Information Services & Use 42 (2022) 397–407
DOI 10.3233/ISU-220176
IOS Press

An introduction to data lakes for academic librarians

Clifford B. Anderson*
Chief Digital Strategist, Vanderbilt University Libraries, 419 21st Avenue South, Nashville, TN, USA

Abstract. Data lakes offer an emerging option for librarians who are seeking to develop data repositories for institutions of higher education. Data lakes provide flexible, secure environments for students and faculty to compute with licensed data. This paper presents a high-level overview of the data lake architecture, highlighting its differences from traditional data warehousing solutions. The paper also touches on the staff expertise required to create successful data lakes in academic libraries.

Keywords: Cloud computing, cluster computing, data governance, data lake, data repository, data swamp, data warehouse, metadata, staff development

1. Introduction

This paper introduces the concept of a data lake as a platform for delivering big, heterogenous datasets to library patrons. A data lake gathers datasets across an organization into a single location, allowing latent connections between datasets to be discovered dynamically. When deployed with appropriate metadata and access controls, data lakes become effective ways for libraries to utilize the text and data mining rights they have licensed for proprietary datasets.

The formation and maintenance of data lakes represent a natural evolution in the responsibilities of data-centric librarians. In a previous paper, Hilary Craiglow and I described the process of licensing, acquiring, and providing licensed data to researchers [1]. That paper presupposed that librarians would deliver datasets to patrons, allowing them to load those data into their computational environments for analysis. While advising patrons about licensing restrictions associated with the datasets, librarians have no way to maintain oversight of downstream uses of data under this scenario. By building a data lake and providing researchers with datasets in that context, librarians can quickly afford researchers access to information while ensuring compliance with licensing terms and other potential restrictions.

This paper aims to offer a high-level overview of the data lake architecture and its benefits for academic librarians and, by extension, faculty, and students at their institutions. I will introduce critical technologies for storing, transforming, and analyzing data at this scale. The pace of technological change is swift, and new tools and methods for creating and fostering data lakes are emerging regularly. My goal is not to survey the field, but to offer a practitioner's perspective. In what follows, I draw on experience working with colleagues in the library, information technology, and the faculty to explore, implement, and develop scholarly projects using a data lake architecture.

*E-mail: clifford.anderson@vanderbilt.edu.

2. What is a data lake?

What is a data lake? The concept of a data lake emerged as a consequence of the dizzying growth of data that characterizes the era of big data. As the volume of data increased across enterprises, data engineers concluded that trusted methods of storing and providing access to that data, such as data warehouses, were failing to keep pace with the accelerating velocity of data [2]. In 2010, James Dixon, chief technology officer at Pentaho Corporation, invented the term "data lake" in the context of a brief blog post [3].

> If you think of a datamart as a store of bottled water – cleansed and packaged and structured for easy consumption – the data lake is a large body of water in a more natural state. The contents of the data lake stream in from a source to fill the lake, and various users of the lake can come to examine, dive in, or take samples [4].

Dixon's metaphor took off among information technologists because it captured the essence of a changing paradigm for managing increasing quantities of data across the enterprise. Rather than investing costly effort upfront to reshape data so it conforms with a predefined schema, the data lake paradigm imposes few requirements on data during the ingestion process. By eliminating bottlenecks (or, to go with the bottled water metaphor, the bottling process!), a data lake grows more quickly and may contain a greater variety of datasets.

The distinction between a data warehouse and a data lake comes down to a difference in curatorial practice. In a data warehouse environment, datasets must be curated before being imported to accord with the existing schemas [5]. This emphasizes the so-called "extract, transform, and load" or ETL process, which maps data from its original form to the normalized form required by the data warehouse [6]. A data lake provides an alternative to the data warehouse, reducing or eliminating the need to stage data before making it available to patrons. This alternative procedure is called "extract, load, and transform" or ELT [7].

