OLTP and OLAP

DATABASE DESIGN



Lis Sulmont
Curriculum Manager

datacamp

Our motivating question:

How should we organize and manage data?

- **Schemas:** *How should my data be logically organized?*
- **Normalization:** *Should my data have minimal dependency and redundancy?*
- **Views:** *What joins will be done most often?*
- **Access control:** *Should all users of the data have the same level of access*
- **DBMS:** *How do I pick between all the SQL and noSQL options?*
- and more!

Our motivating question:

How should we organize and manage data?

- **Schemas:** *How should my data be logically organized?*
- **Normalization:** *Should my data have minimal dependency and redundancy?*
- **Views:** *What joins will be done most often?*
- **Access control:** *Should all users of the data have the same level of access*
- **DBMS:** *How do I pick between all the SQL and noSQL options?*
- and more!

It depends on the intended use of the data.

Approaches to processing data

OLTP

Online Transaction Processing



OLAP

Online Analytical Processing



Some concrete examples

OLTP tasks

- Find the price of a book
- Update latest customer transaction
- Keep track of employee hours

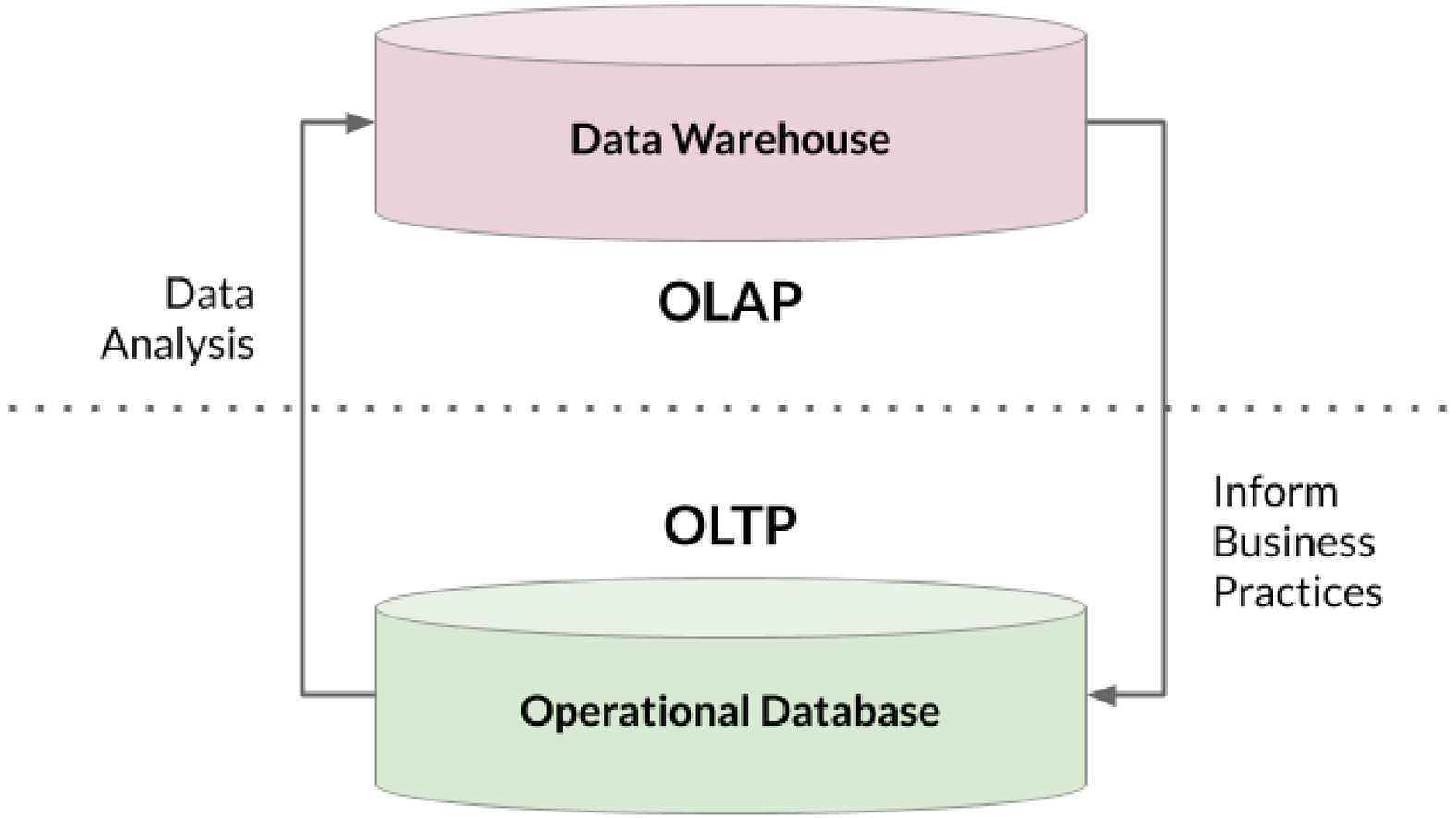
OLAP tasks

- Calculate books with best profit margin
- Find most loyal customers
- Decide employee of the month

