Data warehouse
and
Data Lake

Databases vs. Data Warehouses vs. Data Lakes

"database," "data warehouse," and "data lake"

Are these different words to describe the same thing?

If not, what are the differences?

And when should you choose one over the other?

Contents

Year wise progression of Databases

1960s

- 1960: Development of COBOL
- 1964: Introduction of IMS by IBM

1970s

- 1970: E. F. Codd's Relational Model
- 1976: Entity-Relationship Model
- 1979: Introduction of Oracle Database

1980s

- 1980: IBM's DB2
- 1984: SQL Standardization
- 1986: Introduction of Informix

1990s

- 1995: Introduction of MySQL
- 1997: Introduction of Microsoft SQL Server
- 1999: Introduction of XML for Data Interchange

2000s

- 2000: Introduction of NoSQL Databases
- 2006: Introduction of Amazon DynamoDB
- 2008: Introduction of Apache Hadoop

2010s

- 2010: Introduction of Apache Cassandra
- 2012: Introduction of Google BigQuery
- 2014: Introduction of Apache Spark

2020s

- 2020: Rise of Cloud-Native Databases
- 2021: Introduction of Databricks Lakehouse Platform
- 2023: Emergence of AI-Powered Databases

Job Roles

1960s

- 1960-1969: Data Librarian

1970s

- 1970-1979: Database Administrator (DBA)
- 1976: Data Architect

1980s

- 1980-1989: Data Analyst
- 1986: Database Developer

1990s

- 1990-1999: Data Warehouse Specialist
- 1995: Business Intelligence (BI) Developer

2000s

- 2000-2009: Data Scientist
- 2005: Data Engineer

2010s

- 2010-2019: Machine Learning Engineer
- 2012: Data Governance Specialist

2020s

- 2020-Present: DataOps Engineer
- 2020-Present: Analytics Engineer

What is a database?

- A database is a collection of data or information
- Online Transaction Processing (OLTP)
- Database Management Systems (DBMS)
  - Store data in the database and enable users and applications to interact with the data.
- “database” is commonly used to reference both the database itself as well as the DBMS

Database characteristics

- Relational databases
  - store data in tables with fixed rows and columns
- Non-relational databases (also known as NoSQL databases)
  - store data in a variety of models including JSON (JavaScript Object Notation), BSON (Binary JSON), key-value pairs, tables with rows and dynamic columns, and nodes and edges.

Database characteristics

- Security features to ensure the data can only be accessed by authorized users.
- ACID (Atomicity, Consistency, Isolation, Durability) transactions to ensure data integrity.
- Query languages and APIs to easily interact with the data in the database.
- Indexes to optimize query performance.
- Full-text search.
- Optimizations for mobile devices.
- Flexible deployment topologies to isolate workloads (e.g., analytics workloads) to a specific set of resources.
- On-premises, private cloud, public cloud, hybrid cloud, and/or multi-cloud hosting options.

Why use a database?

- Patient medical records
- Items in an online store
- Financial records
- Articles and blog entries
- Sports scores and statistics
- Online gaming information
- Student grades and scores
- IoT device readings
- Mobile application information