What is the difference between following an extract, transform, and load (ETL) and extract, load, and transform (ELT) routine [8]? In a classical data warehouse environment, the goal of the ETL process is to take incoming datasets, reformat them to match the existing data structure in the data warehouse, and then import them into that environment [9]. By transforming incoming data before adding them to an existing data repository, the data warehouse remains uniform, predictable, and accessible to data analysts. However, as data arrives from multiple sources across an organization, taking time to carry out this ETL process saps resources. Data may wind up parked in holding zones, making it unavailable for real-time analytics. By contrast, the ELT process ingests data in its original format, resulting in a heterogeneous data environment. If a data analyst needs to compare data across datasets, transforming the data into the required format for that comparison takes place within the data repository. By allowing data to flow readily into a data lake, data engineers provide users across their organization access to data more quickly.

These respective procedures differ in their relative advantages [10]. The ETL process focuses on data uniformity at the cost of speed of ingestion. The ELT method, by contrast, optimizes ingestion speed, but imposes higher costs on analysts, who must transform datasets when conducting comparisons. Another difference has to do with infrastructure and tooling. With an ETL approach, data is shifted from its source to a temporary location, where it is cleaned, reformatted, and prepared for ingestion into the data repository. In an ELT process, the transformation of the data into a usable format takes place inside the data lake itself [11].

The shift from ETL to ELT is not a panacea. After all, investing effort in the transformation stage aims to make data discoverable and tractable for end users. As we shall see below, ETL and ELT processes

both require a transformation step. As the concept of data lakes has evolved over the past decade, data engineers have come to recognize that data lakes require maintenance and oversight to serve valuable ends. While Dixon's metaphor suggests leaving datasets in their original state, dropping the transformation step inhibits the practical utility of those data.

3. Why libraries?

Why should librarians consider building data lakes? A short answer is that libraries face similar challenges to data engineers working in the commercial sector. Their patrons want to engage in data-intensive computational research and need access both to the data and to an open-ended environment for computing with that data. Datasets are proliferating in libraries, but librarians frequently lack tools to manage data at scale. For several decades, librarians have assisted researchers with preserving research data, consulting about metadata description, advising about licensing, and recommending relevant data repositories. Acquisition librarians routinely license databases and text and data mining (TDM) rights to the data in those databases.

In "Time to Adopt: Librarians' New Skills and Competency Profiles", Schmidt et al. remark, "One area which is evolving very fast is text and data mining (TDM), and libraries might already have a range of subscriptions and collections which come with appropriate licenses, but have not yet stepped up to provide practical support for researchers to exploit these riches [12]". In my view, librarians have mostly avoided developing robust support models because of the specialized expertise and time commitment necessary to provision, compute, and analyze datasets for individual scholars, research teams, and labs. Nevertheless, there are reasons to think that the advent of data lake architecture has changed the equation.

A first reason for libraries to consider providing an environment for big data analytics is that researchers may not have the technical skills. As interest grows in data-intensive research among faculty, librarians find themselves conducting a data-centric version of the reference interview, assessing patrons' skills at computing with data [13]. Faculty interest in data-intensive computing may outstrip their technical facility, leaving library staff members to fill the gaps.

A second reason is that librarians share the fiduciary responsibility to uphold the licensing agreements that their institutions signed to acquire datasets. As noted above, librarians must communicate licensing terms to patrons and, with colleagues in information technology, may require patrons to become parties to data use agreements.

A third reason for librarians to take on the challenge of managing a data lake for their organization arises from their research expertise. Given their understanding of researchers' expectations and requirements, librarians can create data lakes that supply data in a useful form. As noted above, a data lake containing only raw information without any processing or transformation will inevitably turn into what practitioners term a "data swamp [14]".

4. Architecture

How are data lakes implemented from a technical perspective? A data lake emerges from the orchestration of multiple layers of technology, including storage platforms, cluster computing, and analytical tools. The composition of these layers differs from implementation to implementation. In this high-level introduction, we only touch on essential components.

