Why ?

- operational data keeping (OLTP) Online Transaction Processing
- Analytical decision making (OLAP) Online Analytical Processing

---

Requierments

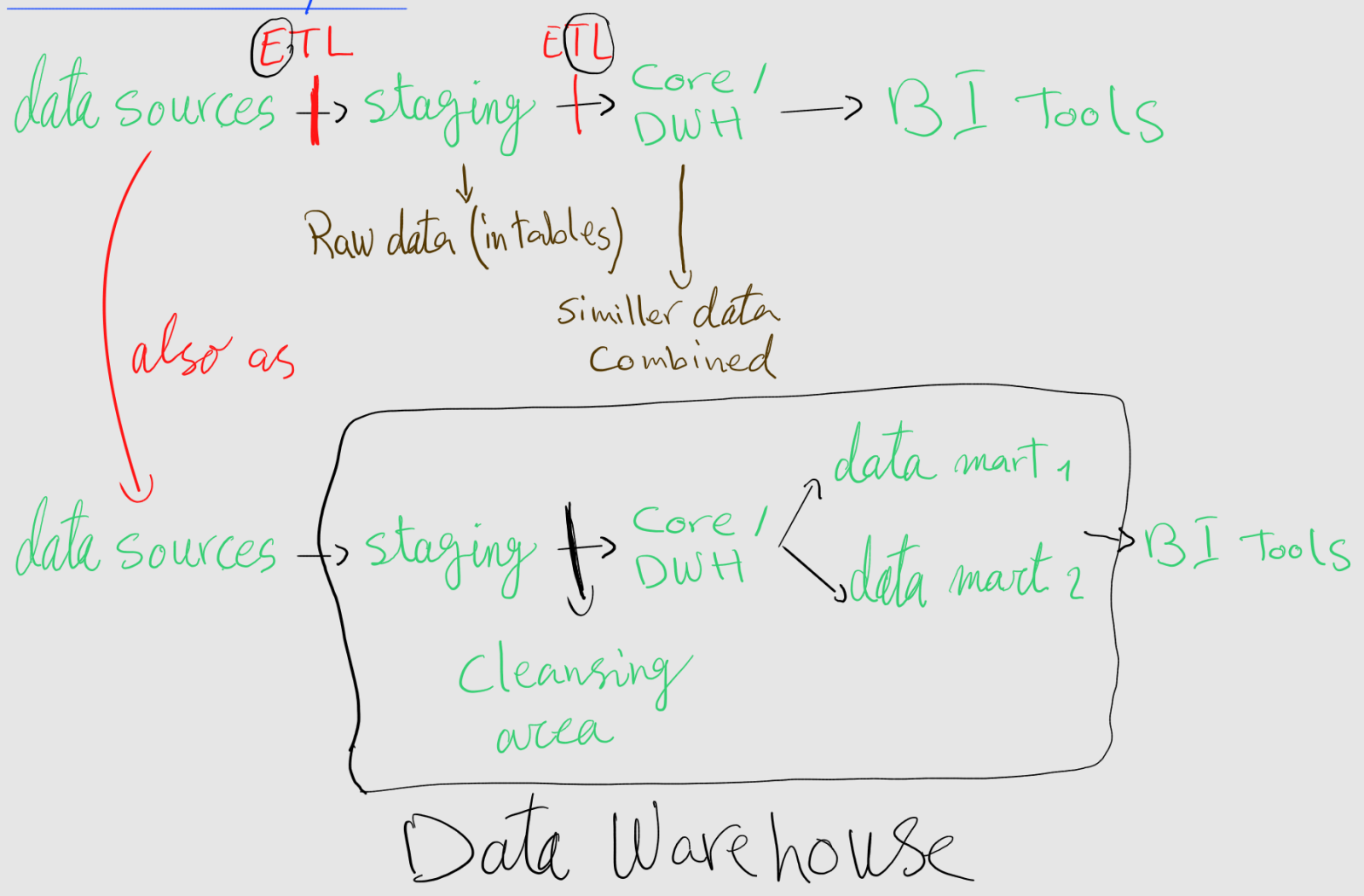
- OLTP:
  - one record at a time
  - Data input
  - no long history
- OLAP:
  - Thousands of records at a time
  - Fast query performance
  - Historical context
  - Usability

Why data lake ?