Database examples

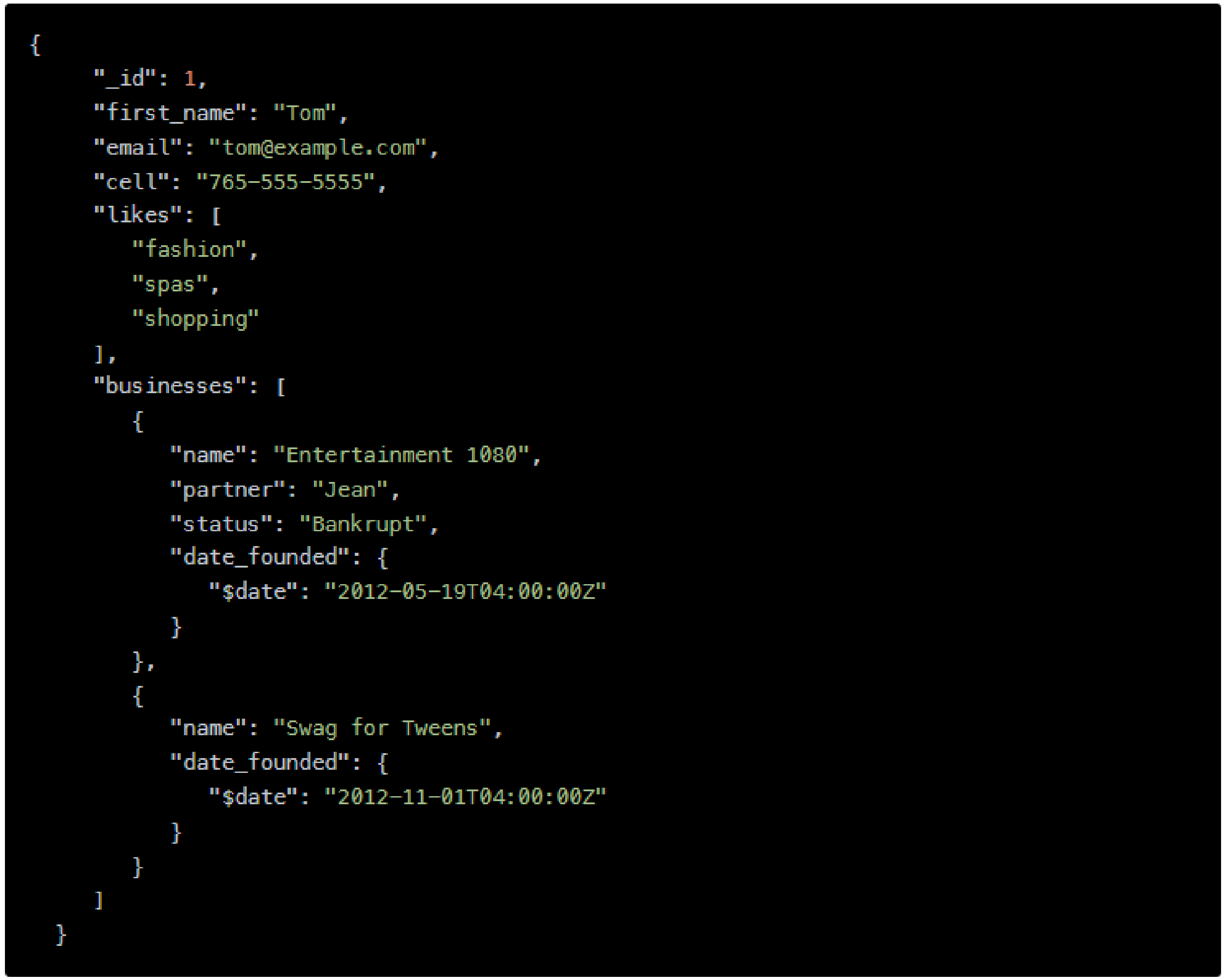
- Relational databases: Oracle, MySQL, Microsoft SQL Server, and PostgreSQL
- Document databases: MongoDB and CouchDB
- Key-value databases: Redis and DynamoDB
- Wide-column stores: Cassandra and HBase
- Graph databases: Neo4j and Amazon Neptune

Document Database

- A document is a record in a document database. A document typically stores information about one object and any of its related metadata.
- Documents store data in field-value pairs. The values can be a variety of types and structures, including strings, numbers, dates, arrays, or objects. Documents can be stored in formats like JSON, BSON, and XML.