5. Storage

Let us start with storage within data lakes. Since a data lake contains different kinds of datasets from heterogeneous sources, the storage layer for a data lake needs to be flexible. A data lake stores structured, semi-structured, and unstructured data [15].

While the architectural concept of a data lake should be kept distinct from its instantiation, Spark has emerged as central for creating data lakes. Researchers at the University of California at Berkeley introduced Spark more than a decade ago as a transformational improvement to the Hadoop ecosystem for computing with big data [16]. Spark's primary innovation was "keeping data in memory", obviating the need to read data from disk during every operation [17]. In 2013, the Apache Software Foundation took over the stewardship of Spark, releasing it under an open-source license, while Matei Zaharia, Ion Stoica, and other innovators of Spark at Berkeley started Databricks to provide commercial support for Spark and its growing ecosystem [18]. In recent years, Spark has emerged as a *de facto* standard for analyzing big data in fields as diverse as bioinformatics and political science.

How do users interact with data within a data lake? Assuming that the data lake builds on Spark, the primary data type is a so-called dataframe. Similar to its equivalents in Python when using pandas [19] or R when applying dplyr [20], a dataframe in Spark represents data in a tabular format with rows and columns. The dataframe in Spark sits on top of lower-level constructs (such as "resilient distributed data" or RDD) [21], which Spark conveniently abstracts away from users in most scenarios. Spark computes data in dataframes efficiently, allowing high-performance queries of datasets with millions of rows.

How is this data serialized? That is, how do these systems store data when not in use? These systems generally do not store data in common formats such as CSV, JSON, plain text, or XML. While data frequently arrives in data lakes in one of these formats, processing a CSV with millions of rows or computing against folders with hundreds of thousands of JSON files proves error-prone and inefficient. New formats have emerged to meet the challenges of scale. In the case of Spark, the two principal standards for storing data are Apache Avro [22] and Apache Parquet [23]. The difference between Avro and Parquet comes down to row-oriented versus column-oriented storage: the first performs better for adding streaming data, while the second optimizes lookup speed for large datasets.

6. Metadata

How do we keep track of datasets in data lakes? As datasets multiply in the data lake, cataloging them and keeping track of their contents becomes increasingly important. To this end, data lakes incorporate data catalogs. The data catalog provides information about datasets, such as their schema, data types, storage format, and last modified date, among other details. The default storage format for most data lakes is the Hive Metastore. Apache Hive [24] is a framework developed for Hadoop to query datasets with SQL. Among the framework's components is a metadata repository called the "Metastore". As storage systems migrated from the Hadoop File System (HDFS) to object stores like AWS S3, the Hive Metastore remained the standard for maintaining metadata about the datasets [25]. By querying the Hive Metastore, users can explore the contents of data lakes without delving into the underlying object storage.

While many data lakes continue to use the Hive Metastore for metadata, newer metadata stores are emerging that offer greater functionality. For instance, Amazon Web Services (AWS) provides an alternative metadata store as part of AWS Glue [26] and Databricks has recently introduced its Unity Catalog [27]. These next-generation metadata stores provide extra functionality, such as simplified administration and greater flexibility with user access controls and permissions. These systems remain essentially similar to their predecessors, supplying indexes to the datasets in the data lake.

7. Cluster computing

A motivating rationale for creating the ecosystem of big data technologies is the necessity of moving beyond a single computer to store and analyze datasets. This is a relative measure for big data because the capacity of computers varies. A researcher may have a laptop with a 500 GB hard disk drive (HDD) and eight GB of random-access memory (RAM). For this researcher, any dataset that exceeds that laptop's capacity is "big data". A data science center, by contrast, may have access to systems such as NVIDIA's DGX A100, which provides up to 320 GB graphic processing unit (GPU) memory, 512 GB system memory, and almost 8 terabytes of storage [28]. In institutions that do not have access to high-end equipment such as the A100, cloud providers also offer fractional access to high-performance systems. In the AWS cloud, for instance, you can select memory or storage-optimized systems using Elastic Compute (EC2) with 512 GB of system memory and nearly unlimited storage capacity. These powerful systems would readily compute data that librarians might reasonably consider as "big".

