iykra Data Fellowship

IYKRA
Data Fellowship Program

Processing, Storing, and Organizing Data

by (Bangun Sasongko)

Trainer Profile

Bangun W. Sasongko

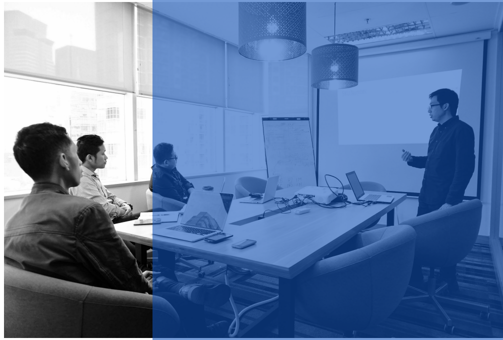
Data Engineer at BRI
Ex. Devops Engineer
Bachelor of Physics, Electrical and Instrumentations
@sasongkobgn
sasongkobgn@gmail.com

Table of Content

Content
OLTP vs OLAP
Storing Data
Database Design

About this Course

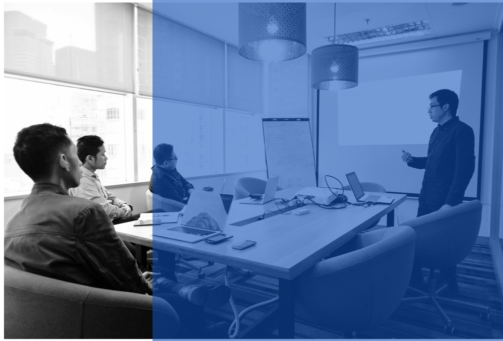
This course will equip you with the fundamentals of database design. We'll begin by exploring two key data processing methods, OLTP and OLAP. You'll also gain a solid understanding of various data storage formats and delve into the fundamentals of data modeling
OLTP vs OLAP
Storing Data
Database Design



The Objectives

By the end of this course, you will be able to:

1. Differentiate between the two main data processing approaches: OLTP (Online Transaction Processing) and OLAP (Online Analytical Processing).
2. Identify and understand the different formats for data storage.
3. Grasp the core concepts of data modeling.
4. Apply these foundational concepts to design basic databases.



Data Fellowship

OLTP vs OLAP

Our motivating questions:

How should we organize and manage data?

It depends on the intended use of the data.

Our motivating questions:

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!

Approaches to processing data

OLTP
Onl**ine** **T**ransaction **P**rocessing

OLAP
Onl**ine** **A**nalytical **P**rocessing

Some concrete examples

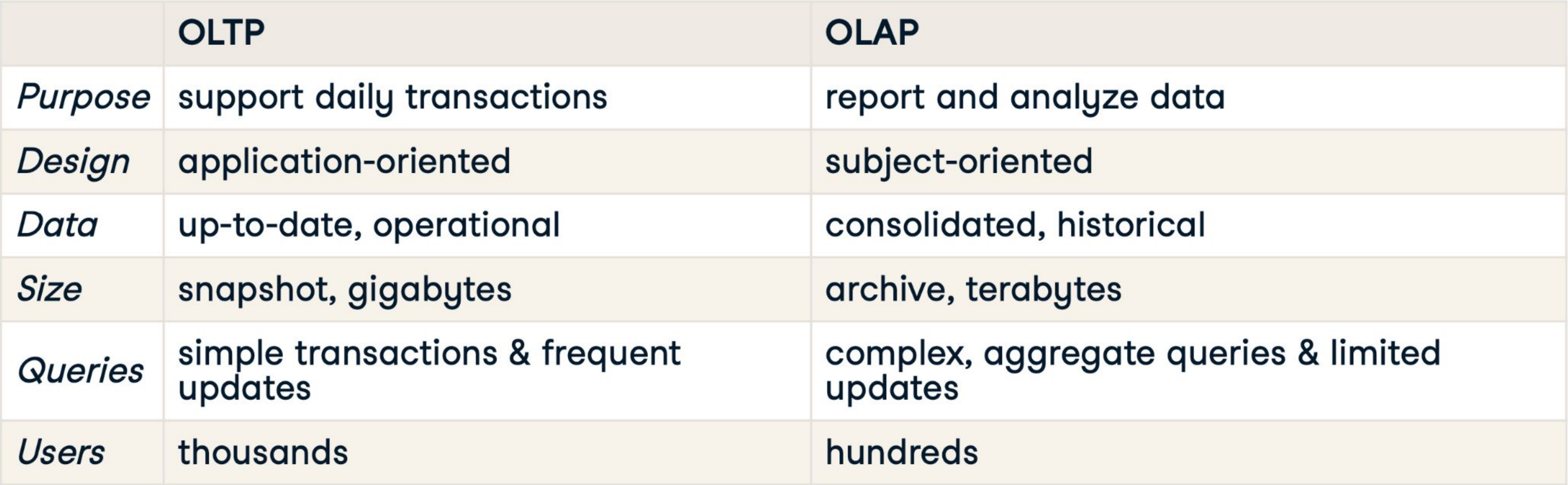
OLTP task
On**l**ine **T**ransaction **P**rocessing