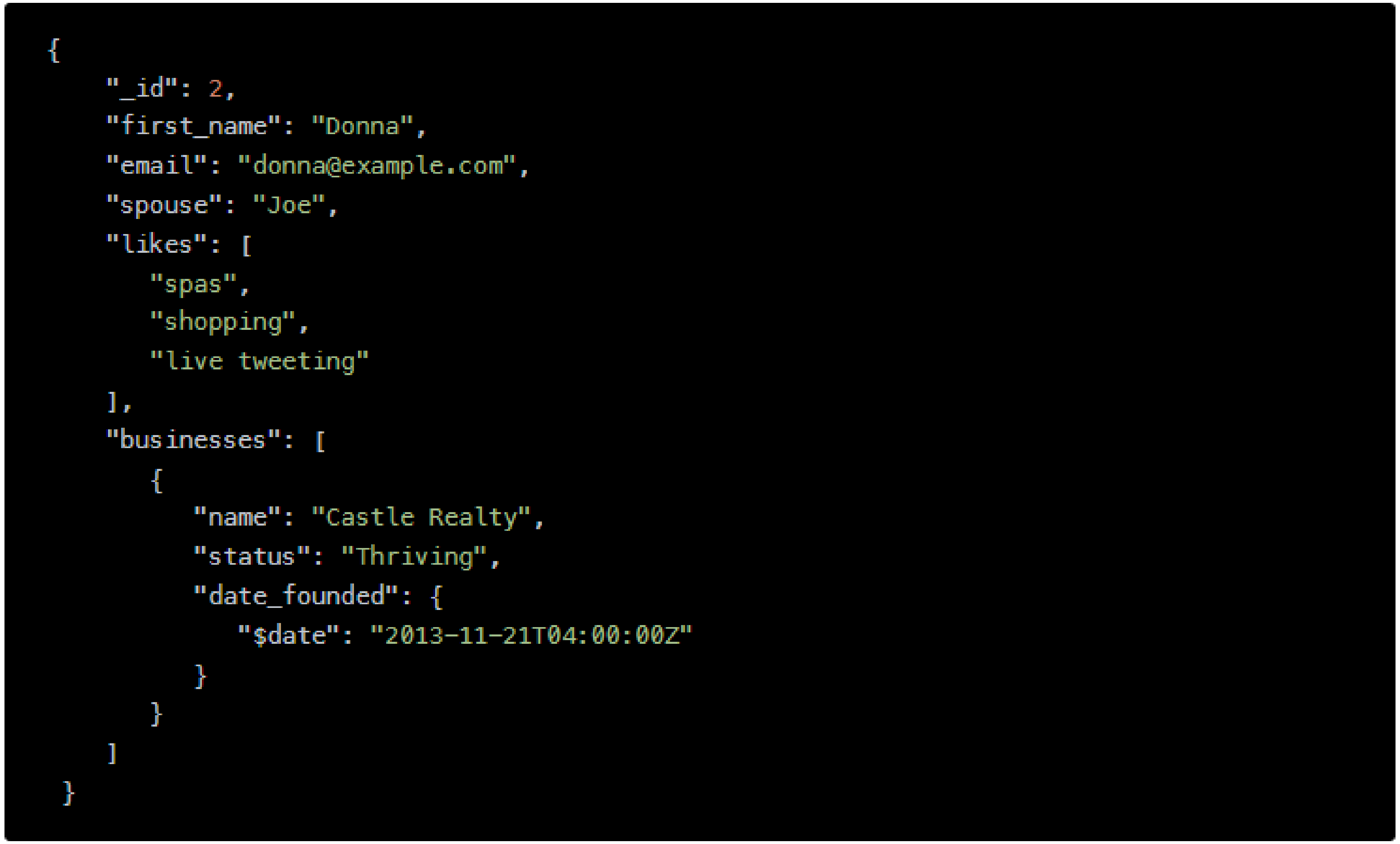
An example

Below is a JSON document that stores information about a user named Tom.



```
{
     "_id": 1,
     "first_name": "Tom",
     "email": "tom@example.com",
     "cell": "765-555-5555",
     "likes": [
        "fashion",
        "spas",
        "shopping"
     ],
     "businesses": [
        {
           "name": "Entertainment 1080",
           "partner": "Jean",
           "status": "Bankrupt",
           "date_founded": {
              "$date": "2012-05-19T04:00:00Z"
           }
        },
        {
           "name": "Swag for Tweens",
           "date_founded": {
              "$date": "2012-11-01T04:00:00Z"
           }
        }
     ]
  }
```