So why do researchers turn to cluster computing rather than these high-performance systems? Cost is a primary driver. The price of an NVIDIA DGX A100 falls in the six-figure range. Provisioning a top-of-the-line system on AWS also comes at a high price. Building a cluster is inexpensive, by contrast. At an individual level, the specifications of the computers that compose a cluster may be modest. The "nodes" in a cluster may only have 8 GB of memory. A cluster unites these inexpensive systems, distributing computational processes across its nodes. By scaling out rather than up, clusters achieve remarkable performance and, in many cases, make possible the analysis of datasets that transcend the capacity of even the most cutting-edge standalone systems.

8. Cloud computing

Cloud computing democratized access to cluster computing. In the recent past, researchers typically have needed access to high-performance computing centers to distribute computations over a cluster. High-performance computing centers use open-source tools such as SLURM (Simple Linux Utility for Resource Management) to schedule tasks and manage operations on their clusters [29]. Setting up and maintaining these centers requires specialized expertise and financial resources that smaller institutions lack. The advent of cloud computing has made cluster computing accessible to organizations of all sizes.

Cloud providers make it straightforward to *scale* clusters. Researchers select from a range of prebuilt machine images or customize their own and then instantiate a cluster with as many nodes as needed to efficiently carry out their computation. Of course, creating clusters with dozens or hundreds of nodes generates significant expense. Frameworks such as Databricks allow you to increase and decrease the size of your cluster dynamically while also permitting you to set limits to the size of your cluster, protecting you from paying for unwanted or idle resources. They also allow you to shut down clusters after a specified period of inactivity.

The cloud providers also provide robust *security* mechanisms to safeguard data on clusters. On some campuses, access to cloud computing resources will be managed centrally through tools such as single sign-on (SSO), while security protocols such as two-factor authentication may be enforced for access. While these mechanisms ensure that users of the data lake belong to the organization, administrators must take responsibility for authorizing access to the data. In a fashion similar to databases, they can build role-based security groups with these frameworks. They may also audit access to data, storing access logs for defined lengths of time.

A final advantage of deploying big data environments such as Spark on the cloud is the innovation dividend. The pace of progress in data engineering is relentless, and keeping track of the latest tools for managing these environments proves challenging. Major cloud providers, including Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Service (GCS), provide auxiliary services that support data lakes, from storage services to analytical tools. Cloud providers also offer products to host data lakes themselves, such as AWS Lake Formation [30], Azure Data Lake [31] or Google BigLake [32].

Of course, not everything is rosy in the cloud. Documentation is sometimes uneven or out-of-date. Setting up roles and permissions is tricky, and mistakes can compromise the security of cloud-based projects. The cost of projects may also be difficult to predict in advance.

9. Data curation

What skills do data librarians need to provision big data environments using Spark? The scenario will differ between platforms, and what follows is informed by our experience setting up AWS EMR and Databricks at the Vanderbilt University library. While implementation details will differ when using other cloud environments and platforms, the essential skill sets remain *mutatis mutandis* the same.

An obvious, but crucial, observation is that provisioning data lakes is inherently an interdisciplinary, cross-functional activity. Setting up, supplying, transforming, and providing access to data requires different skills. While exceptional staff members may develop functional expertise every phase, the more likely scenario is a differentiation of these roles among librarians and information technologists. In publications about data lakes, authors differentiate the contributions of various functional specialists [33]. The enumeration below reflects, with some idealization, our division of labor at the Vanderbilt University Libraries.

