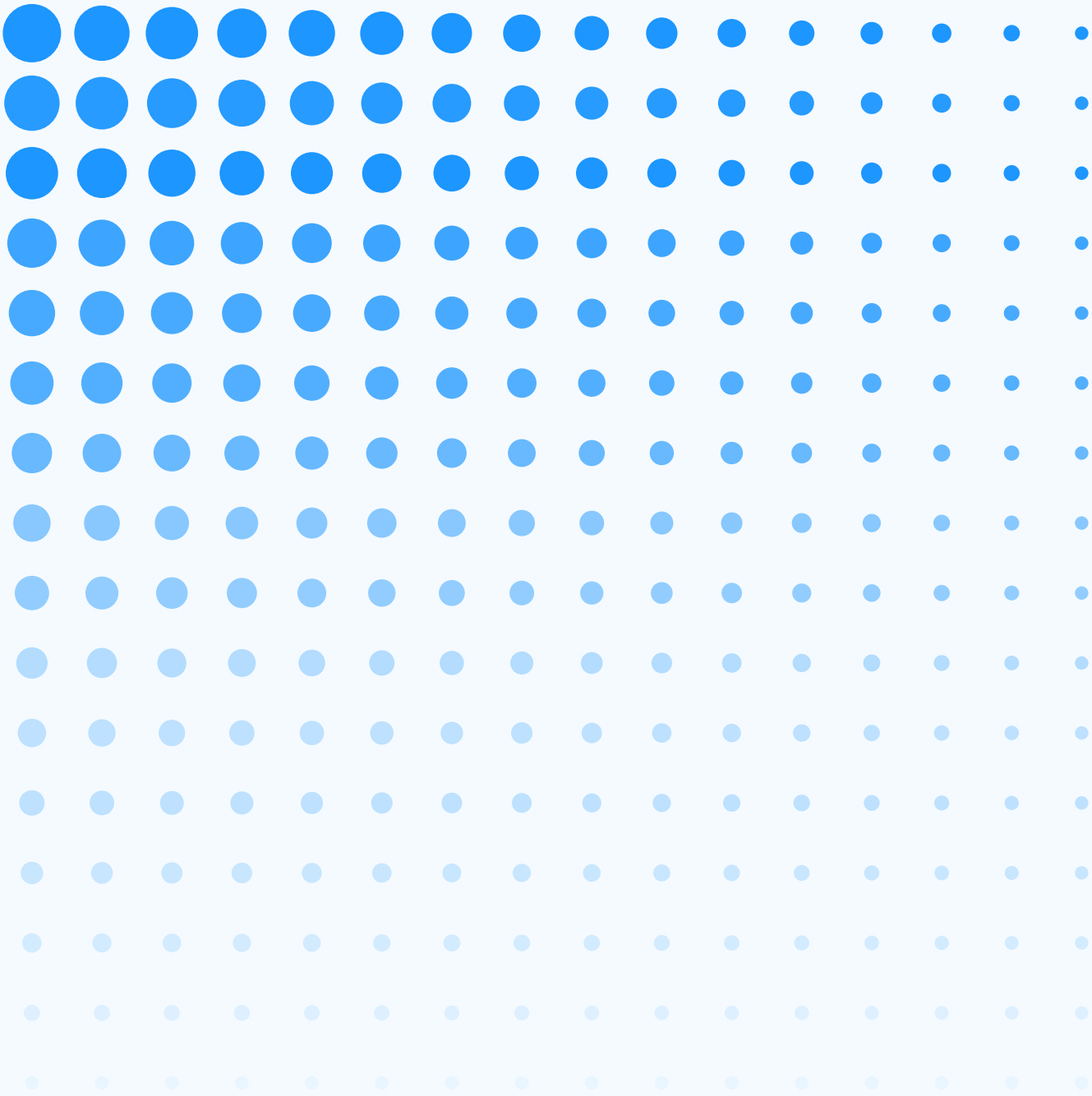
Five Data Collection Methods for Analytics



integrate.io

Table Of Contents

01

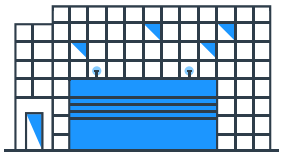
OVERVIEW

Is the data warehouse dead? Are data lakes the next big thing? Or is streaming where it's at? The answer, of course, is: it depends.

