubleshoot, and usually means no one fully understands what the whole file is doing anymore. You should split these large, unwieldy, multi-source compilations into smaller focused, purpose-built reports.
Processes requiring guided workflow	AXL doesn't support the guided workflows, task management, and approvals, which are native to the Anaplan UX. If your process requires collaboration and a clear sequence of steps with defined user hand-offs, you should build it in the Anaplan UX instead.
Collaboration within a workbook	AXL workbooks are designed to live locally on each person's PC. This means that it's not designed for more than one person to view, edit, or interact with the same workbook at the same time. There's no real-time co-editing (commenting, pivoting, filtering, etc.), version merging, or other user awareness the way a governed multi-user interface has, such as Anaplan.
High Concurrency	When multiple people are working in or refreshing the same data from the same connections at the same time, every one of those concurrent updates in their files also hits the source model via API calls that stack up. This adds load and impacts the performance in both the model and the workbooks. The larger the API calls (more cells updating, more formulas, more connections), the greater the impact.
Real-time operational dashboarding	Anaplan UX is better suited for real-time operational dashboards that require instant updates based on model changes, whereas AXL is for periodic and on-demand reporting.

AXL isn’t ideal for	Why it’s not recommended
Mass distribution	AXL doesn’t include governed distribution or bursting. The moment each user downloads their copy of the file, they are no longer connected to the template and tend to drift. Changes made to the template will require a new download, and there’s no visibility into how individual users have modified the reports or the settings. This is manageable with a small team of trained users, but harder at scale.
Distributing to non-AXL users	A recipient without AXL and the required Anaplan access can't refresh an AXL workbook, so what they actually receive is a static snapshot report. This may sound like a good solution, however, slicers, pivoting, advanced filtering, dynamic charts, and some custom calculations won’t work in workbooks that aren’t connected. Building a version that still functions without those features usually means falling back on a large grid or a heavy reliance on formulas. That trade-off works against other best practices covered earlier: it's the "one-stop-shop" and formula-heavy patterns to avoid. If the goal is genuinely live, interactive access, the recipient needs to be a licensed AXL user.
Bursting reports	Using AXL's **Propagate Across** functionality can generate a new workbook or worksheet per hierarchy member. However, **Propagate Across** is a heavy action on both the workbook and the model. Generating a worksheet for each hierarchy member requires an API call per member, so anything beyond a handful of members should be approached very cautiously, and even then, it runs grid by grid rather than for a full workbook at once. There's no native way to burst a complete, multi-grid report from AXL desktop. If the real need is distributing a report broadly, that's a sign to look at a BI tool or a governed distribution process instead, not a way to use Propagate as a workaround.
BI tool replacement	While AXL is excellent for reporting, it's not designed to replace enterprise Business Intelligence platforms for large-scale, interactive data visualization and organization-wide analytics.
Audit and compliance tracking	AXL is a reporting tool, not a system of record. As a desktop add-in, its interactions, including exactly which data was accessed through a given connection, aren’t tracked the way UX or model activity is. Writebacks from AXL will show up in cell history but show up in model history as a single bulk data change rather than as individually attributable entries. This means they don’t give the same audit trail a governed process would. AXL files should be kept as read-only, preventing any accidental edits or updates. However, this means that any notes, commentary, or edits won’t be available for future reference in this workbook.

Designing sustainable AXL reports

To get the best AXL experience, you need thoughtful design in both your source models in Anaplan, and your Excel reports. A clean source won't save a cluttered report, and a well-designed report can't make up for a slow or poorly structured source.

Working through the recommendations below early in your planning will help you design a deployment that scales gracefully. It is worth honestly comparing them to your operational model and expectations before you build.