- *Data acquisition and licensing librarians* To get started with data analysis, you need to acquire relevant datasets. In our case, the university e-resources team negotiated the licenses of the datasets that we planned to incorporate into our data lake. In fact, the library's e-resources specialists had already been contracting text and data mining rights when negotiating the purchase of new databases. In the process of setting up a data lake, librarians from digital scholarship began to work more closely with the e-resources team to collaborate on licensing, acquiring, and "flowing" new datasets into the data lake.
- A *cloud engineer* understands how to work within one or more cloud platforms. This function resembles a dev/ops specialist or systems analyst, but with a focus on cloud technologies rather than on-premises solutions. In our context, the primary responsibility for this role fell to a unit called "Cloud Services" within Vanderbilt University Information Technology. Librarians from digital scholarship collaborated with colleagues in the unit to set up AWS EMR and Databricks. When facing challenges provisioning these cloud environments, we drew on experts at both Amazon and Databricks.
- A *data engineer* develops the pipelines that connect data sources to the computational environments where data will be analyzed. A data engineer commonly performs extract, transform, and load (ETL) or extract, load, and transform (ELT) operations.
- A *data librarian* works with the cloud engineer and the data engineer to maintain the data lake, providing metadata for datasets, updating the data catalog, authorizing user accounts, and developing starter notebooks and SQL dashboards to showcase its contents. The role of the data librarian would be roughly equivalent to the function of the data analyst in a corporate setting. In specialized scenarios, functional specialists such as a *GIS librarian* would assist with developing different representations of the data (note: GIS stands for Geographic Information Systems).

- *Liaison librarians* promote the data lake for research, teaching, and learning. The liaison librarian would also be familiar with datasets in the data lake that intersect with topics in their fields of coverage. As Kristin Partlo remarks (with undergraduates in view, but in generally-applicable terms), "the data reference interview should take a pedagogical approach, helping new researchers develop the capacity to do it on their own, creating self-sufficient learners and nascent social scientists [34]".

When building a data lake for patrons, librarians may want to split the difference between the ETL and ELT approaches. Since most patrons are not data scientists, it is unreasonable to expect them to transform data from one format to another. By transforming datasets into a common schema before loading them into the repository, librarians can also build SQL dashboards and other analytical tools for exploratory analysis. So, relying on a traditional ETL procedure will make patron-facing data lakes more accessible. That said, this approach does not scale readily to formats such as image collections and audiovisual materials. Coalescing datasets into a shared schema also typically involves data loss; columns of the source dataset may be dropped, trimmed, or joined to match the target schema. Given the capacity of cloud systems for storing data at a relatively low cost, librarians may consider storing two copies of each dataset, the original and the transformed versions.

As noted, the transformation step generally requires a commitment of time and computing resources. In our practice at Vanderbilt, the steps look as follows. Let us assume we start with a source dataset of compressed XML files that we want to convert to Parquet. First, we create an EC2 instance on AWS with the requisite permission to move data between S3 buckets on our AWS account. We then copy the source dataset to the EC2 instance and unzip the files with bash. Next, we apply a Python script that converts the individual XML documents into JSON documents, using XPath and XML Schema to select and modify columns. Finally, we use another bash script to combine the individual JSON files, creating a single JSONL file that we move into the destination S3 bucket. These steps take time to carry out, and with every stage, there is the possibility of error. Ideally, this workflow would be automated, especially when data sources need to be updated regularly.

10. From datasets to data lakes

The shift to big data has been taking place for more than a decade. Libraries have already garnered expertise in licensing and providing access to large-scale datasets for computational analysis. Gone are the days when libraries licensed datasets from vendors, only to leave the data sitting on hard drives in closets or saved to network locations without actionable plans to share them with patrons. At this point, librarians have options for storing licensed datasets, from repositories like DSpace or Fedora Commons to digital object stores such as Amazon S3 or Google Cloud Storage. Simply placing datasets in accessible (but secure) locations represents a step forward in library data management.

The shift to data lakes represents an evolutionary next step, from thinking about datasets individually to considering them collectively. Building data lakes allows librarians to think more holistically about their licensed data. They can ask questions, for example, about the range of geographic coverage, date ranges, languages, and formats. Which regions and temporal periods are covered well? Which are relatively absent?

