Data Model Innovations

by Sophia

WHAT'S COVERED

In this lesson, you will explore the major innovations and data model evolutions from the 1960s to the 2020s. Specifically, this lesson will cover:

1. Introduction

Databases have evolved significantly over the past 60 years in technology, scale, and capabilities. When databases were first developed in the 1960s, they were based on hierarchical and network models, where data was arranged as a tree or interconnected network. Its main use was to store and retrieve structured data, but it needed more scalability and flexibility. The relational database model was introduced in the 1970s, revolutionizing the field. During the development of relational databases, SQL (Structured Query Language) became the standard language for querying and manipulating data. Over the course of several decades, relational databases grew rapidly to become the dominant database model.

The database landscape has undergone further transformations in recent years. Cloud computing, big data, and distributed systems have led to new database technologies. As unstructured and semi-structured data volumes increase, NoSQL (non-relational) databases, such as key-value stores, document stores, columnar databases, and graph databases, have become increasingly popular. As an evolution of traditional relational databases, NewSQL databases offer superior scalability and performance while maintaining ACID compliance. As specialized databases, such as time-series, graph, and in-memory databases, have proliferated, organizations can leverage highly optimized data management and analysis solutions. As cloud-based databases and

database-as-a-service (DBaaS) offerings have evolved, deployment, management, and scalability have simplified.

As we saw with the shift from a manual file system to a computerized file system, there is always a focus on finding better ways to manage data. There have been many changes in computerized file systems, with each model trying to fix some of the shortcomings of the previous model. You will find that many of the newer database concepts have a significant resemblance to some of the older data models and concepts.

1a. Databases of the 1960s

In the 1960s, databases were in their infancy, and data management was still emerging. Most databases at that time followed a **hierarchical model** or **network model**. Hierarchical databases organize data in a tree-like structure, with parent-child relationships between records. This model was commonly used in early mainframe systems and allowed efficient data retrieval. Network databases, on the other hand, employ a more complex interconnected structure, where data was linked through pointers. This model provided more flexibility in representing relationships but took more work to manage. Databases in the 1960s were typically implemented on large-scale mainframe computers and used for managing structured data, such as financial records and inventory management.

Storage capacity and processing power were limited compared to today's standards, influencing database design and functionality in the 1960s. During this era, data was stored primarily on magnetic tapes and disks. Accessing and manipulating data required specific programming languages and interfaces. **Common Business-Oriented Language (COBOL)** was a popular programming language for data processing, and systems were designed to handle batch processing rather than real-time transactions.

TERMS TO KNOW

Hierarchical Models

Databases that store data as records and organize them into a tree-like structure where one parent node has many child nodes connected to it through links.

Network Model

The database in the network model represents objects and their relationships in a flexible manner. The schema differs from other hierarchical or lattice models because it is viewed as a graph with nodes and arcs representing objects and relationships.

Common Business-Oriented Language (COBOL)

A popular programming language for data processing; systems were designed to handle batch processing rather than real-time transactions.

2. Hierarchical Data Models of the 1970s

In the 1970s, hierarchical databases continued to be a popular method of managing and organizing data. Hierarchical databases arrange data in a tree-like structure based on a hierarchical structure, in which parent

and child relationships are formed. Hierarchical databases have a primary key for each record, and hierarchical links represent relationships between records. A parent record can have more than one child, but a child record can have only one parent. Data with clear hierarchical relationships, such as files or organizational structures, can be represented using this model.

A hierarchical database provides efficient retrieval and navigation of data since child records can be accessed only through the parent record. However, it is less suitable for representing more diverse and dynamic data structures because the hierarchical database model is not flexible when dealing with complex or changing relationships. In spite of this, the hierarchical model played a significant role in the early stages of database development, laying the foundation for subsequent database models.

3. Relational Models of the 1970s

A British computer scientist named Edgar Codd is said to be the father of relational databases. Codd's 1970 paper, "A Relational Model of Data for Large Shared Data Banks," established the relational model. Codd introduced an innovative way to manage and manipulate data by challenging hierarchical and network database models.