Recommendation	Why it matters
Start from a healthy source	AXL is a reporting layer, not a fix for model performance or scale issues. Confirm the underlying model is optimized before creating your connection. A slow source makes for a slow report.
Keep the source view lean and purpose-built	Connect to a small, well-structured reporting module rather than a large, multi-purpose calculation module. AXL filters locally in Excel rather than in Anaplan, so it will still pull zeros and blanks, even from a Polaris source where those cells aren't considered populated.
Curate line items for your audience	Only include line items you are comfortable with every report user seeing. A user's security controls the data they see but does not restrict specific line items within those modules. To avoid exposing interim calculations or sensitive data accidentally, you should built purpose-built modules.
Break it up, keep each workbook focused	Avoid stacking many grids into one file. A simpler, more intentional workbook refreshes faster and is easier to maintain. Many workbooks is better than many tabs.
Favor dimensional data over flattened structures	Concatenated or flattened dimensions are hard to filter and lose the native pivoting and slicing functionality AXL provides. Summarized, dimensional sources keep that flexibility intact.
Avoid the any by any trap	Minimize the number of cells in grids or freeform reports, rely on slicers, ranges, or other techniques to do the heavy lifting to reduce what is exposed in the grid. Avoid nesting many dimensions across rows and columns.
Reduce calculations and data manipulations	Every calculation or manipulation performed inside the workbook, custom formulas, advanced filters, grid-level calculations, adds processing overhead on top of the initial data pull. To keep a single source of truth, you will want to push logic back into the Anaplan model where it belongs. Minimizing your Excel-side modifications and calculations reduces maintenance burden and your reports stay fast.
Keep writeback small and intentional	Writeback works best in small, deliberate batches, not as something automatic or constant. Refresh the grid immediately before writing back, in case someone else has changed the underlying data before your change. Treat writeback as a supplemental update not as part of your organization's core planning process. That process belongs in the Anaplan UX, where you can track and govern it properly.
Refresh deliberately	For larger datasets consider changing the **Refresh** settings found in the workbook options or grid properties. When possible, try to refresh on demand only. With writeback, update your settings to **On Request** in **Grid Properties. Automatic** mode creates an API writeback each entry made, which is less performant for the workbook and Anaplan model. If someone else may have updated the same cells before your writeback submission, you may want to refresh your grid before entering your values.
Use some functionality only on small dimensionality	There is some functionality that works well when tied to small lists, but when applied against larger list sets creates performance concerns. You should test out all functionality at true scale that includes high volumes of data refreshes or items that are dynamically applied like formatting.
Include a Parameters Worksheet	Add a centralized worksheet that houses the ranges, slicers, XLink outputs, mapping tables, and other reusable data. This makes it easier to reference, identify, and maintain the items you use throughout our workbook.

Recommendation	Why it matters
Blend AXL and Excel features carefully	Named Ranges and VLOOKUP are natural fits: Named Ranges make formulas easier to read, easier to filter by, and easier to reference elsewhere in the workbook, and VLOOKUP is a reliable way to blend AXL-connected data with other Excel content without disturbing the underlying connection. Avoid VBA, macros, volatile formulas (OFFSET, NOW, TODAY, INFO, CELL, INDIRECT, RAND), and native Excel filtering or sorting directly on AXL-connected ranges. These can interfere with what happens during a refresh and produce unpredictable results. Volatile formulas also force a full workbook recalculation any time there's a change. This can have a much bigger impact than that single function alone. Excel's native filtering is not recommended with AXL, as it can also affect AXL report output. ***Note**: Using CoPilot functionality with AXL workbooks is not recommended. CoPilot requires files to be stored on and SharePoint which can make the AXL connections unstable.
Use naming conventions that make AXL-ready content easy to find	A consistent suffix, such as _AXL, on reporting modules and saved views helps users search and filter to the appropriate target connections inside the add-in.
Trust, but verify, inherited security	AXL carries over everything from the Anaplan model: • Dynamic Cell Access (DCA) • Read/write access by version, module, or list • Selective Access on lists UX Techniques such as show/hide or applying filters don't apply to the underlying modules. This means they aren't applied in AXL reports either. Review all permissions for all models before a template goes out, since that determines the users access in AXL. Many users have access to more than one model or use case. Confirm permissions with all teams before deploying AXL to your ecosystem to ensure that data is not accidentally exposed through unintended AXL connections.
Create with a golden template	Local copies keep the file safe and provide user flexibility, but they also let stale versions spread. Maintain one master, version-controlled template that everyone downloads periodically, and provides a backup in case any connections become unstable.
Respect Excel's limits going in	Excel can technically hold many rows, but the experience degrades long before any hard limit is reached. Keep AXL reports summarized and dimensional rather than raw data dumps.
Limit connections per workbook	Every additional module, source, or connection adds refresh time and a maintenance burden. Fewer, well-scoped connections outperform one workbook trying to do everything.
Assign clear ownership	A named owner who understands both Excel and the underlying model keep a workbook healthy long after it is first built.
Ecosystem support is available	AXL may require your local IT team for installation and upgrades. You will also want an "AXL Admin", someone internally who understands Excel, AXL, and the underlying model(s) well enough to help others when something breaks. Before scaling adoption beyond a few people, confirm that you have the right team in place ready to support your users. IT should be prepared for the technical side and an internal champion trained and available for day-to-day questions.