The next step is to combine datasets from heterogeneous sources. In some cases, researchers may want to explore the connections between licensed datasets and open-source datasets such as Project Gutenberg [35], Wikipedia [36], and WordNet [37]. In other cases, they may seek to uncover linkages between datasets provided by different vendors. A data lake provides an environment for discovering tacit ties between data sources.

11. Data governance

Building a data lake goes beyond adding heterogeneous datasets to a single platform. Data arrive with different terms of use. Vendors who license datasets to libraries expect librarians to uphold these terms. In some cases, vendors reserve the right to audit how libraries store their data and grant patrons access. So, data lakes need to keep track of who is authorized to interact with the data and maintain detailed logs of who has interacted with the data.

These requirements have proved challenging to satisfy. In the absence of a delivery platform that provides robust data governance features, librarians have found themselves caught between vendors' license stipulations and researchers' computational ambitions. In the recent past, when data was routinely delivered to libraries on external hard drives, libraries could loan it out like any other resource. Of course, the first step for most researchers was to copy the data into a research computing environment, exercising the commonsense judgment that libraries acquired data to facilitate its use for research.

In a data lake environment, the facilities for storing and analyzing data are interconnected, obviating most need to transfer data to a different environment for analysis. While a principle of creating robust data lakes is maintaining a separation between storage and compute [38], a data lake makes connecting them straightforward.

The failure to implement a robust data governance plan risks turning a data lake into a data swamp. "A data swamp is a data pond that has grown to the size of a data lake but failed to attract a wide data analyst community, usually because of a lack of self-service and governance facilities [39]". The literature uses the term 'data swamp' ambiguously. For some, a data swamp contains datasets, but fails to make them adequately discoverable. Datasets in a data swamp may become essentially inaccessible due to insufficient metadata [40] or poorly implemented governance [41]. For others, a data swamp emerges as a natural risk of the data lake approach.

> So called "data lakes" embrace the storage of data in its natural form, integrating and organizing in a Pay-as-you-go fashion. While this model defers the upfront cost of integration, the result is that data is unusable for discovery or analysis until it is processed. Thus, data scientists are forced to spend significant time and energy on mundane tasks such as data discovery, cleaning, integration, and management–when this is neglected, "data lakes" become "data swamps [42]".

By common consent, the key to avoiding the creation of 'data swamps' is to invest in robust metadata description. "The single most important principle that prevents a data lakehouse from becoming a swamp is the degree of cataloging done within its layers [43]". The quality of the metadata may be the determining factor in the success of the data lake as a whole, providing a map to navigate between the atolls of data. Librarians can make signal contributions here, though they may find the kinds of cataloging initially unfamiliar. For instance, a number of tools use Machine Learning to populate metadata automatically and to assess its quality. A data catalog may also include details about the types of columns in datasets and the joints between datasets. In the end, the data catalog serves the same purpose as a library catalog, that is, to foster discovery and use of the content of a collection.

12. Cloudy horizons

As demand for access to licensed data grows, libraries are seeing waxing interest among vendors in providing access within their own computational environments. These environments make promises similar to data lakes, allowing users to write code in languages such as Python or R and enabling the

importation of open-source packages and libraries. In most scenarios, users write their code in notebooks and execute it over a subset of data selected from a vendor's catalog. These vendor-supplied systems generally run on analogous cloud infrastructure. While the details of their implementation are not always available, they may serve in practice as a kind of data lake (or, perhaps more accurately speaking, a data warehouse). Of course, a limitation is that they provide access only to the vendor's catalog of datasets. If you want to explore the connections between datasets from different vendors (or open-source data sources), these environments may prove constricting. Another drawback is that they may circumscribe the kinds of computational techniques that you can deploy. While these environments typically allow you to load a range of Python packages, they may not permit you to import models from Machine Learning repositories such as Hugging Face [44], for example.

