Data Engineering Fundamentals

SOURCES, FORMATS, MODELS AND DATA PROCESSING

STUDENT: PAULO EDUARDO DA SILVA JUNIOR

Summary

1. Data Sources;
2. Data Formats;
3. Data Models;
4. Storage and Processing Engines;
5. Data Flow Modes;
6. Batch Processing vs. Stream Processing;
7. Conclusion.

Data Sources

1. User input: text, images.
2. System-generated data: logs.
3. Internal databases.
4. Third-party data.

Data Formats

1. JSON: Structured and unstructured, human readable.
2. Row-Major (CSV): Fast row access.
3. Column-Major (Parquet): Fast column access, binary.
4. Text vs. Binary: Binary saves space and is faster.

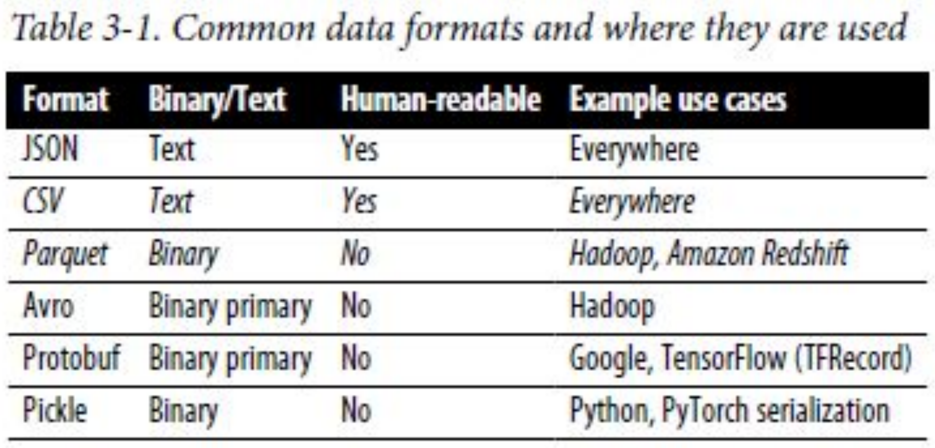


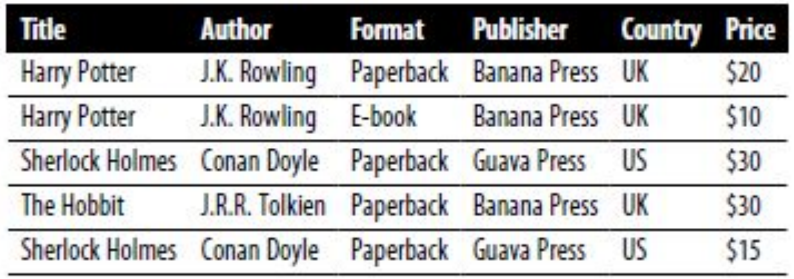
Table 3-1. Common data formats and where they are used

Format	Binary/Text	Human-readable	Example use cases
JSON	Text	Yes	Everywhere
CSV	*Text*	*Yes*	*Everywhere*
Parquet	*Binary*	*No*	*Hadoop, Amazon Redshift*
Avro	Binary primary	No	Hadoop
Protobuf	Binary primary	No	Google, TensorFlow (TFRecord)
Pickle	Binary	No	Python, PyTorch serialization

Data Models

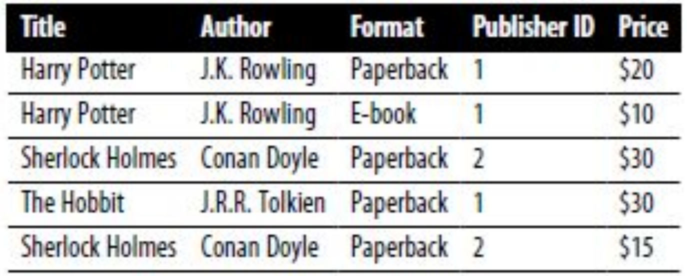
1. Relational: Normalized data, complex queries.