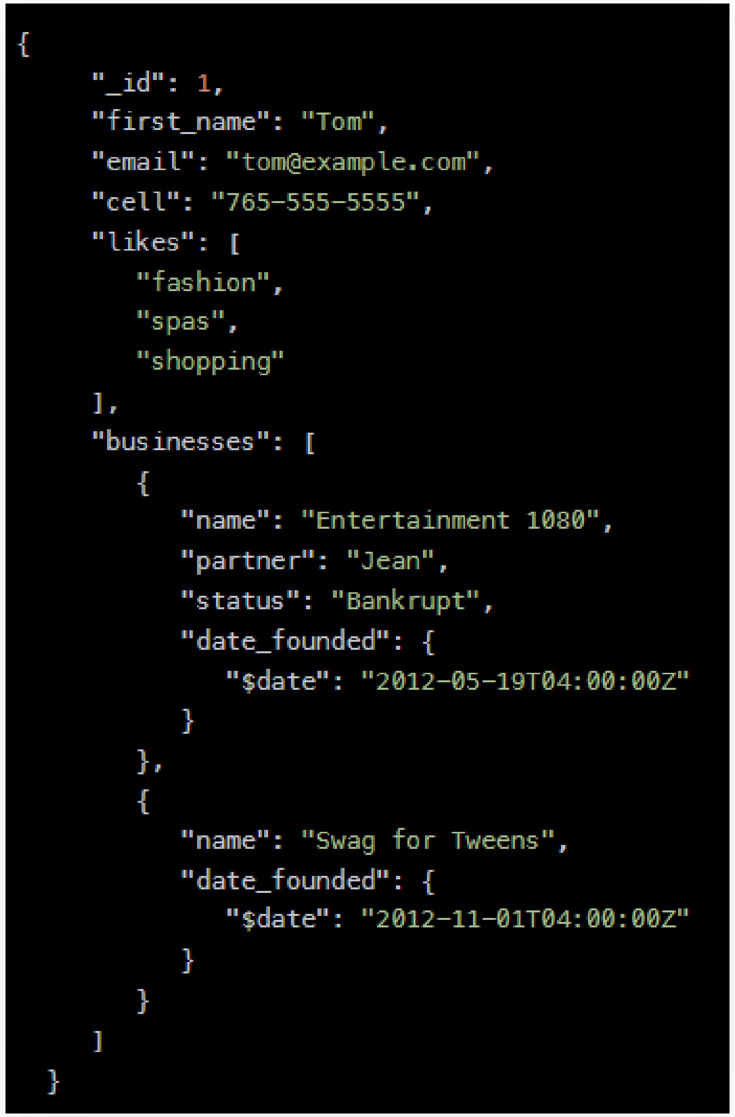
Document below that stores information about Donna could be added to the users collection



```
{
    "_id": 2,
    "first_name": "Donna",
    "email": "donna@example.com",
    "spouse": "Joe",
    "likes": [
       "spas",
       "shopping",
       "live tweeting"
    ],
    "businesses": [
       {
          "name": "Castle Realty",
          "status": "Thriving",
          "date_founded": {
             "$date": "2013-11-21T04:00:00Z"
          }
       }
    ]
 }
```

Key features of document databases

- **Document model:** Data is stored in documents (unlike other databases that store data in structures like tables or graphs). Documents map to objects in most popular programming languages, which **allows developers to rapidly develop their applications**.
- **Flexible schema:** Document databases have flexible schemas, meaning that not all documents in a collection need to have the same fields. Note that some document databases support schema validation, so the schema can be optionally locked down.
- **Distributed and resilient:** Document databases are distributed, which allows for horizontal scaling (typically cheaper than vertical scaling) and data distribution. Document databases provide resiliency through replication.
- **Querying through an API or query language:** Document databases have an API or query language that allows developers to execute the CRUD operations on the database. Developers have the ability to query for documents based on unique identifiers or field values.



```
{
     "_id": 1,
     "first_name": "Tom",
     "email": "tom@example.com",
     "cell": "765-555-5555",
     "likes": [
        "fashion",
        "spas",
        "shopping"
     ],
     "businesses": [
        {
           "name": "Entertainment 1080",
           "partner": "Jean",
           "status": "Bankrupt",
           "date_founded": {
              "$date": "2012-05-19T04:00:00Z"
           }
        },
        {
           "name": "Swag for Tweens",
           "date_founded": {
              "$date": "2012-11-01T04:00:00Z"
           }
        }
     ]
  }
```