The most effective AXL reports are built on a foundation of thoughtful design in both Anaplan and Excel.

When to use grids, freeform, and formula reports

AXL offers three main ways to connect to Anaplan data via Excel, each with its own trade-off between performance, scale, and customizability. Picking the appropriate option for the job matters as much as designing the report itself.

Report type	Best for	Trade-offs
Grid	Straightforward, tabular pulls that mirror the model's structure. Standard reporting, quick one-off analysis, and most everyday use cases.	Least customizable layout, but the most efficient and predictable option: one query per grid.
Freeform	Highly formatted, presentation-style reports, such as board packs, where the layout itself matters and cells need to be placed flexibly around the workbook.	Still one query similar to a grid, with much more design flexibility, but more work to build and maintain than a standard grid.
Formula	Precise, cell-level customization when a grid or freeform layout definitively can't get the job done.	Every single formula is its own query, so formula-heavy reports scale the worst and put the most load on performance. Use sparingly and only when necessary. It can lead to API throttling or performance degradation, or even model blocking for other users of the source Anaplan model, not just the individual user's workbook.

As a general rule: Start with a grid, move to freeform if more layout control is needed, and reach for individual formulas only for cells that truly require them, not as the default way to build a report.

Understanding query behavior

Every interaction with an AXL-connected report triggers a query against the source model. The query volume is one of the biggest levers for both report speed and model impact. Understanding what triggers a query will help you improve performance.

- **What counts as a query:** Grids and Freeform reports each count as a single query when they load.
- **Formula reports are different:** Every individual formula is its own separate query, so a sheet full of formulas can generate far more query volume than a single grid.
- **Everyday interactions add up:** Every slicer change, pivot, expand or collapse, or similar interaction triggers a new query unless the report is set to refresh on demand rather than automatically. Using intentional refreshes after you have made all changes to your view limits the consecutive queries. This will improve workbook performance and reduce the impact on the connected model. This has greater impact when concurrent users hit the same models.
- **Linking interactions count too:** Tracked clicks in grids, Xl3Links, active cell tracking, dynamic visualizations, breakouts, and Grid Links all generate additional queries behind the scenes. Use them only where they add real value to the report, and minimally in a single workbook.
- **Minimize size and frequency:** Favor a set of smaller, well-scoped grids with more slicers and defined ranges over a single enormous grid that tries to cover everything. Avoid over-nesting.
- **Watch for calculation-heavy features:** Custom calculations, advanced filters, and grid-level calculations all add to query load and affect performance, so use them deliberately, not by default.
- **Set refresh to on-demand where possible:** Review every refresh and query setting deliberately, rather than relying on defaults or arbitrarily large numbers or long query timeouts. Refreshes should be set to on-demand, meaning the report updates only when a user actively refreshes the workbook, grids, or formulae. That is slightly less convenient than automatic refresh, but it's intentional and keeps query volume and load on the source model under control.
- **Model performance matters:** Your query times directly correlate to the connected model's underlying performance. Things like Model open times will still apply, and the initial connection has to validate access, permissions, and the model's structure. The more connections you have in the workbook, the longer this will take. Once the connections are established and the Anaplan models are 'open,' it is much faster to just refresh the data. However, models with naturally slow performance, high concurrency, or high volumes or frequency of actions will result in longer query times even if the workbooks are simple and designed with best practices.

Things to keep in mind

None of these are reasons to avoid AXL. They're what separates a workbook that holds up for years from one that turns into a headache.