- stores raw data without process
- " Big data
- unstructured
- not ready to be used

DWh architecture →

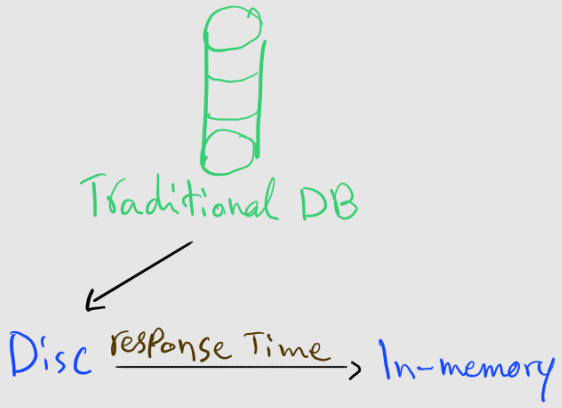
DWH layers

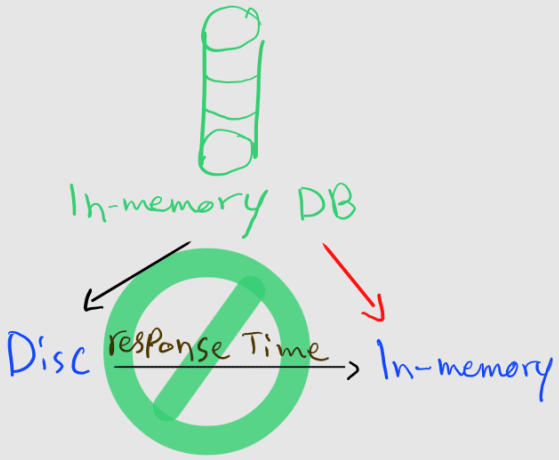


staging area

- quickly extract
- move the data into relational database
- start transformations from there
- landing zone extracted data
- data in tables and on a separate database
- Temporary or Persistant

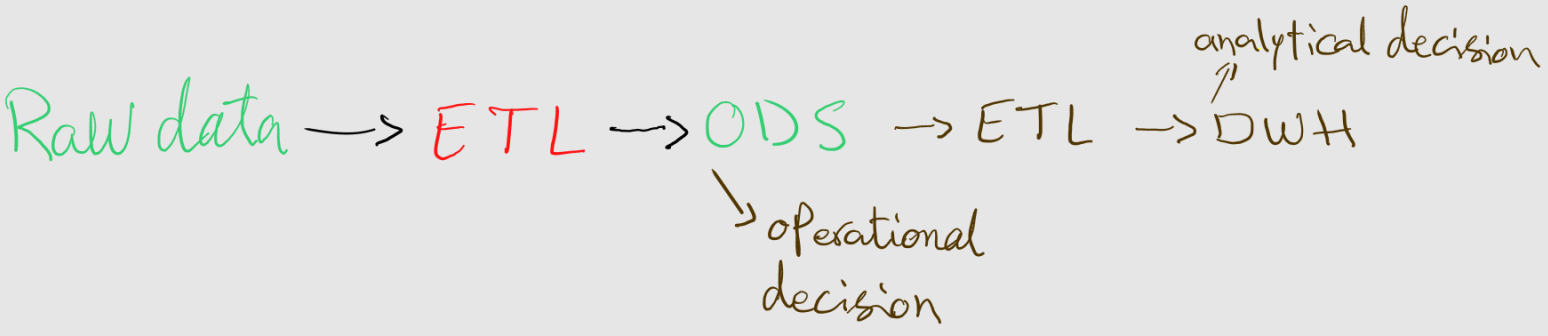
In memory databases ——> used usualy with data marts





Operational data storage (ODS) (uncommon)