While data lakes may not solve every data management problem in libraries, they offer a means for providing students and faculty with easy access to datasets in a secure environment with adequate computational tools. Data lakes also promise to reduce effort for librarians, at least over time. While the initial cost of setting up a data lake is high, requiring significant institutional coordination, libraries gain greater efficiency by centralizing datasets in a single environment. As demands for data-driven research grow, they need only focus on computing within the data lake and not on delivering a myriad of datasets in different formats to researchers with dissimilar computing platforms. Ideally, the savings in time can be redirected toward the data catalog, training material, and instructional sessions.

How will data lakes evolve in the academic ecosystem? Will every university library develop its own data lake, or will data lakes become managed by academic consortia? Todd Carpenter, Executive Director of the National Information Standards Organization (NISO), speculated recently about pooling publishers' datasets in a jointly governed data lake.

> Could we envision a world in which researchers didn't have to recreate their own relevant repository of content that they've cobbled together from the collection of resources to which their institution might have TDM [text data mining] access? Imagine, if a trusted third party could be established to provide a resource to which researchers might turn that would provide an analyzable corpus of all scholarly literature, how valuable such a resource might be [45].

Carpenter acknowledges the licensing challenges that collaborating on a common data lake would pose, while also noting the potential security advantages of such an arrangement. Will we see such a shared data lake emerge along the lines that Carpenter imagines? Or will academic data lakes develop in a different direction, bringing together datasets from labs within institutions along with licensed data from third parties?

While clouds may obscure the horizons of data lakes in academic libraries, the need for these large repositories seem established as rivers of structured, semi-structured, and unstructured information continue to flow into libraries at ever-increasing rates.

Acknowledgements

The author gratefully acknowledges financial support from the Databricks University Alliance.

About the Author

Clifford B. Anderson is Director of Digital Research at the Center of Theological Inquiry in Princeton, NJ. He is also Chief Digital Strategist at the Vanderbilt University Library. He holds a secondary

appointment as Professor of Religious Studies in the College of Arts & Science at Vanderbilt University. Among other works, he is co-author of XQuery for Humanists (Texas A&M University Press, 2020) and editor of Digital Humanities and Libraries and Archives in Religious Studies (De Gruyter, 2022). He received a BA in Philosophy from Kenyon College in 1992; a Master of Divinity from Harvard Divinity School in 1995; a Master of Theology from Princeton Theological Seminary in 1996; a Doctor of Philosophy in Systematic Theology in 2005, and a Master of Science in Library and Information Science from Pratt Institute in 2012. Phone: 615-322-6938; E-mail: clifford.anderson@vanderbilt.edu; Web: www.cliffordanderson.net.