In a relational database, data is stored in tables consisting of rows and columns, with each table representing a relation in the relational model. Some of the key structural principles of relational databases include simplicity, independence from data, and mathematical foundations. A relational database is manipulated using formal mathematical tools called relational algebra and relational calculus.

Relational models have significant advantages, such as data independence, flexibility in retrieval, and the ability to establish relationships between tables using primary and foreign keys. In addition to facilitating efficient querying and analysis, they provide a logical and conceptual framework for managing data.

Codd's work led to the creation of Structured Query Language (SQL), an interface for working with relational databases. SQL became the de facto language for querying and manipulating relational databases. Having a common language made it easier to retrieve and manipulate data consistently and intuitively across different relational database systems.

Data management was transformed by the relational model, which is the foundation of most modern databases. The system paved the way for commercial relational database management systems (RDBMS) such as Oracle, IBM DB2, and Microsoft SQL Server. The use of these systems has become widespread in a variety of industries.

4. Object-Oriented in the 1980s

In the mid-1980s, there was a growing interest in combining object-oriented programming principles with the relational model of databases, leading to the emergence of object-relational and object-oriented databases.

Object-oriented programming (OOP) is a way of organizing computer programs around objects, which are self-contained units that represent real-world entities or concepts. These objects can have data (attributes) and behaviors (methods).

Object-relational databases (ORDBMS) bridge the gap between traditional relational databases and the object-oriented paradigm. They extend the relational model by supporting complex data types such as arrays, nested tables, and user-defined types. This allows for more flexibility in storing and querying structured and semi-structured data. ORDBMS also incorporate object-oriented concepts like inheritance and encapsulation, enabling the modeling of real-world entities as objects with associated attributes and behaviors.

In parallel, **object-oriented databases (OODBMS)** comprise a database management system (DBMS) that supports the modeling and creation of data as objects. OODBMS focus on directly integrating object-oriented programming principles into database management systems. OODBMS store data as objects and offer encapsulation, inheritance, and polymorphism mechanisms. They provide a seamless and transparent mapping

between the object-oriented programming language and the underlying database, offering benefits such as improved performance, data integrity, and support for complex relationships.

Both object-relational and object-oriented databases aim to provide a more natural and flexible way of representing and managing data, particularly for applications dealing with complex and evolving data structures. Over time, these database models gained attention and saw some adoption, but they didn't undermine the dominance of relational databases. Relational databases continue to be widely used due to their maturity, standardization, and the extensive ecosystem built around them.

EXAMPLE Some of the common object-oriented database types include Versant, Objectivity/DB, and Oracle 12c.

TERMS TO KNOW

Object-Relational Database (ORDBMS)

A type of database similar to a relational database but with an object-oriented database model; objects, classes and inheritance are directly supported in database schemas and in the query language.

Object-Oriented Programming (OOP)

A way of organizing computer programs around objects, which are self-contained units that represent real-world entities or concepts.

Object-Oriented Databases (OODBMS)

A database model that directly integrates object-oriented programming principles into database management, storing data as objects.

5. XML Hybrids in the 1990s

In the mid-1990s, **XML hybrid databases** emerged as a way to integrate XML (Extensible Markup Language) data into traditional database systems. XML is a widely used markup language for structuring and exchanging data, similar to HTML in web design. However, it doesn't naturally fit into the relational data model. XML hybrid databases combine relational databases' power with XML's flexibility. They store XML data in a structured manner, allowing querying and indexing of XML elements and attributes. These databases typically provide a mapping between XML and relational structures. This enables efficient storage and retrieval of XML data while leveraging the benefits of relational database management systems.

XML hybrid databases employ technologies like XML Schema Definition (XSD) to define XML documents' structure and validity constraints. They may support XML-specific query languages such as XQuery or XPath for querying XML data within the database. Additionally, they often provide mechanisms to handle XML updates, versioning, and integration with other data formats.

