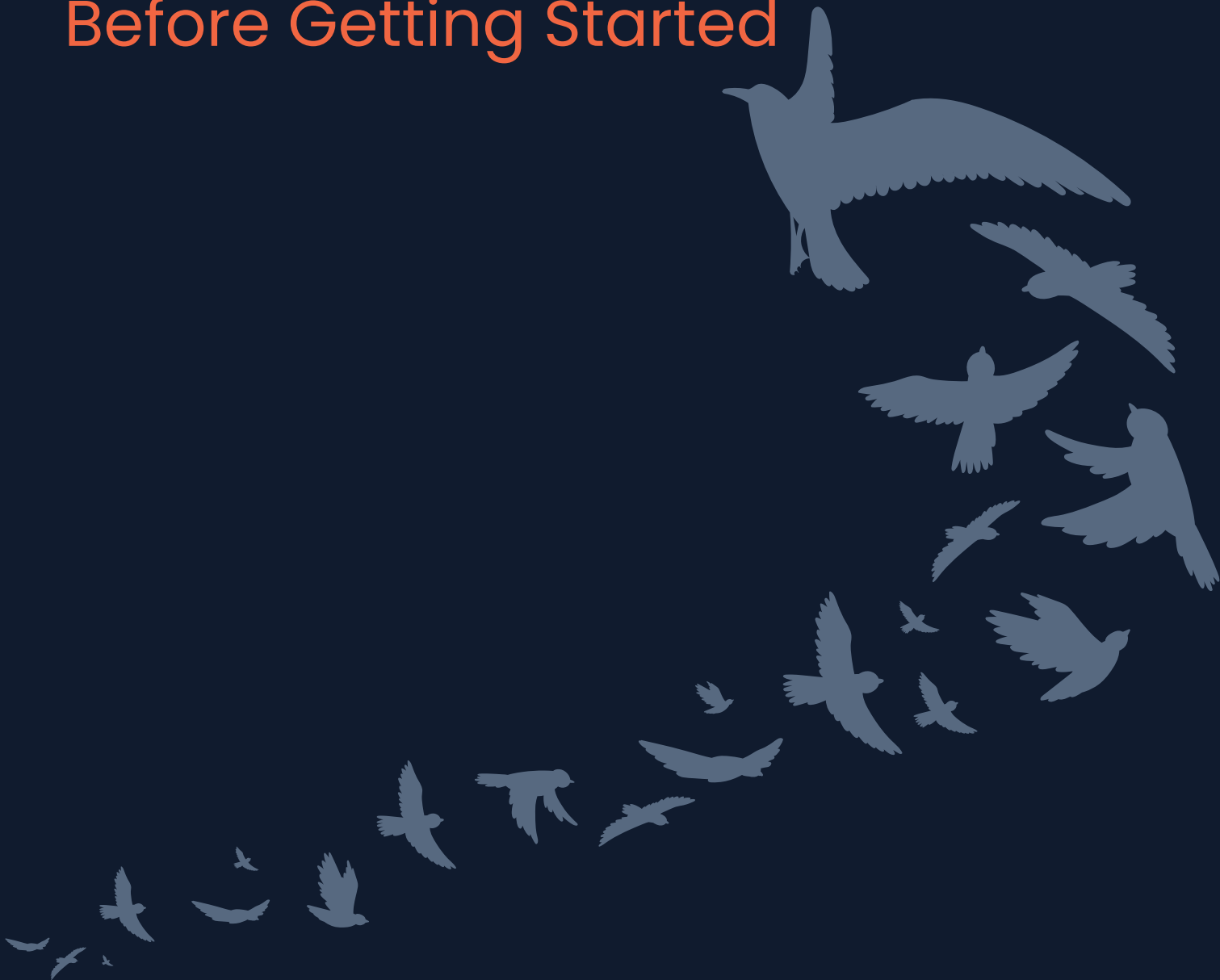




OCTOPAI

Migrating from a Legacy to Modern Database:

The 8 Things You Must Know
Before Getting Started





The Arctic tern is a remarkable species of bird, known for its long-distance migrations. It travels from Greenland to Antarctica and back every year—some 44,000 miles in total annually. It makes your garden-variety database migration look like a walk in the park, right? Nah. All the Arctic tern has to do is fly from one of Earth's poles to the other—difficult, for sure, but not complicated.

But migrating from a legacy database to a modern one, such as the transition from SQL Server to Snowflake, or from Oracle to Amazon Redshift? Now that's both difficult and complicated, and it usually involves an extraordinary number of tedious, manual tasks just to lay the groundwork for what you hope is a smooth transition.

What is "Migration" Anyway?

In IT, the term "migration" refers to any transition from one system, platform, standard, or physical location to another. Server hardware can migrate from one data center to another, or the application hosted on a physical server can migrate to a virtual server or to the cloud. Program code can migrate from one version of .NET Framework to a more recent version. A company can migrate from one ERP system to another.