References

[1] C.B. Anderson and H.A. Craiglow, Text mining in business libraries, *Journal of Business & Finance Librarianship* **22**(2) (2017), 149–165. doi:10.1080/08963568.2017.1285749, accessed October 12, 2022.
[2] C. Madera and A. Laurent, The next information architecture evolution: The data lake wave. in: *Proceedings of the 8th International Conference on Management of Digital EcoSystems*MEDES. Association for Computing Machinery, New York, NY, 2016, pp. 174–180. doi:10.1145/3012071.3012077, accessed October 12, 2022.
[3] A. Gorelik, *What Is a Data Lake?*. O'Reilly Media, Sebastopol, California, 2020.
[4] J. Dixon, Pentaho, hadoop, and data lakes, *James Dixon's Blog*, 2010. https://jamesdixon.wordpress.com/2010/10/14/pentaho-hadoop-and-data-lakes/, accessed October 12, 2022.
[5] Or alternatively the administrator of the data warehouse must evolve the data schemas to match the requirements of incoming data.
[6] R. Gopalan, *The Cloud Data Lake*Early Release eBook. O'Reilly Media, Sebastopol, CA, 2022, 3.2.1.
[7] Ibid.
[8] See https://www.ibm.com/cloud/blog/elt-vs-etl-whats-the-difference, accessed October 12, 2022.
[9] R. Gopalan, *The Cloud Data Lake*, 2.2.3.
[10] E.M. Haryono et al. Comparison of the E-LT vs ETL method in data warehouse implementation: A qualitative study. in: *2020 International Conference on Informatics, Multimedia, Cyber and Information System (ICIMCIS)* 2020, pp. 115–120. doi:10.1109/ICIMCIS51567.2020.9354284, accessed October 12, 2022.
[11] R. Gopalan, *The Cloud Data Lake*, 2.2.3.
[12] B. Schmidt et al. Time to adopt: Librarians' new skills and competency profiles. in: *Positioning and Power in Academic Publishing: Players, Agents and Agendas* 2016, pp. 1–8. doi:10.3233/978-1-61499-649-1-1, accessed October 12, 2022.
[13] K. Partlo, The pedagogical data reference interview, *IASSIST Quarterly* **33**(4) (2009), 6–10.
[14] P. Menon, *Data Lakehouse in Action*. Packt, Birmingham, UK, 2022, chap. 1.
[15] R. Gopalan, *The Cloud Data Lake*, chap. 2.
[16] M. Zaharia et al., Fast and interactive analytics over hadoop data with spark, *Usenix Login* **37**(4) (2012), 45–51.
[17] Ibid., 46.
[18] See https://www.databricks.com/, accessed October 12, 2022.
[19] See https://pandas.pydata.org/, accessed October 12, 2022.
[20] See https://dplyr.tidyverse.org/, accessed October 12, 2022.
[21] See https://spark.apache.org/docs/latest/rdd-programming-guide.html, accessed October 12, 2022.
[22] See https://avro.apache.org/, accessed October 12, 2022.
[23] See https://parquet.apache.org/, accessed October 12, 2022.
[24] See https://hive.apache.org/, accessed October 12, 2022.
[25] See https://lakefs.io/hive-metastore-why-its-still-here-and-what-can-replace-it/, accessed October 12, 2022.
[26] See https://aws.amazon.com/glue/, accessed October 12, 2022.
[27] See https://www.databricks.com/product/unity-catalog, accessed October 12, 2022.
[28] See https://www.nvidia.com/en-us/data-center/dgx-a100/, accessed October 12, 2022.
[29] See https://slurm.schedmd.com/documentation.html, accessed October 12, 2022.
[30] https://aws.amazon.com/lake-formation/?whats-new-cards.sort-by=item.additionalFields.postDateTime&whats-new-cards.sort-order=desc, accessed October 12, 2022.
[31] See https://azure.microsoft.com/en-us/solutions/data-lake/, accessed October 12, 2022.
[32] See https://cloud.google.com/biglake, accessed October 12, 2022.

[33] A. Gorelik, *What Is a Data Lake?* Table 2-2; Gopalan R., The Cloud Data Lake, 3.3.1.
[34] K. Partlo, The pedagogical data reference interview, 10.
[35] See https://www.gutenberg.org/, accessed October 12, 2022.
[36] See https://dumps.wikimedia.org/, accessed October 12, 2022.
[37] See https://wordnet.princeton.edu/, accessed October 12, 2022.
[38] R. Gopalan, *The Cloud Data Lake*, 1.4.3.
[39] A. Gorelik, *What Is a Data Lake?* chap. 1.
[40] P. Menon, *Data Lakehouse in Action*, chap. 2.
[41] A. Gorelik, *What Is a Data Lake?* Chap. 1; Menon P., *Data Lakehouse in Action*, Chap. 1.
[42] W. Brackenbury et al. Draining the data swamp: A similarity-based approach. in: *Proceedings of the Workshop on Human-In-the-Loop Data Analytics*HILDA'18. Association for Computing Machinery, New York, NY, 2018, pp. 1–7.10.1145/3209900.3209911, accessed October 12, 2022.
[43] P. Menon, *Data Lakehouse in Action*, chap. 2.
[44] See https://huggingface.co/, accessed October 12, 2022.
[45] T. Carpenter, Machines Don't Read The Way We Do | NISO Website https://www.niso.org/niso-io/2022/06/machines-dont-read-way-we-do, accessed October 12, 2022.