Table 3-2. Initial Book relation



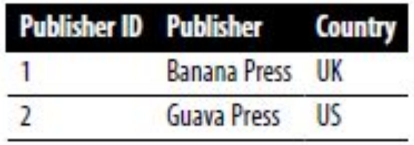
Title	Author	Format	Publisher	Country	Price
Harry Potter	J.K. Rowling	Paperback	Banana Press	UK	$20
Harry Potter	J.K. Rowling	E-book	Banana Press	UK	$10
Sherlock Holmes	Conan Doyle	Paperback	Guava Press	US	$30
The Hobbit	J.R.R. Tolkien	Paperback	Banana Press	UK	$30
Sherlock Holmes	Conan Doyle	Paperback	Guava Press	US	$15

Table 3-3. Updated Book relation



Title	Author	Format	Publisher ID	Price
Harry Potter	J.K. Rowling	Paperback	1	$20
Harry Potter	J.K. Rowling	E-book	1	$10
Sherlock Holmes	Conan Doyle	Paperback	2	$30
The Hobbit	J.R.R. Tolkien	Paperback	1	$30
Sherlock Holmes	Conan Doyle	Paperback	2	$15

Table 3-4. Publisher relation



Publisher ID	Publisher	Country
1	Banana Press	UK
2	Guava Press	US

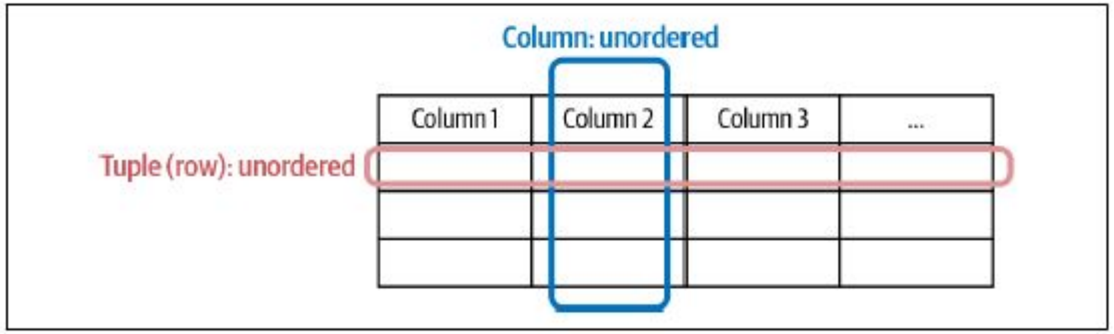


Figure 3-4. In a relation, the order of neither the rows nor the columns matters

Data Models

1. Relational: Normalized data, complex queries.
2. NoSQL: Flexibility and efficiency in complex relationships.

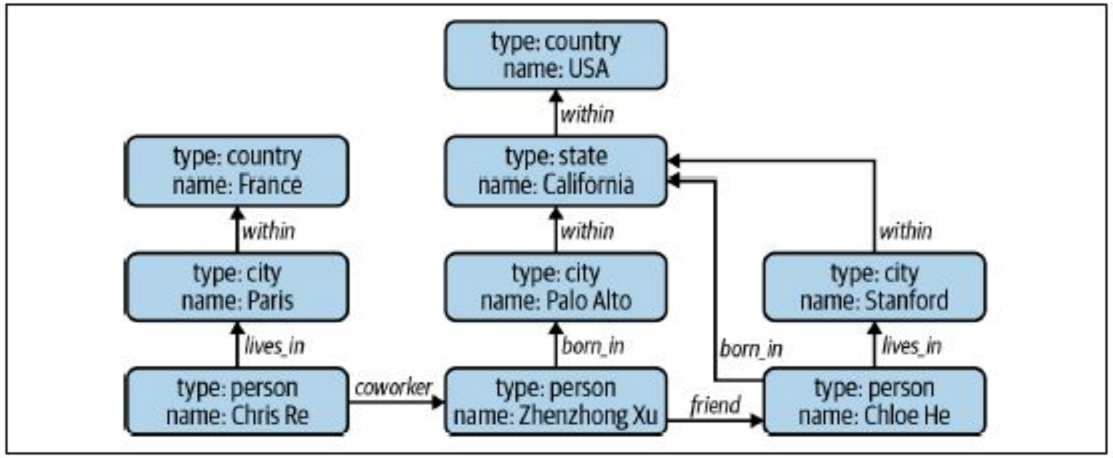
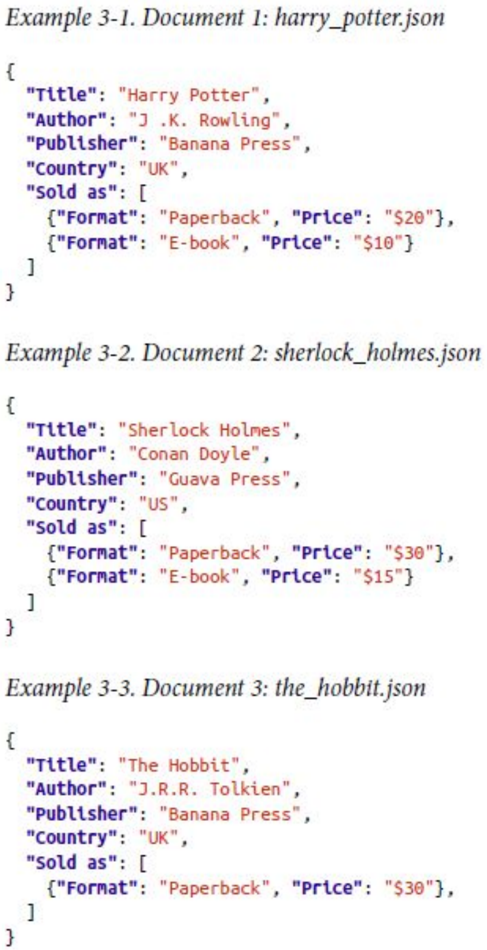