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

OLAP task
On**l**ine **A**nalytical **P**rocessing

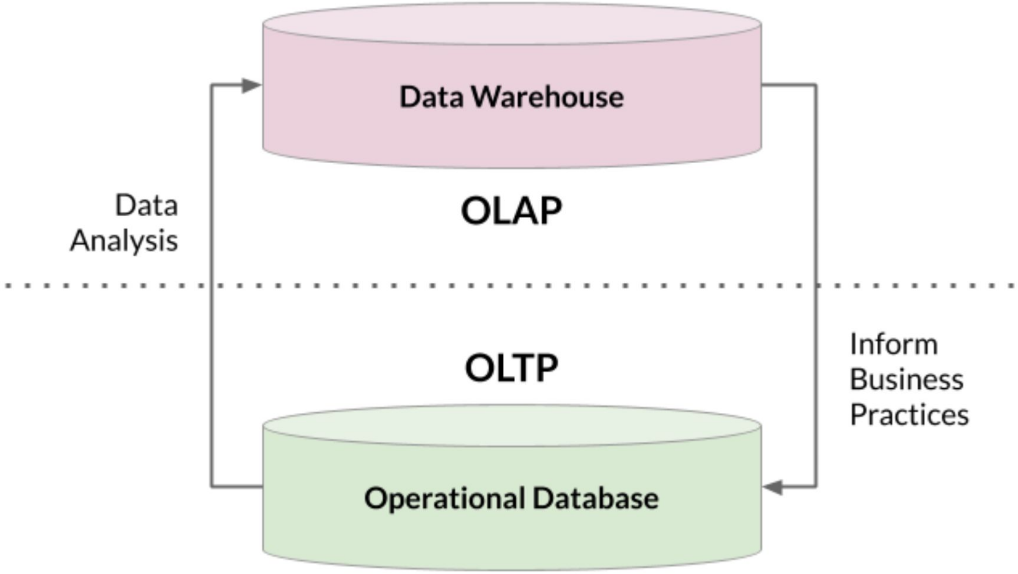
- Calculate books with best profit margin
- Find most loyal customers
- Decide employee of the month

OLTP vs. OLAP



	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?

Data Fellowship

Storing data

Structuring data

1. **Structured data**

- Follows a schema
- Defined data types & relationships

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

2. **Structured data**

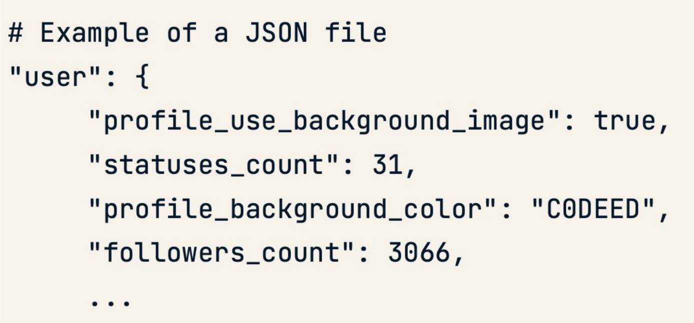
- Does not follow larger schema
- Self-describing structure