- **It's still Excel.** AXL inherits Excel's native ceilings on performance and file size, and it runs as a Windows-installed, on-prem add-in rather than a hosted experience.
- **Every interaction hits the model directly.** A slicer change, a refresh, and a write-back all send a request straight to the Anaplan model, no more efficient than a standard import or direct entry. Concurrent AXL use can slow things down for others working in the same model or in their own AXL files at the same time.
- **Version mismatches can break things quietly.** A workbook built or saved in a newer version of AXL is not guaranteed to work correctly for a teammate running an older one. Keep everyone on the same, current release, especially for shared or templated workbooks, and treat an out-of-date install as a likely cause when something that used to work suddenly doesn't.
- **Use .xlsx or .xls file types only**: Other Excel file types are not supported and may lead to file instability.
- **Moving environments can shift list numbering.** If you promote a workbook from a test environment to production, or rebuild a list, the underlying member IDs can change even when the display names look the same. Revalidate your reports after any ALM promotion or list rebuild rather than assuming they carried over cleanly, even when using **Edit and Remap** functionality.
- **Reporting across models or sources takes extra mapping work.** Similar to above, AXL doesn’t automatically recognize a list member in one Anaplan model as the same list member in another model, or in an external system. Plan for an explicit hierarchy-mapping step whenever a report spans multiple models or sources. If the list names are identical across sources, using the **caption** for the list members will make this easier.
- **Writeback works best when properly scoped and infrequent.** Writeback is a good fit for a small, trained group making occasional, well-defined minimal updates. It is not designed to collect input from many people at once. Having a non-AXL user open the file can break the connections.
- **Polaris compression doesn't carry over to Excel.** Polaris models work fine in AXL; that's not in question, but they can blow up a workbook fast if the report isn't scoped carefully. Polaris handles huge volumes of sparse data efficiently, but that compression doesn’t carry over to AXL, where it pulls through all cells, even the zeros and blanks. With all filtering applied locally in Excel, this can quickly bloat your workbook and exceed recommended guardrails for cell updates, complex filtering, or maximum rows. Polaris reporting through AXL should stay at a summarized, high-level hierarchy; don’t assume what works in the model will work the same way in a workbook.
- **On-Demand Calculations (ODC) may impact performance**: Polaris uses ODC to compute a line item's values only as they're viewed, rather than pre-calculating everything up front. Queries of a high-level, summarized value still require Polaris to calculate the full path needed to build that rollup, so if the underlying module has deep hierarchies, a report's layout and selections can trigger real calculation effort even when the output looks simple. A high-level report pulled from a shallow, purpose-built module has far less impact on the model than the same high-level view pulled from a module with many dimensions and deep hierarchies underneath it.
- **Moderation is key for AXLFormats.** The built-in AXL grid formatting helps you build presentation-ready, customized reporting using familiar Excel formatting options. However, excessive use of unique formatting across multiple grids can slow performance. Disable grid formatting when you don’t need it, or reduce formatting rows by applying formatting across similar members rather than repeating individual member names.

Planning for scale and complexity

An Anaplan XL report rarely succeeds or fails because of the specific use case. A monthly finance report pack is a great fit when a small team owns the workbook and refreshes Anaplan data into a controlled report. The same use case becomes something to avoid when hundreds of stakeholders are expected to collaborate in a single shared file. The table below shows what a well-planned deployment looks like, where to plan carefully, and where you may need to change your approach.

Dimension	Designed for success	Plan carefully	When to rethink your approach
User scale	1–10 trained report builders or finance power users	10–20 users refreshing or updating controlled workbooks	Hundreds of users expected to refresh, edit, comment, or write back
Distribution	A small team refreshes and distributes controlled static outputs	Some users refresh their own copies of the workbook	Live-connected workbooks sent broadly to users including those without AXL
Collaboration	A small team prepares analysis or reporting	Limited review by a small group of trained users	One shared workbook as the primary collaboration, commentary, or approval solution
Writeback	None required	Limited updates by trained users, clearly scoped and tested	Broad use of writeback, mass commentary, or transactional inputs
Data volume	Summarized reporting with controlled workbook design	Multiple connections, a few grids close to the limit, and several complex formulas	High-volume extraction, heavy transformation, or data standardization
Operating model	A clear owner, trained users, a consistent refresh process, and documented maintenance schedule	Ownership or support model is still forming	No owner, no enablement, and no IT support – we recommend you settle these before rollout

Where a different tool might serve you better

AXL is one part of a broader Anaplan toolkit, and a few categories of work are better handled elsewhere.