Figure 3-5. An example of a simple graph database



Example 3-1. Document 1: harry_potter.json

```
{
  "Title": "Harry Potter",
  "Author": "J .K. Rowling",
  "Publisher": "Banana Press",
  "Country": "UK",
  "Sold as": [
    {"Format": "Paperback", "Price": "$20"},
    {"Format": "E-book", "Price": "$10"}
  ]
}
```

Example 3-2. Document 2: sherlock_holmes.json

```
{
  "Title": "Sherlock Holmes",
  "Author": "Conan Doyle",
  "Publisher": "Guava Press",
  "Country": "US",
  "Sold as": [
    {"Format": "Paperback", "Price": "$30"},
    {"Format": "E-book", "Price": "$15"}
  ]
}
```

Example 3-3. Document 3: the_hobbit.json

```
{
  "Title": "The Hobbit",
  "Author": "J.R.R. Tolkien",
  "Publisher": "Banana Press",
  "Country": "UK",
  "Sold as": [
    {"Format": "Paperback", "Price": "$30"},
  ]
}
```

Data Models

1. Relational: Normalized data, complex queries.
2. NoSQL: Flexibility and efficiency in complex relationships.
3. Structured vs. Unstructured: Schema-based vs. flexible data in data lakes.

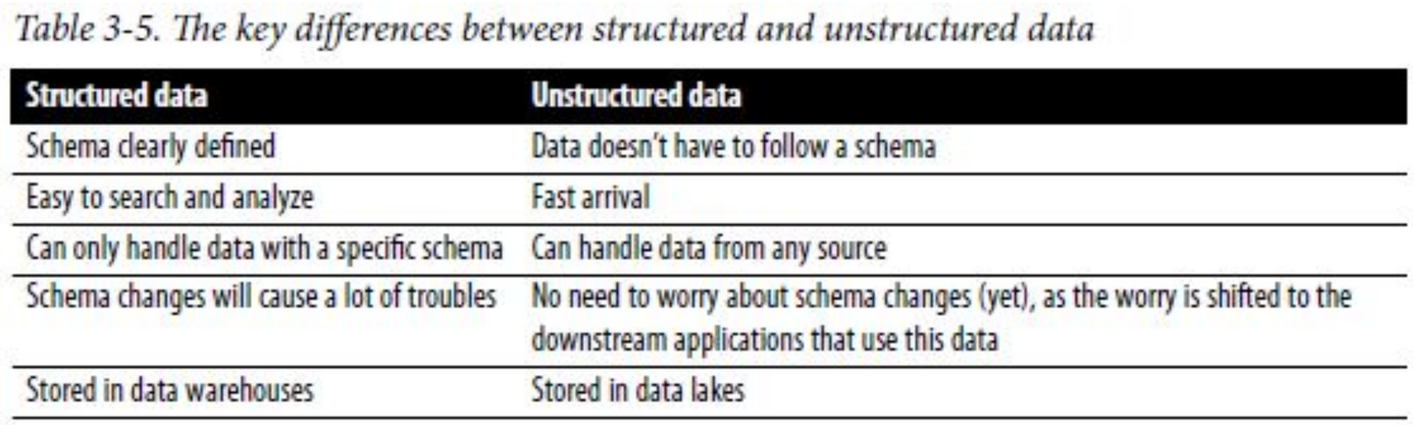


Table 3-5. The key differences between structured and unstructured data

Structured data	Unstructured data
Schema clearly defined	Data doesn't have to follow a schema
Easy to search and analyze	Fast arrival
Can only handle data with a specific schema	Can handle data from any source
Schema changes will cause a lot of troubles	No need to worry about schema changes (yet), as the worry is shifted to the downstream applications that use this data
Stored in data warehouses	Stored in data lakes

Storage and Processing Engines

1. OLTP: Rapid transaction processing.
2. OLAP: Complex queries on large volumes.
3. ETL: Data extraction, transformation, and loading.