XML hybrid databases have found applications in various domains, such as content management systems, web services, and data integration scenarios where XML plays a significant role. They bridge the relational and XML worlds. This allows organizations to effectively manage and leverage XML data within their existing relational

database infrastructure, facilitating interoperability and data integration across different systems and technologies.

TERM TO KNOW

XML Hybrid Database

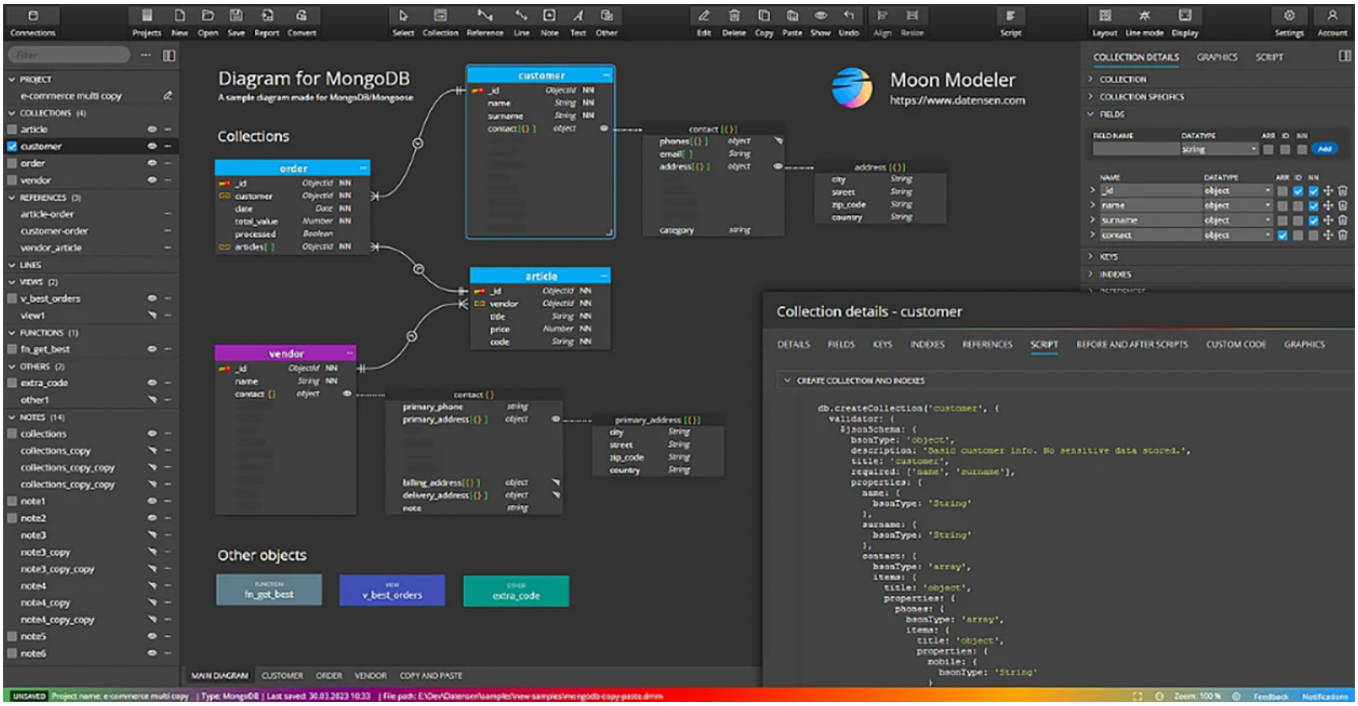
A database type that bridges the structured formatting of XML and the relational model of modern databases.