OLAP vs. OLTP

	OLTP	OLAP
Purpose	support daily transactions	report and analyze data
Design	application-oriented	subject-oriented
Data	up-to-date, operational	consolidated, historical
Size	snapshot, gigabytes	archive, terabytes
Queries	simple transactions & frequent updates	complex, aggregate queries & limited updates
Users	thousands	hundreds

Working together



Takeaways

- Step back and figure out business requirements
- Difference between OLAP and OLTP
- OLAP? OLTP? Or something else?

Let's practice!

DATABASE DESIGN

Storing data

DATABASE DESIGN



Lis Sulmont
Curriculum Manager

Structuring data

1. Structured data

- Follows a schema
- Defined data types & relationships

_e.g., SQL, tables in a relational database _

2. Unstructured data

- Schemaless
- Makes up most of data in the world

e.g., photos, chat logs, MP3

3. Semi-structured data

- Does not follow larger schema
- Self-describing structure

e.g., NoSQL, XML, JSON

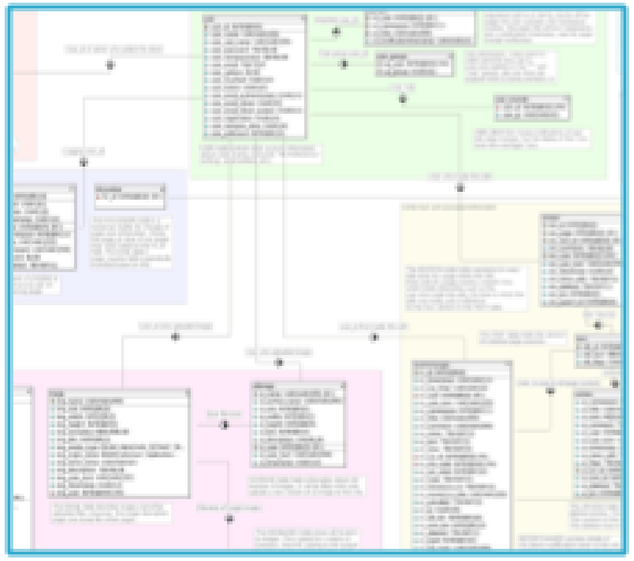
```
# Example of a JSON file
"user": {
     "profile_use_background_image": true,
     "statuses_count": 31,
     "profile_background_color": "C0DEED",
     "followers_count": 3066,
     ...
```

Structuring data

Easier to Analyze



```
<?xml version="1.0"
      encoding="iso-8859-1" ?>
<languages>
  <language id="fr">
    <name lang="fr">Français</name>
    <name lang="en">French</name>
    <name lang="es">Frances</name>
    <name lang="de">Französisch</name>
    <name lang="eo">Franca</name>
  </language>
```



More Flexibility and Scalability

[1] Flower by Sam Oth and Database Diagram by Nick Jenkins via Wikimedia Commons https://commons.wikimedia.org/wiki/File:Languages_xml.png

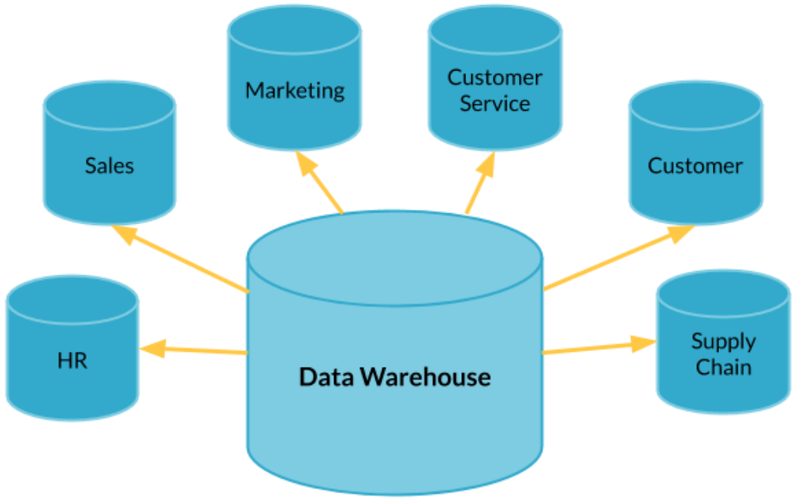
Storing data beyond traditional databases