Likes

ID	user_id	like
10	1	fashion
11	1	spas
12	1	shopping

Users

ID	first_name	email	cell
1	Tom	tom@example.com	765-555-5555

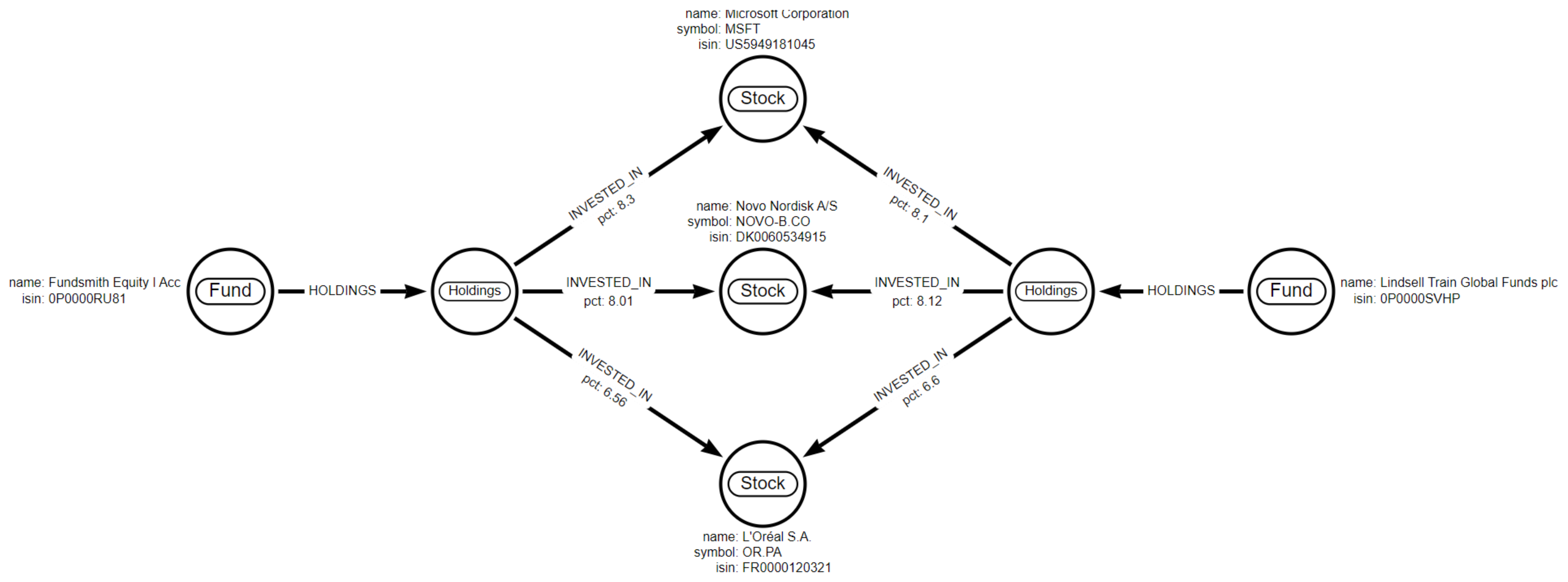
Businesses

ID	user_id	name	partner	status	date_founded
20	1	Entertainment 1080	Jean	Bankrupt	2011-05-19
21	1	Swag for Tweens	NULL	NULL	2012-11-01

Why not just use JSON in a relational database?

1982
2000
1998
1991
CENTRAL CITY POLICE DEPARTMENT
The Central City Tribune
NORA ALLEN MURDERED BY HUSBAND
TEN YEARS IS NEW YEAR APPROACHES
The Central City Tribune
OCTOR ACCUSED OF KILLING WIF
ORA ALLEN DIES
S QUESTIONED
CENTRAL CITY POLICE DEPARTMENT
DNA EVID
INCONCL
CENTRAL CITY POLICE DEPARTMENT
CENTRAL CITY POLICE DEPART

Graph Databases



https://neo4j.com/

Portfolio Diversification Oversight

- Clients' portfolios may inadvertently need more diversification due to over-concentration in specific stocks.
- Inadequate insight into underlying stock dependencies can lead to unexpected vulnerabilities when market conditions change.
- With a holistic view, identifying the extent of cross-dependencies between different funds is easier, potentially leading to undetected risks.

Risk Assessment and Exposure Evaluation

- The absence of detailed dependency mapping makes it difficult to assess the overall risk exposure of a fund’s portfolio.
- In cases of over-leverage to specific stocks, a market downturn could trigger cascading losses impacting multiple funds.

Regulatory Compliance and Reporting

- Regulatory bodies increasingly require accurate and transparent reporting on investment dependencies to prevent market manipulation and fraud.

- Without a unified solution, generating comprehensive dependency reports becomes a manual and error-prone task.

- Organisations risk non-compliance and reputational damage if they cannot provide timely and accurate dependency data to regulatory authorities.

Data warehouse

What is a data warehouse?