↳ used for operational decision making



Dimentional modeling ↴

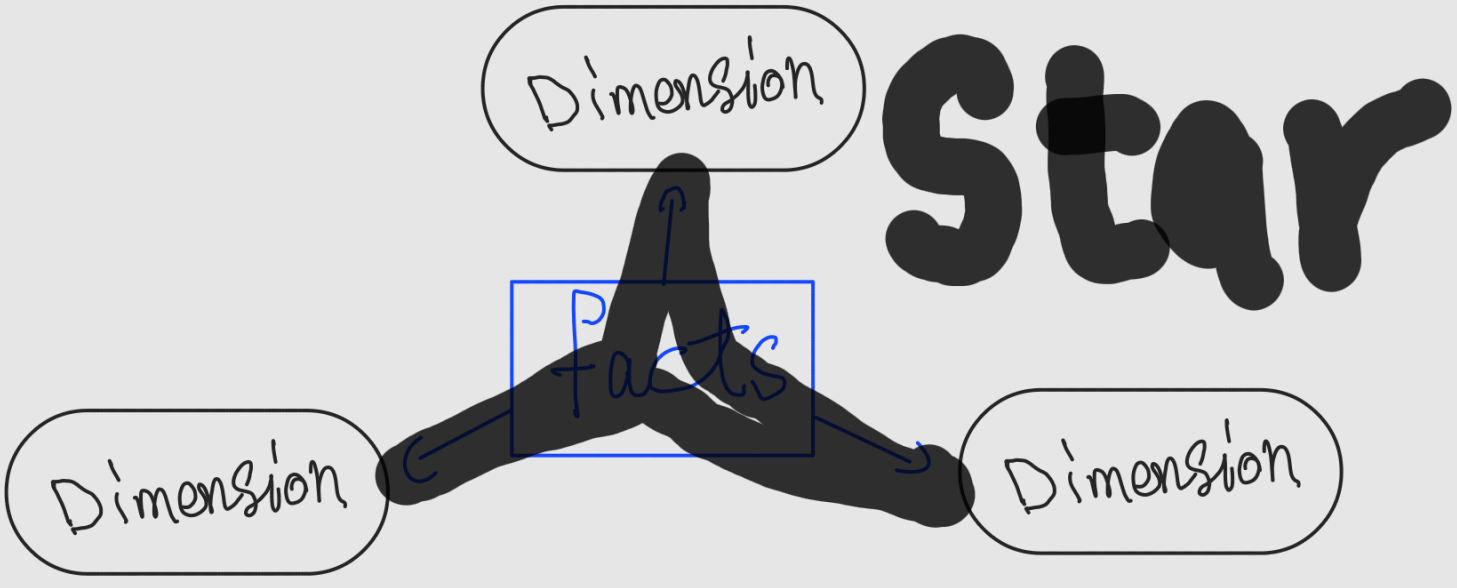
↳ method of organizing data (in data warehouse)

↳ facts

↳ Measurements like Profit

↳ Dimensions

↳ context like category



↳ Unique technique of structuring data

↳ Commonly used in DWh

↳ optimized for faster data retrieval

↳ oriented around Performance & usability

↳ Designed Reporting / OLAP

- Facts

- foundation of DWH
- key measurments → يعني هو الحاجة الاساسية اللي مجمعة الـ Ids وحواليها الـ Dim_Tables اللي فيها تفاصيل البيانات (عصب الـ Tables)
- aggregated and analyzed
- aggregatable (numerical values)

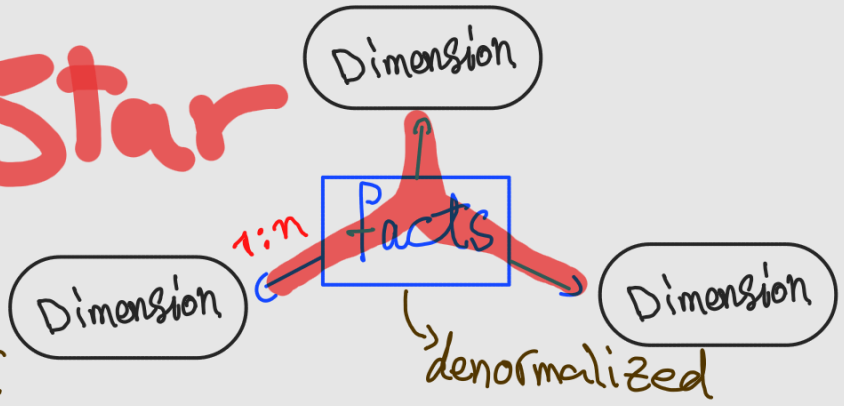
- fact table : PK, FK & facts
- Grain : the most atomic level facts are defined

- Dimensions

- categorizes facts
- Supportive & descriptive
- filtering, Grouping & Labeling
- non-aggregatable
- (more) static