e.g., NoSQL, XML, JSON

2. Unstructured data

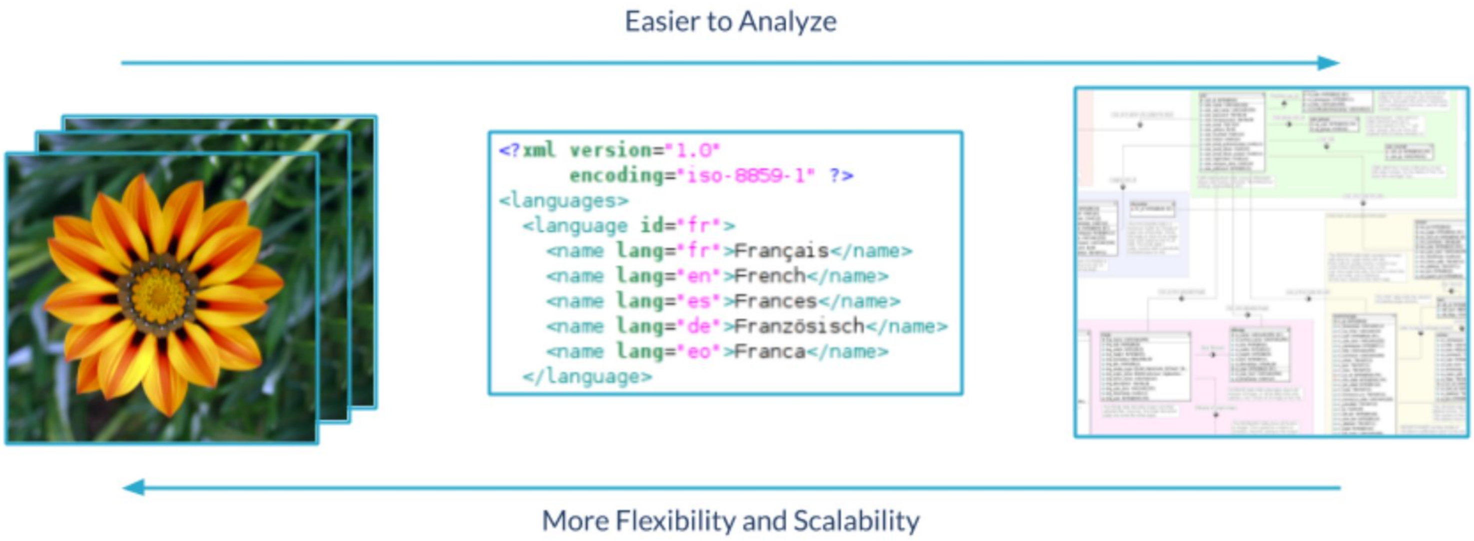
- Schemaless
- Makes up most of data in the world

e.g., photos, chat logs, MP3



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

Structuring data



```
<?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>
```

[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

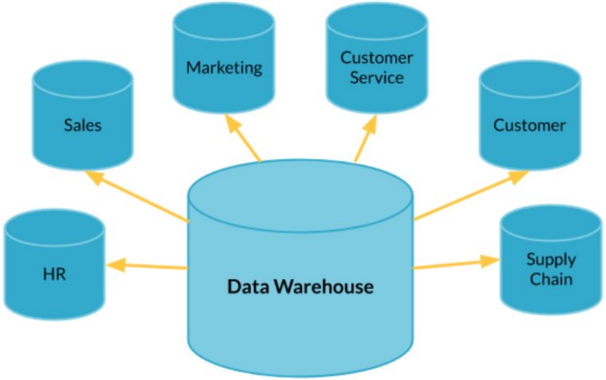
- **Data warehouses**
  - For analyzing archived structured data ? OLAP
- **Data lakes**
  - For storing data of all structures = flexibility and scalability
  - For analyzing big data