- A data warehouse is a system that stores highly structured information from various sources.
- Data warehouses typically store current and historical data from one or more systems.
- The goal of using a data warehouse is to combine disparate data sources in order to analyze the data, look for insights, and create business intelligence (BI) in the form of reports and dashboards.
- "Is a data warehouse a database?"
    - Yes, a data warehouse is a giant database that is optimized for analytics.

Data warehouse characteristics

- Extract, transform, load (ETL) processes move data from its original source to the data warehouse.
- They work well with structured data. Some data warehouses also support semi-structured data.

Why use a data warehouse?

- Data warehouses are a good option when you need to store large amounts of historical data and/or perform in-depth analysis of your data to generate business intelligence. Due to their highly structured nature, analyzing the data in data warehouses is relatively straightforward and can be performed by business analysts and data scientists.
- Note that data warehouses are not intended to satisfy the transaction and concurrency needs of an application. If an organization determines they will benefit from a data warehouse, they will need a separate database or databases to power their daily operations.

Data warehouse examples

- Amazon Redshift.
- Google BigQuery.
- IBM Db2 Warehouse.
- Microsoft Azure Synapse.
- Oracle Autonomous Data Warehouse.
- Snowflake.
- Teradata Vantage.

What is a data lake?

- A data lake is a repository of data from disparate sources that is stored in its original, raw format.
- their ability to store data in a variety of formats including JSON, BSON, CSV, TSV, Avro, ORC, and Parquet.
- the primary purpose of a data lake is to analyze the data to gain insights
- "Is a data lake a database?"
  - A data lake is a repository for data stored in a variety of ways including databases. With modern tools and technologies, a data lake can also form the storage layer of a database.
  - Tools like Starburst, Presto, Dremio, and Atlas Data Lake can give a database-like view into the data stored in your data lake.

Data lake characteristics

- Data lakes store large amounts of structured, semi-structured, and unstructured data. They can contain everything from relational data to JSON documents to PDFs to audio files.
- Data does not need to be transformed in order to be added to the data lake, which means data can be added (or "ingested") incredibly efficiently without upfront planning.
- The primary users of a data lake can vary based on the structure of the data. Business analysts will be able to gain insights when the data is more structured. When the data is more unstructured, data analysis will likely require the expertise of developers, data scientists, or data engineers.
- The flexible nature of data lakes enables business analysts and data scientists to look for unexpected patterns and insights. The raw nature of the data combined with its volume allows users to solve problems they may not have been aware of when they initially configured the data lake.
- Data in data lakes can be processed with a variety of OLAP systems and visualized with BI tools.

Why use a data lake?

- Data lakes are a cost-effective way to store huge amounts of data. Use a data lake when you want to gain insights into your current and historical data in its raw form without having to transform and move it. Data lakes also support machine learning and predictive analytics.
- Like data warehouses, data lakes are not intended to satisfy the transaction and concurrency needs of an application.

Data lake examples

- AWS S3
- Azure Data Lake Storage Gen2
- Google Cloud Storage

Key Summaries

- A database stores the current data required to power an application.
- A data warehouse stores current and historical data from one or more systems in a predefined and fixed schema, which allows business analysts and data scientists to easily analyze the data.
- A data lake stores current and historical data from one or more systems in its raw form, which allows business analysts and data scientists to easily analyze the data.

Comparision

Database	Data Lake	Data Warehouse	
Workloads	Operational and transactional	Analytical	Analytical
Data Type	Structured or semi-structured	Structured, semi-structured, and/or unstructured	Structured and/or semi-structured
Schema Flexibility	Rigid or flexible schema depending on database type	No schema definition required for ingest (schema on read)	Pre-defined and fixed schema definition for ingest (schema on write and read)
Data Freshness	Real time	May not be up-to-date based on frequency of ETL processes	May not be up-to-date based on frequency of ETL processes

Comparision

Users	Application developers	Business analysts, application developers, and data scientists	Business analysts and data scientists
Pros	Fast queries for storing and updating data	Easy data storage simplifies ingesting raw data A schema is applied afterwards to make working with the data easy for business analysts Separate storage and compute	The fixed schema makes working with the data easy for business analysts
Cons	May have limited analytics capabilities	Requires effort to organize and prepare data for use	Difficult to design and evolve schema Scaling compute may require unnecessary scaling of storage, because they are tightly coupled

which is right for me?