- **Traditional databases**
  - For storing real-time relational structured data ? **OLTP**
- **Data warehouses**
  - For analyzing archived structured data ? **OLAP**
- **Data lakes**
  - For storing data of all structures = flexibility and scalability
  - For analyzing **big data**

Data warehouses

- Optimized for analytics - **OLAP**
  - Organized for reading/aggregating data
  - Usually read-only
- Contains data from multiple sources
- Massively Parallel Processing (MPP)
- Typically uses a denormalized schema and dimensional modeling





Data marts

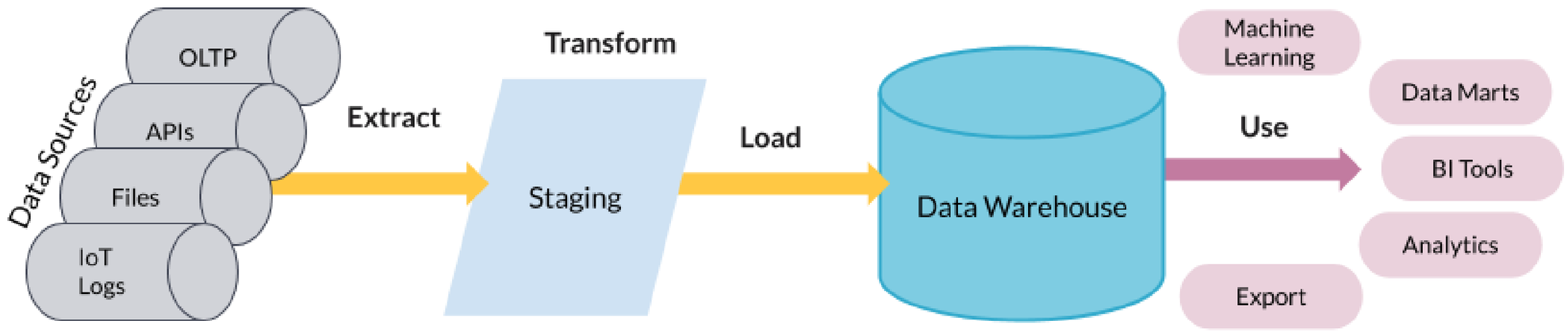
- Subset of data warehouses
- Dedicated to a specific topic

Data lakes

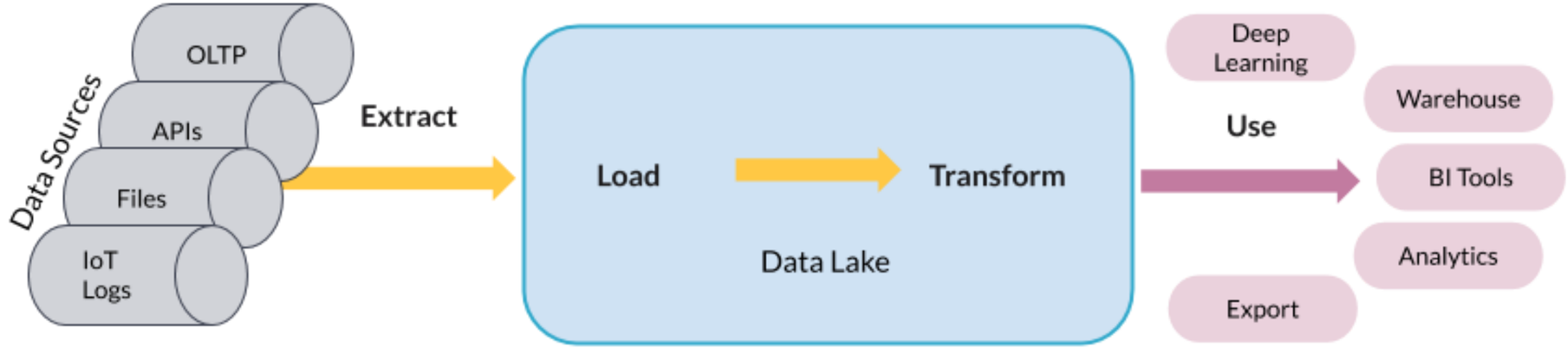
- Store ***all*** types of data at ***a lower cost***:
  - e.g., raw, operational databases, IoT device logs, real-time, relational and non-relational
- Retains all data and can take up petabytes
- Schema-on-read as opposed to schema-on-write
- Need to catalog data otherwise becomes a **data swamp**
- Run **big data analytics** using services such as **Apache Spark** and **Hadoop**
  - Useful for deep learning and data discovery because activities require so much data