The answer, of course, is: it depends. It depends on your needs, budget, and what you plan to do with the data. Choosing the right method to collect data is important, so this whitepaper will present the various methods for collecting data for analytics, and it will help you to understand how they work and identify which one is right for your Data needs.

02

DATA WAREHOUSE



What is a Data Warehouse? A data warehouse is the traditional way of gathering data for analytics.

It is a logical entity that can reside on one physical relational database management system (RDBMS), or span across a number of physical servers.

Data contained in the data warehouse is already prepared and ready for querying. How? The data in the data warehouse is modeled to optimize speci c queries and answer speci c questions. To turn raw data into prepared data in the data warehouse, it is put through an ETL (extract, transform, load) process. This extracts the data from various sources, transforms it for querying (e.g. calculating total sales per region) and then loads the transformed data into the data warehouse. The data warehouse is then queried to answer prede ned questions or ad hoc queries, e.g., what are the total sales per region? Nonetheless, since the schema and the data are pre-built, you cannot ask questions that refer to, for example, data that does not exist.

Because data warehouses are well designed and highly organized, they can provide very accurate results. Schema's are notoriously painstaking to change, especially if the data sources or questions or any other factors change on a daily or weekly basis.

When should I use a data warehouse?

- The data sources are relatively constant.
- The questions that need to be asked of the data are mostly known in advance.
- The data structure is relatively static.
- Mistakes and inaccuracies cannot be tolerated, e.g. for accounting.
- Tight access control and security are a must.

How can I implement a data warehouse storage layer?

- RDBMS: MySQL, Oracle, SQL Server, etc.
- Columnar databases: Vertica, ParAccel, Amazon Redshift, Google BigQuery

03

DATA LAKE

What is a data lake? “Data lake” is a relatively new term that is credited to James Dixon, Pentaho’s CTO. Essentially, a data lake is a big repository of raw or slightly prepared data.

While data warehouses store data in a well-organized and hierarchical format, data lakes store data in a at structure, usually as les. Data in the lakes is associated with a unique ID and tagged with metadata so that the relevant data can be found and queried once a new question is asked.

Data lakes do not require much planning—typically, there is no schema or an ETL process in place. Thanks to the low cost of data storage both on premise and in the cloud, and the abundance of virtual instances, a data lake can be set up quickly.

Even before anyone knows what questions they want to ask, data can be poured into the lake from different sources and in several formats right away. Better yet: while a data warehouse has very strict access restrictions and security that keep the data in the hands of the few, data lakes can be open to more people in the organization in read-only format, thus putting the data in the hands of the many.

But there is a catch. Since data lakes contain a great variety of data formats and huge volumes of data, querying data from data lakes is much more diffcult. They will not work with most of your standard BI tools (though some are starting to support data lakes), and require coding if you want to generate any insights from the data. Thus, a data lake is a playground for people with advanced data skills— the data scientists. Here they really get to shine and show o their perfect backstrokes.

Luckily, a data lake does not need to stand on its own in the wild. Its data can be organized by running it through an ETL process, and then storing the output back in the lake or in a data warehouse—a win–win for data scientists and others alike.

When should I use a data lake?

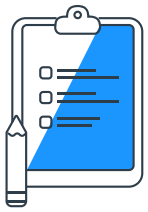
- The data needs to be collected immediately. There is no time for planning.
- There is a small budget for data storage.
- The data sources and formats are highly dynamic.
- The questions are not known in advance, change frequently, or need to be asked ad hoc.
- Data scientists need a playground to and develop new insights.
- More people in the organization require accessto the data.

Which tools can I use to implement a data lake?

- Hadoop Distributed File System (HDFS)
- Amazon Simple Storage Service (S3)
- Possibly combined with a data warehouse, HBase or a NoSQL database like MongoDB

04

BIG EVENT LOG



What is a big event log? Big event logs (BELs) are exactly what they say they are: a big log of events.