If you're looking at	Consider instead
Extract, load, and transform (ETL) work	Anaplan Data Hub, Anaplan Data Orchestrator (ADO), a dedicated data management system, or ETL tool
Transactional-level or massive datasets	Polaris Engine, data warehouse, or a BI tool
Complex or high-volume writeback	Anaplan UX or Import Process
Guided workflow, task management, or approvals	Workflow
Guided planning or data entry across many users	Anaplan UX
High-dimensional modeling at scale	Polaris
On demand querying without managing a dedicated workbook	Analyst
Enterprise-wide data standardization across systems	ADO, a Master Data Management (MDM) solution, or data warehouse/lake

Recommended guardrails

Use these recommended guardrails as a starting point for calibration. Even when staying within the recommendations, a poor underlying model or workbook design still doesn't guarantee performance. You need to account for the whole ecosystem.

Guardrail	Recommended threshold
AXL workbooks open at once	One
Connections per workbook	15 or fewer (ceiling 20 to 25)
Grids per workbook	30 or fewer (ceiling ~50)
Worksheets per workbook	10 or fewer (ceiling ~50)
Formula cells per workbook	Well under ~1,000.
Cells written back per edit	1,000 or fewer
Cells per grid	Aim for ~100K or fewer
Cells per workbook	Aim for ~1 million across all worksheets
Maximum cellset request size	Keep at product default
List member ceiling for a connection	*1,000,000 members (hard limit)
Read/write users per file at a time	One
Query timeout	Keep at product default; do not extend it just to push through a slow report
Text limit per line	*8,190
File size	*Depending on your Excel Version

*Hard limits from Excel

When to bring in extra help

Most AXL deployments are well within reach of a capable reporting team. In a few cases, talking with your Internal Anaplan Center of Excellence (COE) team, Anaplan account team, a partner, or Professional Services will save time and de-risk the build.

Consider professional services or solution architecture when:

- The use case includes a complex workbook design.
- Introducing new datasets or entirely new use cases to AXL.
- Trying to build anything that requires in-depth analysis.
- The use case involves large refreshes or multiple connections.
- There are model performance concerns.
- You are deciding between AXL, UX, Polaris, Data Hub, ADO, BI, or an MDM approach.
- AXL is part of a larger reporting or planning operating model or implementation you want to get right.

Involve IT when:

- Installation or upgrade support is required.
- Users may not have permission to install Excel add-ins.
- You need to deploy across many users.
- Security, source connections, or desktop environment requirements need validation.
- An offline installation is required and users need support on updating prerequisites

A pre-deployment checklist

Before you build, it is worth confirming that the foundations are in place. The following conditions tend to be true of the smoothest AXL deployments:

- There is a clear workbook owner.
- The workbook fits within one of the ideal uses of AXL above.
- The workbook refresh process is defined and understood.
- A validation and redistribution plan is in place when list changes impact defined selections.
- Report builders are trained on AXL and Excel.
- Performance has been tested with realistic data and workbook design.
- Any writeback is limited in scope and clearly controlled.
- User scale and concurrency are understood.
- Anaplan and AXL access assumptions have been defined and vetted for each user group.
- IT can support installation, upgrades, and the supported Excel environment.
- Governance and maintenance processes are documented.
- The team understands what AXL does, and which parts of the process belong to other Anaplan capabilities.
- The report builders have appropriate knowledge of the Anaplan model and data.
- There is a change management plan that includes communicating the rollout, training users, and managing the transition plan from the old reporting process to AXL.

Final thought

AXL earns its place by being genuinely good at Excel-native reporting and analysis on top of a well-designed Anaplan model, not by trying to be everything for all situations. The teams that get the most out of it share a few habits: a clear owner, a lean, well-governed design, and an honest sense of when a report has outgrown a single workbook.