ETL


Data Sources
OLTP
APIs
Files
IoT Logs
Extract
Transform
Staging
Load
Data Warehouse
Use
Machine Learning
Data Marts
BI Tools
Analytics
Export

ELT


Data Sources
OLTP
APIs
Files
IoT Logs
Extract
Load
Transform
Data Lake
Use
Deep Learning
Warehouse
BI Tools
Analytics
Export

Let's practice!

DATABASE DESIGN

Database design

DATABASE DESIGN



Lis Sulmont
Curriculum Manager

datacamp

What is database design?

- Determines how data is logically stored
  - How is data going to be read and updated?
- Uses **database models:** high-level specifications for database structure
  - Most popular: relational model
  - Some other options: NoSQL models, object-oriented model, network model
- Uses **schemas:** blueprint of the database
  - Defines tables, fields, relationships, indexes, and views
  - When inserting data in relational databases, schemas must be respected

Data modeling

Process of creating a *data model* for the data to be stored

1. **Conceptual data model**: describes entities, relationships, and attributes

- *Tools:* data structure diagrams, e.g., entity-relational diagrams and UML diagrams

2. **Logical data model**: defines tables, columns, relationships

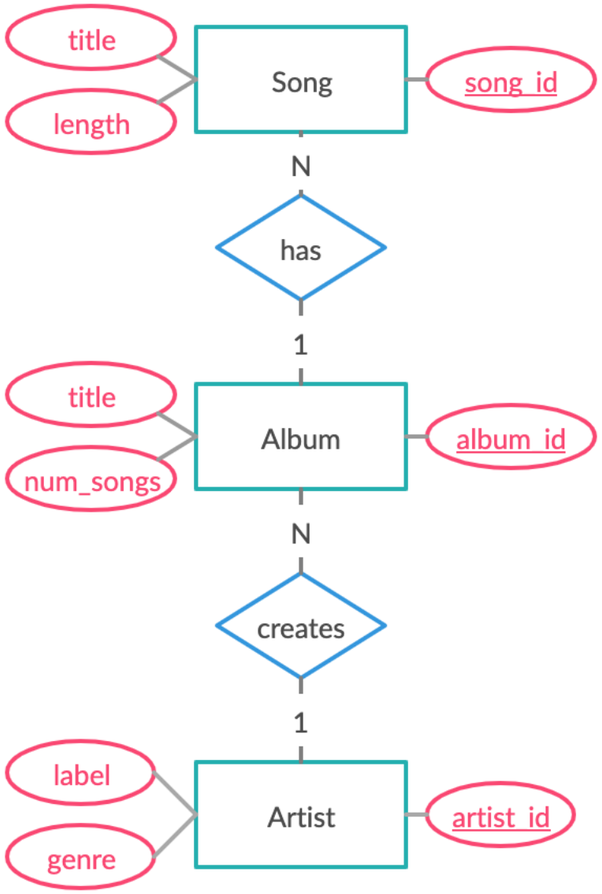
- *Tools:* database models and schemas, e.g., relational model and star schema