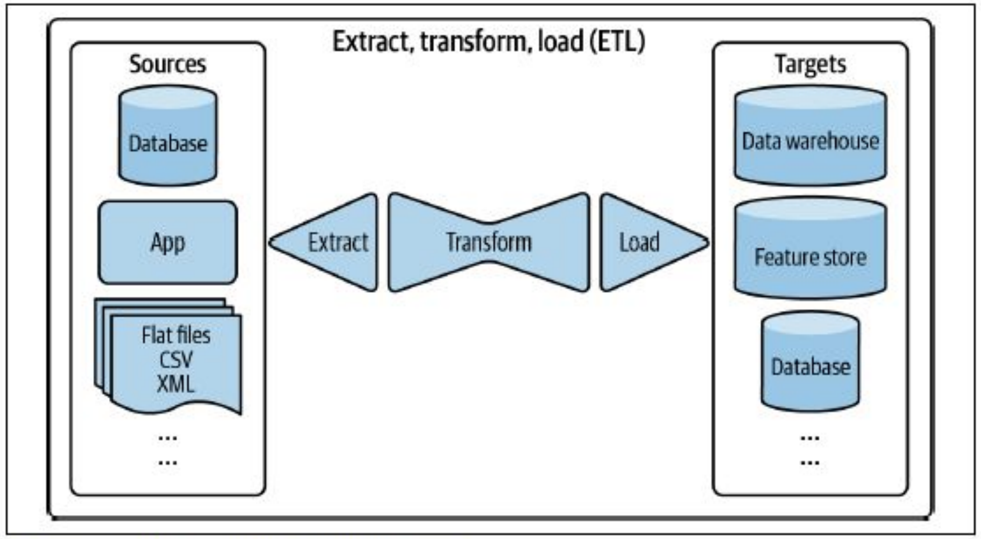


Figure 3-7. An overview of the ETL process

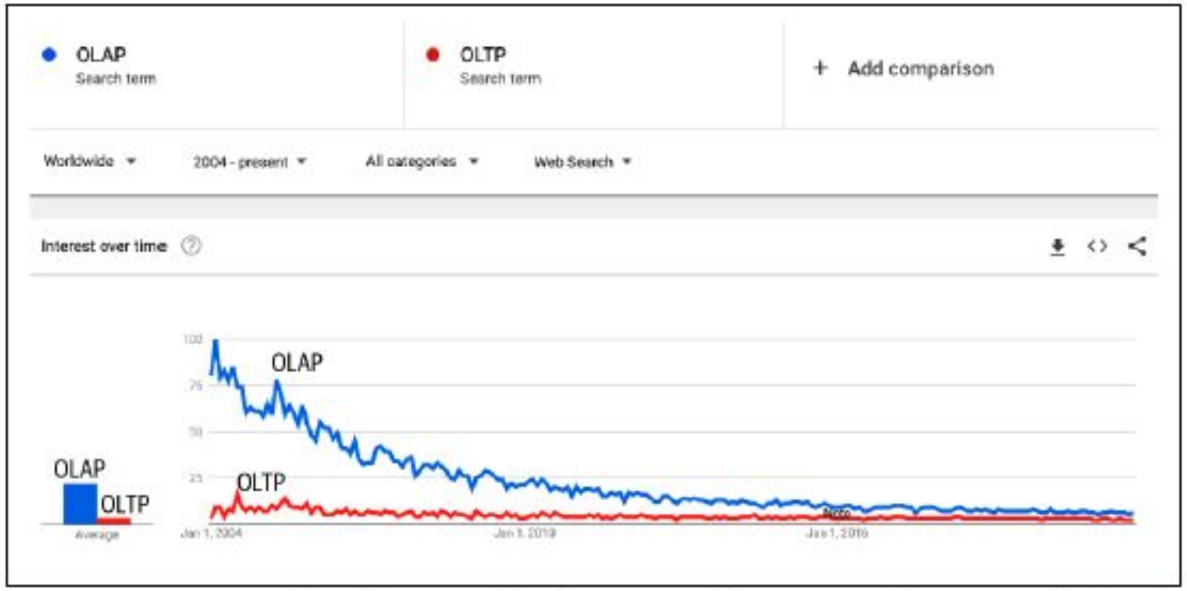


Figure 3-6. OLAP and OLTP are outdated terms, as of 2021, according to Google Trends

Data Flow Modes

1. Databases: Simple, but with latency.
2. Services: Direct communication, microservices architecture.
3. Real Time: Broker (Apache Kafka) for low latency.