- **Is my data structured, semi-structured, or unstructured?**
  - Data warehouses support structured and semi-structured data whereas data lakes support all three.
- **Will my analysis benefit from having a pre-defined, fixed schema?**
  - Data warehouses require users to create a pre-defined, fixed schema upfront, which lends itself to more limited (but easier) data analysis. Data lakes allow users to store data in its raw, original format, which makes it easier to store data without having to apply and maintain structure.
- **Where is my data currently stored?**
  - Data warehouses require you to create ETL processes to move your data into the warehouse. Depending on where the data is stored, a data lake may not require any data to be moved. For example, MongoDB Atlas Data Lake is able to access data stored in an Amazon S3 bucket, which can be quite advantageous for organizations who are already storing their data there.

Relative benefits

- Data Warehouses: Ideal for business intelligence, reporting, and historical data analysis. Suited for scenarios requiring structured data and complex queries.
- Data Lakes: Suitable for advanced analytics, machine learning, and real-time processing. Handles diverse data types and large volumes of data.
- Data Warehouses: Generally more expensive due to the cost of storage and processing structured data. Optimized for performance but can be cost-prohibitive for storing large volumes of raw data.
- Data Lakes: More cost-effective for storing vast amounts of raw data. Scalability and flexibility make them a popular choice for organizations dealing with diverse data sources.

An example of Data Lake in Banking

- Consider a bank that needs to analyze customer transaction data to detect fraudulent activities. By using a data warehouse, the bank can store historical transaction data in a structured format, making it easier to run complex queries and generate reports. Simultaneously, the bank can leverage a data lake to store real-time transaction data from various sources, enabling real-time fraud detection using machine learning algorithms.

- Imagine a fintech company that needs to generate regular compliance reports and perform ad-hoc analyses on customer transaction data.
  - Dataware house or Data lake?
  - For compliance reports, a data warehouse is suitable because it provides structured data storage and efficient querying capabilities.
- For exploring new customer insights using machine learning
  - Dataware house or Data lake?
  - A data lake is more appropriate due to its ability to store and process raw data from various sources.

Data Warehouse Architecture

- Key components include the ETL process, schema design, data marts, and OLAP systems
  - o **Extract:** Data is extracted from various source systems, such as transactional databases, flat files, and external APIs.
  - **Transform:** The extracted data is cleaned, transformed, and standardized to ensure consistency and accuracy. This step may involve data cleansing, data enrichment, and data aggregation.
  - **Load:** The transformed data is loaded into the data warehouse, where it is organized into tables and schemas for efficient querying.

Dataware house schema design

- Schema design is crucial for organizing data within the warehouse.
- Two common schema designs are the star schema and the snowflake schema:
    - **Star Schema:** A simple, denormalized structure with a central fact table surrounded by dimension tables. This design is easy to understand and use, making it suitable for straightforward querying and reporting.
    - **Snowflake Schema:** A more complex, normalized structure where dimension tables are further broken down into related tables. This design reduces data redundancy and improves data integrity but can be more challenging to query.