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

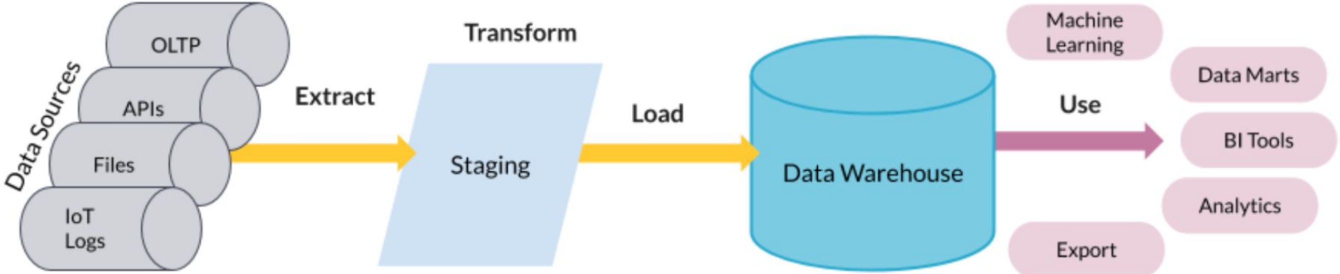


Data lakes

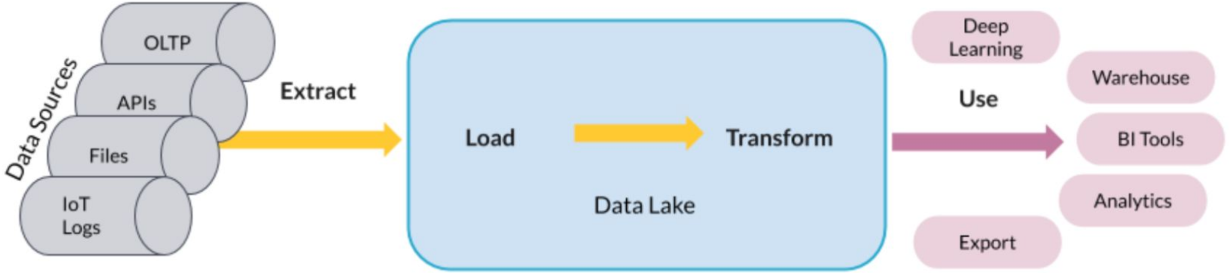
- Store ***all*** types of data at a **lower cost**:
  - e.g., raw, operational databases, IoT device logs, real-time, relational and non-relational

- 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

Data Fellowship

Database design

What is database design?

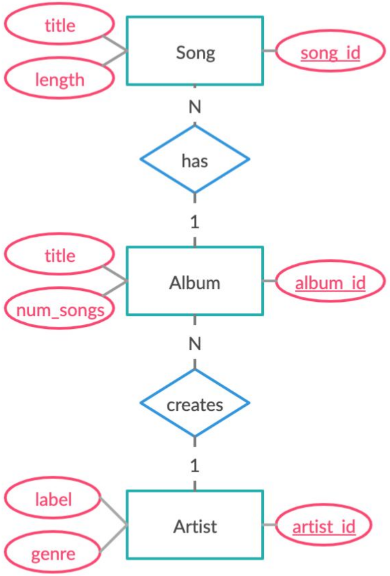
- Determines how data is logically stored
  - How is data going to be read and updated?
- Uses **database models**: high-level specifications for database structure
  - Most popular: relational model
  - Some other options: NoSQL models, object-oriented model, network model
- Uses **schemas**: blueprint 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

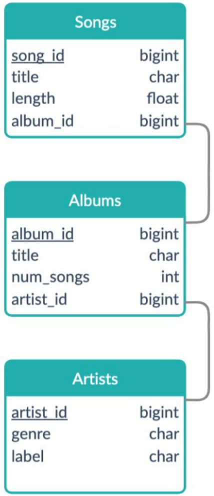
1. **Conceptual data model**: describes entities, relationships, and attributes
   - *Tools*: data structure diagrams, e.g., entity-relational diagrams and UML diagrams
2. **Logical data model**: defines tables, columns, relationships
   - *Tools*: database models and schemas, e.g., relational model and star schema
3. **Physical data model**: describes physical storage
   - *Tools*: partitions, CPUs, indexes, backup systems and tablespaces