Because migrations are almost always painful, complex, time-consuming, high-risk projects, no one ever migrates anything just for fun. Usually, there is a circumstance that forces a migration (as when migrating an application from a deprecated operating system to a currently supported one) or a compelling business need.

In this eBook, "migration" refers to moving from an older database to one that is more modern, capable, responsive, and easy to use. Many enterprises are embarking on this journey because even though it's a huge project, the expected return on investment is too good to pass up.

Database migrations are no less complicated than any other kind, but there are actions you can take to make the transition relatively painless.

Here are 8 crucial things you can (must!) do to make your migration smoother:

1 Discard



Have you ever completed a physical move of your home or office and wondered, “I haven’t even looked at half of this stuff in years. Why did I pay good money to move a bunch of useless junk?”

It’s much the same with a database migration: Don’t migrate anything you don’t need. One of the first steps in a database migration is to take stock of your existing database tables and other objects. Chances are good that some—perhaps many—of your objects have one or more of the following characteristics:

- It mostly or completely duplicates one or more other assets in the database
- It was needed for a one-time or short-term purpose and is no longer relevant
- It is legacy data and no longer relevant
- It describes business processes or products that are no longer relevant

You get the idea. It’s time to pick out the database objects that are still useful and discard those that are unnecessary (after double checking that nothing in the data landscape is dependent on it, of course).

Depending on the size and age of your database, this exercise could take a considerable amount of time. In a large organization with a complex data environment, it’s not always easy to track down who or what is using each object. Some tables may only be used for processes or reports once a year, but that once-yearly use is important and relevant. It may require some detective work, but it will be worth the effort because reproducing irrelevant or redundant assets in the new database is a total waste of time.

2 Simplify



Nineteenth-century author Henry David Thoreau of Walden fame had it right: “Simplify, simplify.” He didn’t have to deal with a database migration—lucky guy—but his words are still sage advice for a database migration project.

Modern database systems, like Snowflake or Redshift, use pay-per-use models. Depending on the platform and the plan, you might be paying for the number of bytes scanned when you run queries, or for computing time, or for the data you store.

Storing redundant data? With most modern databases, you’re going to pay more than you should. Need lots of exploration queries because you’re not sure where to find the data you need? Again, you’re going to pay more than you should with most modern databases.

In short, the more complex, complicated or convoluted your data environment is, the more money you will plunk into the coffers of the database providers.

If, on the other hand, you can eliminate redundancies and simplify dependencies before the move, you’ll save money both during the migration and ever after.

Simplifying effectively is difficult to do without a good way to visualize the environment. Complicated data environments are usually tricky to visualize; the “big picture” soon resembles a bowl of spaghetti.

3 Clean up



Modern database platforms make it a point to eliminate data silos by consolidating all your data sources into a cloud-based data warehouse or data lake. Sounds great... except what if your data lake is polluted? Now every user and every report in your company is dipping in the dirty data water! Ick.

What's the solution? Cleaning up the data sources before they enter the lake!

A database migration is the ideal time to look into the sources and processes for your data, especially mission critical data. Good database migration tools will enable you to do so using the techniques of root cause analysis and impact analysis.

Root cause analysis: A good database migration tool enables you to determine which data sources and ETL processes are loading a particular table or report. Checking or confirming the dependencies before you migrate them to the new database will ensure clean, accurate, reliable data going forward.

Impact analysis: This tool shows you the paths from source systems, through any ETL processes, into database tables, all the way to the reports that depend on the data. You don't want to end up tossing a database object into the trash only to find out (post-migration, of course) that it was a key contributor to the annual company profitability report for the CEO.

4 Understand



Even once you know **where** an asset is used, understanding **what** that asset represents is critical to making a decision regarding if and how to migrate it. For example, once you have a clear understanding of what a given data asset represents, you may realize it is a duplicate of a different asset. In that case, it should be replaced instead of migrated, eliminating unnecessary migration and maintenance of redundancies.

Additionally, while impact analysis might expose some data assets as obviously unused and irrelevant, for others you may need to do a little digging to find out who uses them, what they're used for, who is responsible for these assets and what dependencies exist between them and other assets.

If there are certain critical dependencies on an asset, but your information discovery shows that the asset is not of the desired quality level, you might want to identify other assets that could replace it when you migrate.

5 Compare



There are many, many moving parts involved with migrating a database. When you're working with gigabytes, terrabytes or petabytes, it's easy to accidentally leave a few kilobytes out of the migration, or to include something you didn't intend to.

How can you verify that everything you intended to migrate—and only what you intended to migrate—actually made it across? You need a comparison tool—one that represents the two environments graphically and not a spreadsheet file that requires you to eyeball object names row by row. It doesn't take long before all those object names start to look alike. And with the kind of complex, multilayered dependencies which often exist in database systems, attempting to eyeball it would probably give you motion sickness.