3. **Physical data model**: describes physical storage

- *Tools*: partitions, CPUs, indexes, backup systems and tablespaces

[1] https://en.wikipedia.org/wiki/Data_model

Conceptual - ER diagram



Entities, relationships, and attributes

Logical - schema

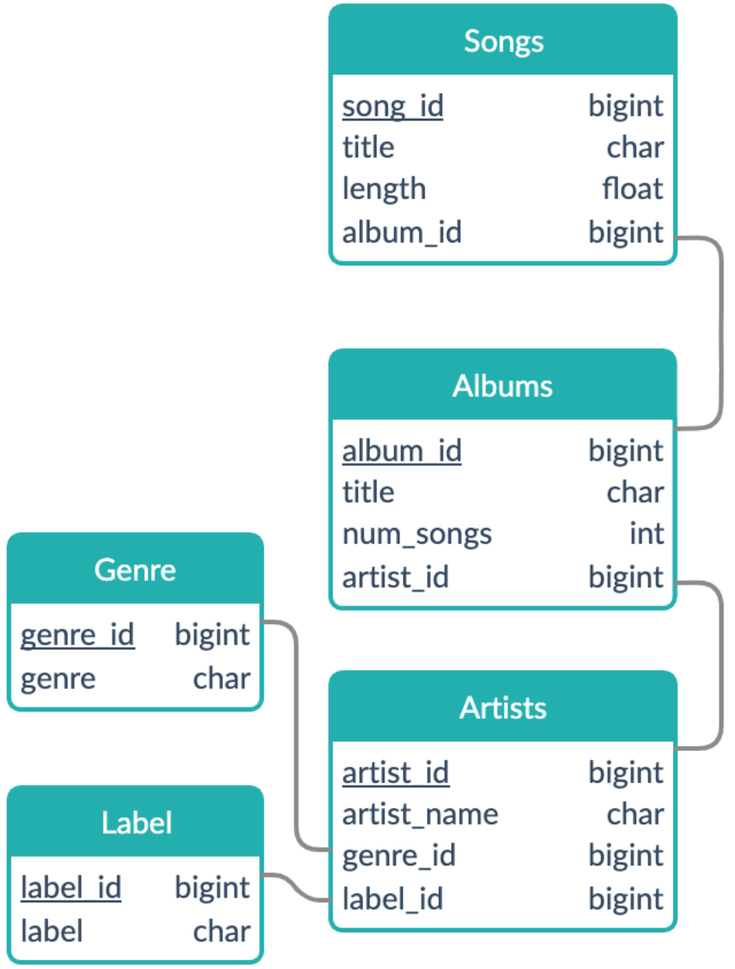


Fastest conversion: entities become the tables

Other database design options



Determining tables



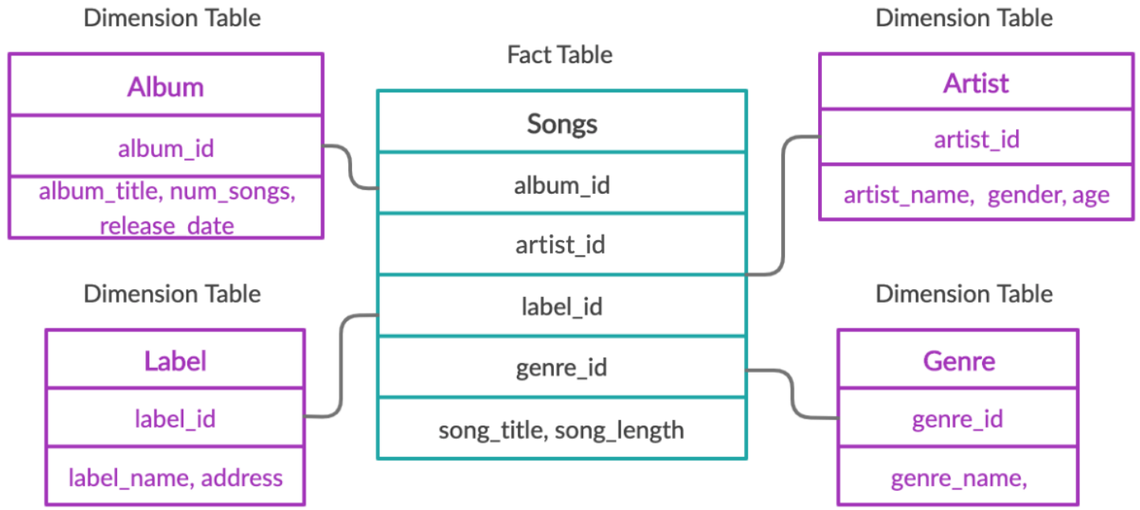
Beyond the relational model

Dimensional modeling

Adaptation of the relational model for data warehouse design

- Optimized for **OLAP** queries: aggregate data, not updating (OLTP)
- Built using the star schema
- Easy to interpret and extend schema

Elements of dimensional modeling



Organize by:

- What is being analyzed?
- How often do entities change?

Fact tables

- Decided by business use-case
- Holds records of a metric
- Changes regularly
- Connects to dimensions via foreign keys

Dimension tables

- Holds descriptions of attributes
- Does not change as often

Let's practice!

DATABASE DESIGN