Justify and Give example

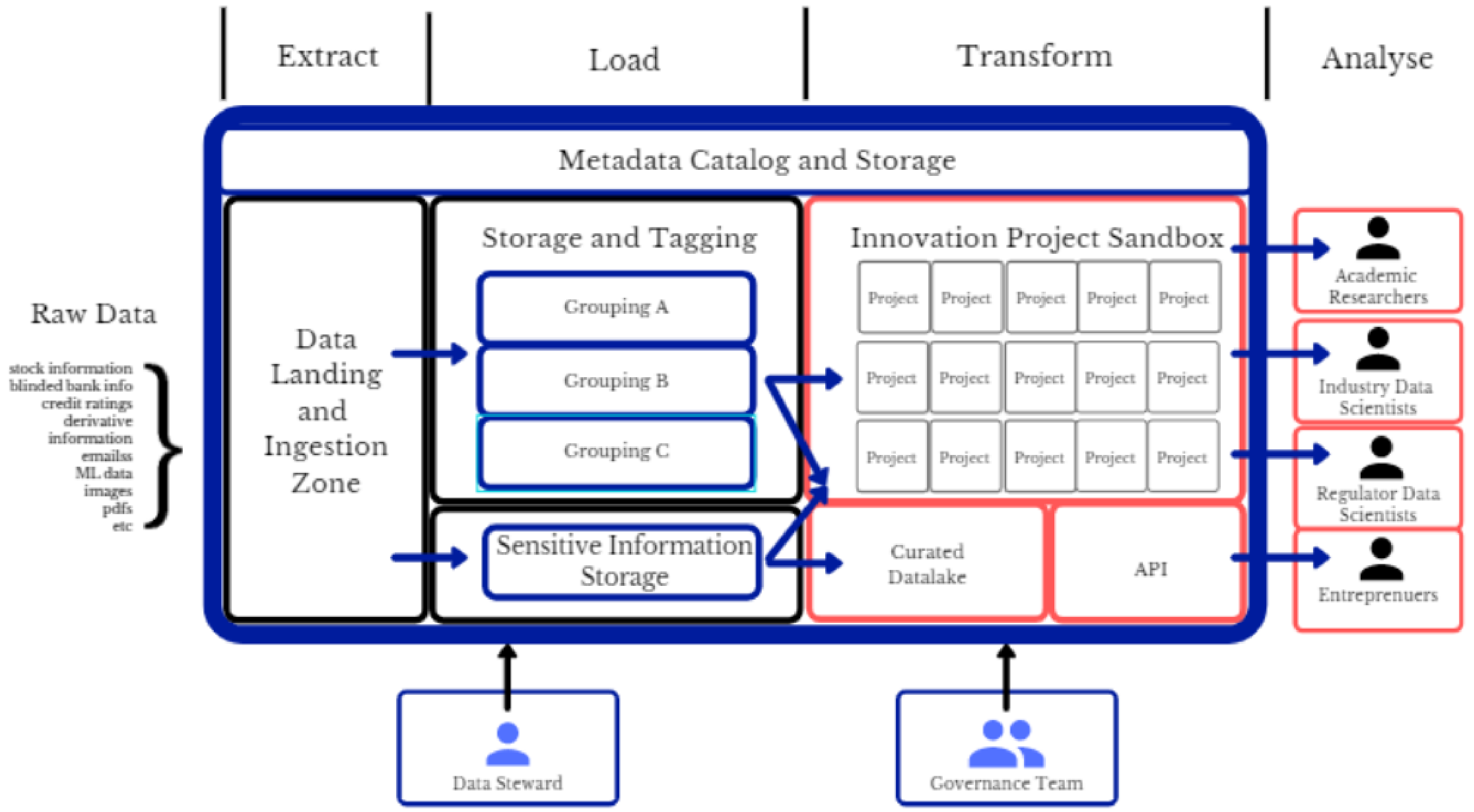
- The star schema is easy to understand and use, making it suitable for straightforward querying and reporting. Its denormalized structure reduces the complexity of queries, improving performance.
  - Ideal for simple reporting and analysis scenarios where ease of use and query performance are priorities.

- The snowflake schema enhances data integrity and consistency by reducing redundancy. It also provides greater flexibility for handling complex data relationships.
  - Suitable for complex analysis scenarios where data integrity and flexibility are more important than query simplicity.

Bank's Data Warehouse Use Case

- Managing Customer Data
  - transaction records, customer profiles, and external data feeds – ETL
- Generating Insights
  - bank can conduct detailed analyses to gain insights into customer behavior, transaction patterns, and market trends.
    - identify high-value customers, analyze spending habits, and detect potential fraud.
- Supporting Decision-Making
  - Targeted marketing campaigns, optimize product offerings, and improve customer satisfaction

The building blocks of a Fintech data lake



https://pure.strath.ac.uk/ws/portalfiles/portal/92191270/Broby_Hopper_CFRI_2019_Creating_a_Financial_Data_Lake_for_Academic.pdf

Case study

re:Invent 2016: Steve Randich of FINRA Describes How Using AWS Has Improved System Performance 400X

https://aws.amazon.com/solutions/case-studies/finra/

Case study

National Australia Bank
YURI MISNIK
Executive GM, Business Enabling Technology, National Australia Bank

AWS re:Invent 2018: National Australia Bank Begins Enterprise Transformation Using AWS

https://aws.amazon.com/solutions/case-studies/national-australia-bank/

Discussion on research article

- Applications of Generative AI in Fintech

Students read the usecases

- Go throurgh the usecases from AWS, Azure and GC for financial services and submit a report upto two pages.
    - AWS
    - Azure