Conceptual - ER diagram



Entities, relationships, and attributes

Logical - schema



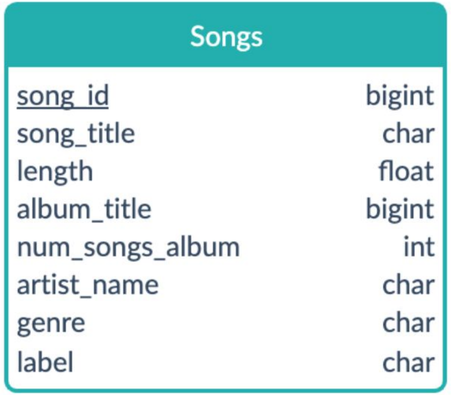
Songs	
song_id	bigint
title	char
length	float
album_id	bigint

Albums	
album_id	bigint
title	char
num_songs	int
artist_id	bigint

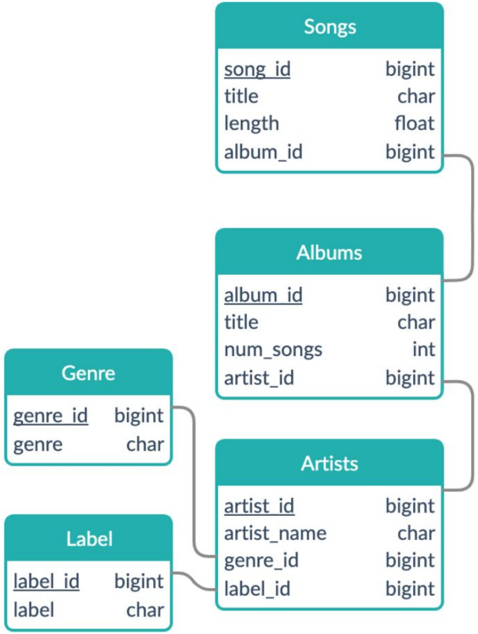
Artists	
artist_id	bigint
genre	char
label	char

Fastest conversion: entities become the tables

Other database design options



Songs	
song_id	bigint
song_title	char
length	float
album_title	bigint
num_songs_album	int
artist_name	char
genre	char
label	char



Songs	
song_id	bigint
title	char
length	float
album_id	bigint

Albums	
album_id	bigint
title	char
num_songs	int
artist_id	bigint

Genre	
genre_id	bigint
genre	char

Artists	
artist_id	bigint
artist_name	char
genre_id	bigint
label_id	bigint

Label	
label_id	bigint
label	char

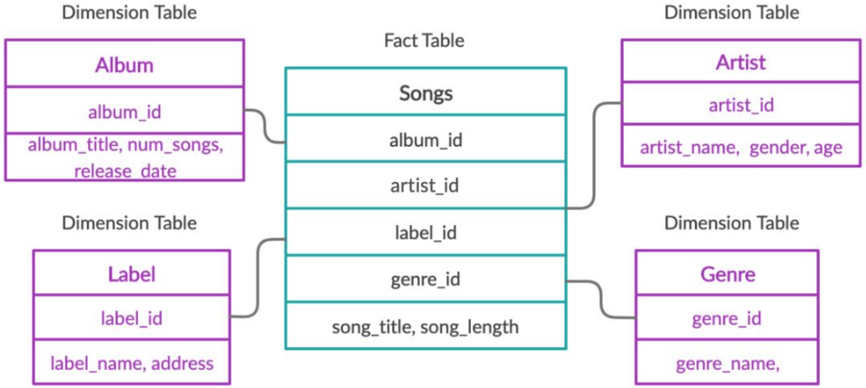
Determining tables

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



Dimension Table

Album

album_id

album_title, num_songs, release_date

Fact Table

Songs

album_id

artist_id

label_id

genre_id

song_title, song_length

Dimension Table

Artist

artist_id

artist_name, gender, age

Dimension Table

Label

label_id

label_name, address

Dimension Table

Genre

genre_id

genre_name,

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

References

Datacamp : database design
A cloud guru : database design

iykra Data Fellowship

Thank you!

nothin’ is impossible until its done