6 Improve



As arduous and time-consuming as a database migration project can be, it affords a rare opportunity to improve not only the database system but the underlying data environment.

Most data systems that modern businesses want to migrate were built many years ago and tinkered with ever since. Lots of fingerprints have been left on them by people who didn't always know what they were doing. They may have known what they wanted to accomplish but not the "right" way to do it or at least not the correct way with the tools available now and/or the new database technology you are migrating to.

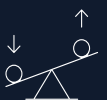
An important part of any database migration project is to use the opportunity to clean up the mess, getting rid of poorly written (and poorly documented) code and kludgy workarounds.

7 Adjust



As different systems commonly use different methods and require adjustments to existing code, syntax supported by your legacy database may need to be adjusted not only in the different database components that hold code (such as Views, Stored Procedures and Functions) but also in any part of the data landscape that runs code against these databases.

8 Leverage



Additionally, since you will anyway be touching and assigning work to be done on many data assets, and it is likely that you will discover a lot of information about each, centralizing those insights will leverage that information for all stakeholders during the migration and for further use post-migration. It's an ideal opportunity to achieve data literacy, adopt modern best practices, enforce some consistency, and leave an easy, logical, working environment for the next crew that comes along.

After all, this whole database migration is (assumedly) to create a more effective and efficient data environment. Leveraging the deeper understanding you now have of your data to establish a common, single source of truth is a giant step in that direction.



The Key to a Successful Migration: A Data Intelligence Platform

You might be thinking to yourself at this point, "Great tips, but how the heck am I going to be able to do all that? I can't even find the source of a reporting error without spending hours digging around my systems."


Well, a data intelligence platform like Octopai, providing automated data lineage, data discovery and an automated data catalog, is key to be able to do all eight of these things properly and effectively ahead of migrating systems.

Want to simplify the data dependencies you need and discard the ones you don't prior to migrating? Octopai's automated data lineage, including comprehensive column-to-column lineage, traces the path of your data objects both back to their source and forward to reports or other targets. This comprehensive visual mapping of the objects, assets, tables and processes across your data systems enables you to clearly identify unused objects, unnecessary duplicates and unnecessarily complicated dependencies. Octopai's automated data catalog enables you to dive deep into any and every asset and discover who uses it and for what, and how valuable it really is (or is not) for your organization. Once you identify what you want to discard or simplify, you can

use Octopai's data lineage visualization tools to check the projected impact of the planned change. Clear information and direction translate into a fast, incident-free move.

Want to make sure all your processes and scripts will work when transferred to the new database? Differences in syntax between the old and new database types can break your scripts and cause data havoc. Octopai's data discovery supports quick discovery of all processes and scripts that need to be altered or replaced.

Want to avoid migration mistakes and do-overs? Octopai's data intelligence platform puts the map in your hands before you even start driving, enabling you to get to your destination faster and without wrong turns. For the assets that will be transferred, Octopai helps your team easily see the relationships between different data sources and the dependencies that need to be created between the new objects. In addition, accurate data many times relies on the upstream processes running in the correct sequence. Octopai provides a clear visual breakdown of the data path, enabling you to analyze and set the timing accordingly.



Want to avoid headaches and confusion when part of your data is in the old system and part is in the new system? Just because you're migrating doesn't mean the day-to-day work stops. In fact, it gets harder, because knowing where to locate what and when is enough to give the most dedicated data analyst a migraine. Octopai's end-to-end data lineage, from the legacy/operational system to reporting, simplifies insight into and through your database along with all other layers of the company's data ecosystem. Put on your Octopai glasses, and your migraine will fade as your vision clears.

Need to verify that everything you wanted to migrate made it across? Octopai's data lineage mapping enables you to check pre-migration, during migration, and post-migration that everything is going to the right place. Octopai provides views of both the old and new environments in one screen (updated automatically as the migration

progresses!) so you can do an apples-to-apples comparison and learn whether the environment you wanted at the far end of the migration is the one you got.

Wish to facilitate your users' adjustment to the new database? Octopai's automated data catalog, which can be built and populated as part of the migration, can help users locate the data assets in their new location and improve the usability of and trust in the data.

Looking for an opportunity to optimize your data environment? Leverage the work you're putting in to understanding your data by centralizing all your information using Octopai's automated data catalog. The establishment of a single, central, automatically-updating source of truth can drive effective data use forward like nothing else. Optimizing data flow and usage is, after all, the ultimate purpose of a data migration - so make the most of it!

Those lucky old Arctic terns just need their instincts to successfully migrate. You need intelligence. Data intelligence. With Octopai's Data Intelligence Platform, you'll be able to migrate your database wisely, effectively, and in a manner that will deliver cost-effective data-driven insights for years to come.