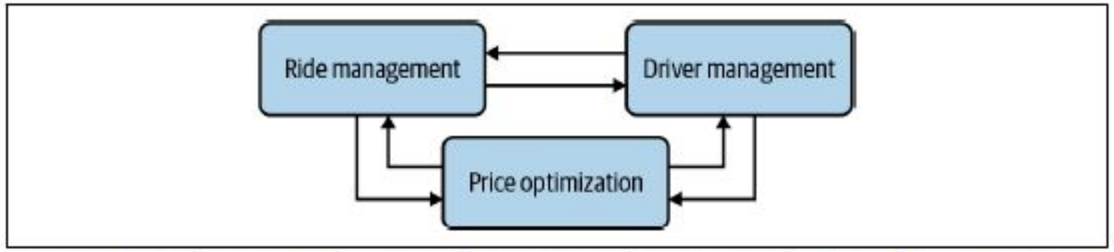
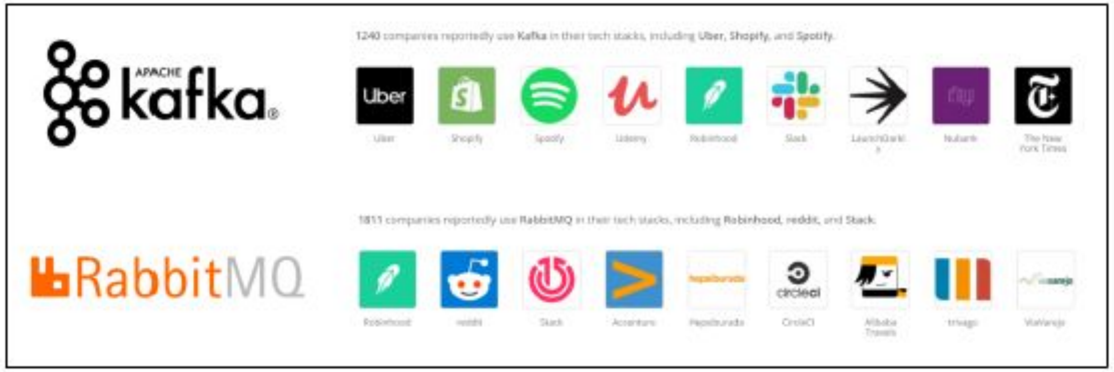


Figure 3-8. In the request-driven architecture, each service needs to send requests to two other services



1240 companies reportedly use **Kafka** in their tech stacks, including **Uber, Shopify,** and **Spotify**.

1811 companies reportedly use **RabbitMQ** in their tech stacks, including **Robinhood, reddit,** and **Slack**.

Figure 3-11. Companies that use Apache Kafka and RabbitMQ. Source: Screenshot from Stackshare

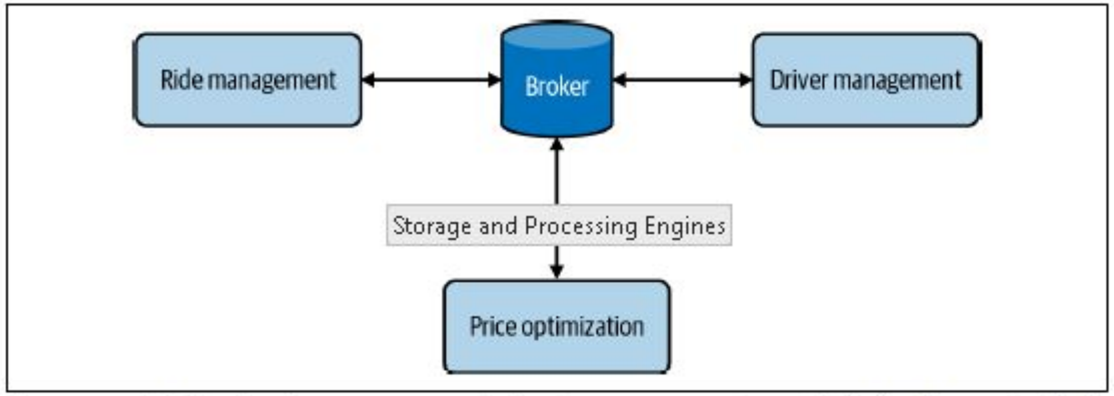


Figure 3-9. With a broker, a service only has to communicate with the broker instead of with other services

Batch Processing vs. Stream Processing

Batch: For historical data, periodic processing.

Stream: Real-time data with Apache Flink.

Conclusion

1. Data fundamentals for ML in production.
2. Choosing the right formats and engines is essential.
3. Efficiency and flexibility for large volumes and velocity of data.