6. NoSQL in the 2000s and Beyond

The current generation of emerging data models from the early 2000s to the current time focuses on **NoSQL**. "NoSQL" stands for "not only SQL" and refers to a class of databases that are non-relational, as you learned earlier in this course. They enable the storage and retrieval of large volumes of unstructured and semi-structured data in a flexible and scalable manner. In contrast to relational databases, which are always structured as tables with relationships between them, NoSQL databases can employ a variety of data models, such as key-value, document, columnar, and graph. With each model optimized for a specific use case, organizations can select the most appropriate model based on their data needs.

Data models that are flexible, horizontally scalable, and fast performance make NoSQL databases ideal. Their ability to handle big data, real-time data streams, and distributed systems makes them an excellent choice. In order to support high scalability and data availability, NoSQL databases offer automated sharing, replication, and fault tolerance features. Data can be distributed across multiple servers or clusters to provide horizontal scaling. The system can handle rapidly growing workloads and efficiently process large datasets.

In spite of their flexibility and scalability, NoSQL databases may sacrifice some features of traditional relational databases, such as strong consistency guarantees and complex querying capabilities. NoSQL databases, however, have proved useful for many modern applications, including web applications, social networking platforms, IoT (Internet of Things), and real-time analytics. They cater to the evolving needs of handling diverse and massive data sources by offering a different paradigm for data management.



EXAMPLE There are many available options with this model, including SimpleDB from Amazon, Bigtable from Google, Cassandra from Apache, and MongoDB.

TERM TO KNOW

NoSQL

Not only SQL; a type of data system designed around unstructured data that needs to be processed using a high availability and scalable process.

SUMMARY

In this lesson, you learned that database technology has undergone significant advancements and transformations from the 1960s to today. **In the 1960s**, databases were mainly **hierarchical and network-based data models**, with data organized in tree-like or interconnected structures. A **relational model was introduced in the 1970s** by Edgar F. Codd, and it revolutionized the field by introducing tables with rows, columns, and relationships between them. As SQL became the standard data manipulation and querying language, relational databases became the dominant model.

New database technologies emerged with the rise of the internet, big data, and distributed computing in the late 20th and early 21st century. This included **object-oriented in the 1980s** and **XML hybrids in the 1990s**. Data models such as key-value, document, columnar, and graph were developed for NoSQL databases to handle unstructured and semi-structured data. These databases were scalable, performant, and capable of handling a wide variety of data types. Cloud computing has transformed the database landscape, enabling on-demand database services, flexible scaling, and greater accessibility.

The current generation of databases focuses on NoSQL, a class of non-relational databases that enable storage and retrieval of large volumes of unstructured and semi-structured data in a flexible and

scalable manner. Despite their benefits, NoSQL databases may sacrifice some features of relational databases, such as complex querying and strong consistency guarantees.

Source: THIS TUTORIAL WAS AUTHORED BY DR. VINCENT TRAN, PHD (2020) AND Faithe Wempen (2024) FOR SOPHIA LEARNING. PLEASE SEE OUR **TERMS OF USE**.

TERMS TO KNOW

Common Business-Oriented Language (COBOL)
A popular programming language for data processing; systems were designed to handle batch processing rather than real-time transactions.

Hierarchical Models
Databases that store data as records and organize them into a tree-like structure where one parent node has many child nodes connected to it through links.

Network Model
The database in the network model represents objects and their relationships in a flexible manner. The schema differs from other hierarchical or lattice models because it is viewed as a graph with nodes and arcs representing objects and relationships.

NoSQL
Not only SQL; a type of data system designed around unstructured data that needs to be processed using a high availability and scalable process.

Object-Oriented Databases (OODBMS)
A database model that directly integrates object-oriented programming principles into database management, storing data as objects.

Object-Oriented Programming (OOP)
A way of organizing computer programs around objects, which are self-contained units that represent real-world entities or concepts.

Object-Relational Database (ORDBMS)
A type of database similar to a relational database but with an object-oriented database model; objects, classes and inheritance are directly supported in database schemas and in the query language.

XML Hybrid Database
A database type that bridges the structured formatting of XML and the relational model of modern databases.