They take a middle path between data warehouse strictness and data lake looseness by storing data as events in one location, though in a semi-structured format. Each event has at least two common properties: date-time and event type—other properties can be set per type of event (e.g. “item name” for the “added to cart” event). They can be stored as JSON objects, or in a relational database with sparse tables (meaning that most of the columns will have null values).

There are a few limitations. First of all, a BEL only logs events (e.g. the user added a T-shirt to the shopping cart), not inventories (e.g. T-shirt prices). Also, the time granularity needs to be consistent across the entire log. If some of the events are logged in days and others in seconds, the former events will need to be logged in seconds as well.

Otherwise, BELs have a few advantages. While other data collection methods require major changes to the data architecture, BELs can t in without many changes—all they require is for the application to log events. Because all the data is available in one place, another advantage is that a BEL allows you to run wide queries easily without having to join to any other data sources.

When should I use a big event log?

- The data can be described as a series of events.
- It is too hard to change the existing data architecture, but easy to log events.
- Wide queries are required.

How can I implement a big event log?

- NoSQL databases: e.g. MongoDB
- Search engines: e.g. Elasticsearch
- RDBMS with sparse tables
- Data lakes

05

STREAM PROCESSING

What is stream processing?

At the moment, streaming is the thing—everybody wants it, even if they do not know why. The bottom line is that stream processing can keep you to keep up to date on what is happening right now, for example, how many people are currently reading your new blog post, whether someone just liked your latest Facebook status, etc.

For most use cases, stream processing provides nice-to-have features. However, sometimes it is a must. It allows you to perform analysis in near real time and to act on it quickly. Let us say that you run a big ad agency. Stream processing can keep you posted on whether your latest online ad campaign— which your client paid a lot of money for—is actually working, and if not, you can make immediate changes before the budget gets washed down the drain. Another use case is providing real-time analytics for your own app—this looks good and your users may need it.

Stream processing is implemented by frameworks that can handle streaming data. Most likely you will use stream processing side by side with your batch processing and other methods (also see the next section).

When should I use stream processing?

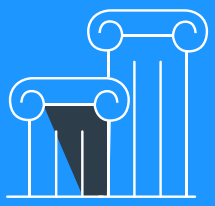
- The data needs to be known in real time for news portals, stock brokers, ad agencies, etc.
- Users need real-time data about your app.

How can I implement stream processing?

- Streaming frameworks: Apache Spark, Apache Storm, Apache Samza, Amazon Kinesis, etc.

06

LAMBDA ARCHITECTURE



What is Lambda Architecture (LA)? Lambda Architecture takes a hybrid approach that lets you enjoy all of the data collection methods.

Basically, it is a data hub that processes events,like big event logs. However, LA lets other services subscribe to the events, and sends them the events as they come in.

Therefore, you can have your data warehouse, data lakes, big event log, real-time system analytics,and other future services subscribe to lambda's events, and handle them accordingly. Lambda also allows you to have machine learning and prediction engines subscribe to the data for deeper forms of analytics.

When should I use Lambda Architecture?

- Several services need to handle the data.
- If you need something that runs alongside other data collectors.

How can I implement LA?

- Message publisher/subscriber with Rabbit, Kinesis or Kafka
- Storm or Spark for stream processing
- Hadoop MapReduce, Spark or other platforms for batch processing
- HDFS, data lakes, data warehouse for data storage

07

CONCLUSION

There are quite a few ways to collect data.

So before you do anything else, sit down, and gather some data about your data, data architecture and your organization's requirements. Depending on the use case, you may end up using one or more methods:

Data Warehouse

- Constant data sources and structure
- Questions are known in advance
- Little tolerance for inaccuracies
- Tight security

Data Lakes

- Immediate data collection
- Low budget
- Dynamic data sources, formats and questions
- Loose security, read-only access for the entire organization
- Playground for data scientists

Big Event Log

- Data can be described as events • Fits existing data architectures
- Wide queries
- Real-time data is required for you or your users

Lambda Architecture

- Hybrid data collection
- Runs alongside other methods