- Dimension table : PK , Dimension , (FK)
- People , Products , Places , time

Star schema



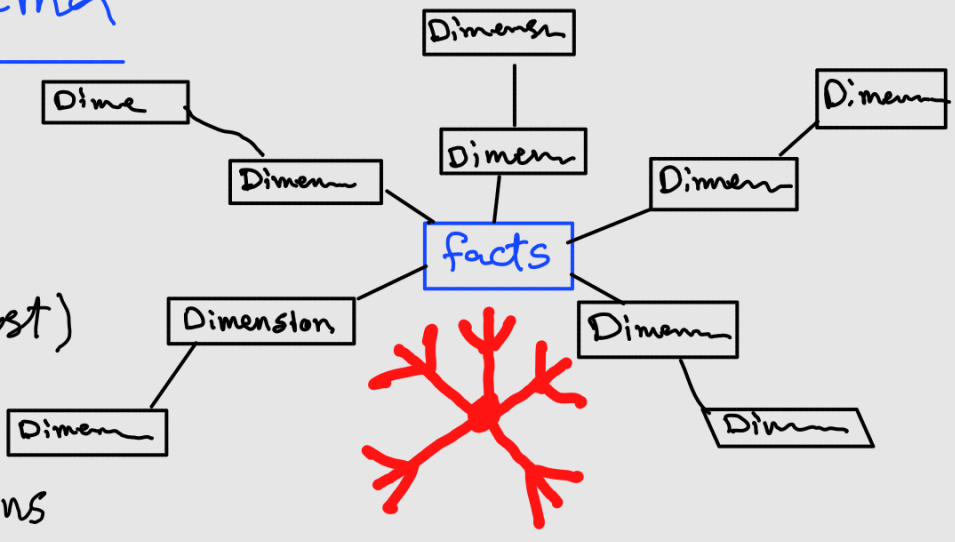
↳ Denormalized

- There is data redundancy
- optimized to get data out
- query Performance (read)
- User experience

↳ Normalized

- Technique to avoid redundancy
- minimizes storage
- Performance (write / update)
- many tables → many Joins necessary

Snowflake schema



↳ more normalized

- Advantage
  - ↳ less space (storage cost)
  - ↳ less redundant data
  - ↳ solves write slow downs
- Disadvantage
  - ↳ more complex
  - ↳ more joins (more complex sql queries)
  - ↳ less performance Data marts

Fact table types

1- Transactional fact table

- 1 row = measurments of 1 event / transaction
- 1 transaction defines the lowest grain
- FK * الـ row الواحد يعبر عن معلومة واحدة وضمنها مقاييس واحد والباقي
- Tend to have a lot dimensions associated
- Can be enormous in size

2- Periodic snapshot table

- 1 row = summarizes measure of many events / transaction
- summarized of standard period (1 day, 1 week, etc)
- lowest period defines the grain
- FK * يعبر عن معلومات عن مدة معينة وكلها مقاييس ومتغيرات
- Tend to be not as enormous in size
- Tend to have a lot of facts and fewer dimensions associated
- no events = null or 0

3- Accumulating snapshot fact table

- 1 row = summarizes measure of many events / transactions
- summarized of lifespan of one process (order fulfillment)
- definite beginning & definite ending (& steps between)

* تجميعة لكل معلومات عن حدث معين زي توصيل الطلبات

- Multiple Date / Time foreign keys (for each process step)
- Date / Time keys associated with role-play dimension

4- Factless fact table

* Table بمعلومات بين analysis لـ معلومات اي معلومات على

employee registration زي

• Steps to create a fact Table

1- Identify business process for analysis

↳ Sales, order processing

2- Identify the grain → what one record in a fact table represents

↳ Transaction, order, order lines, Daily, Daily + Location

3- Identify dimensions that are relevant

↳ What, when, where, How and why

↳ Time, location, Products, costumers

4- Identify facts for measurements

↳ defined by the grain & not by specific use-case

- Natural Keys
  - ↳ come out of the source system
- Surrogate key
  - ↳ Artificial keys
  - ↳ more beneficial
    - integer number
    - _PK or _FK suffix
    - created by the database / ETL Tool

• Dimension tables

- Hierarchies in dimensions
  - ↳ preferred to be denormalized / flattened

1- Date dimension

* Replace nulls with descriptive values

2- Conformed dimensions

↳ Shared by multiple fact tables / stars

↳ used to compare facts across different fact table

* فيه Dimension Table مشتركة مابين اتنين facts table وبيقوم shared attributes

3- Degenerate dimension

↳ dimension key without an associated dimension

* هي لوحدها بيعبر عن معنى او مقياس بس ملهاش Table موصللها ؛ زي الـ order Id

4- Junk dimensions (Transactional indicator dimension)

* Table او Attributes بتعبر عن مقياس Yes/No او ليها نوعين بس في الاخر، فا بتتحط في تابل لوحدها

5- Role-Play dimensions

↳ Dimension that is referenced multiple times by a fact

* attribute واحد زي الـ Date مثلا بيعبر عن اكتر من حاجة في نفس الوقت

- Slowly changing dimensions

Type 0 : Retain Original

- There won't be any changes
- Date Table (expect for holidays etc.)
- original
- very simple and easy to maintain

Type 1 : Overwrite

- old attributes are just over written
- very simple
- No fact Table needs to be modified

Problem:

- History is lost
- insignificant changes
- might affect / break existing queries

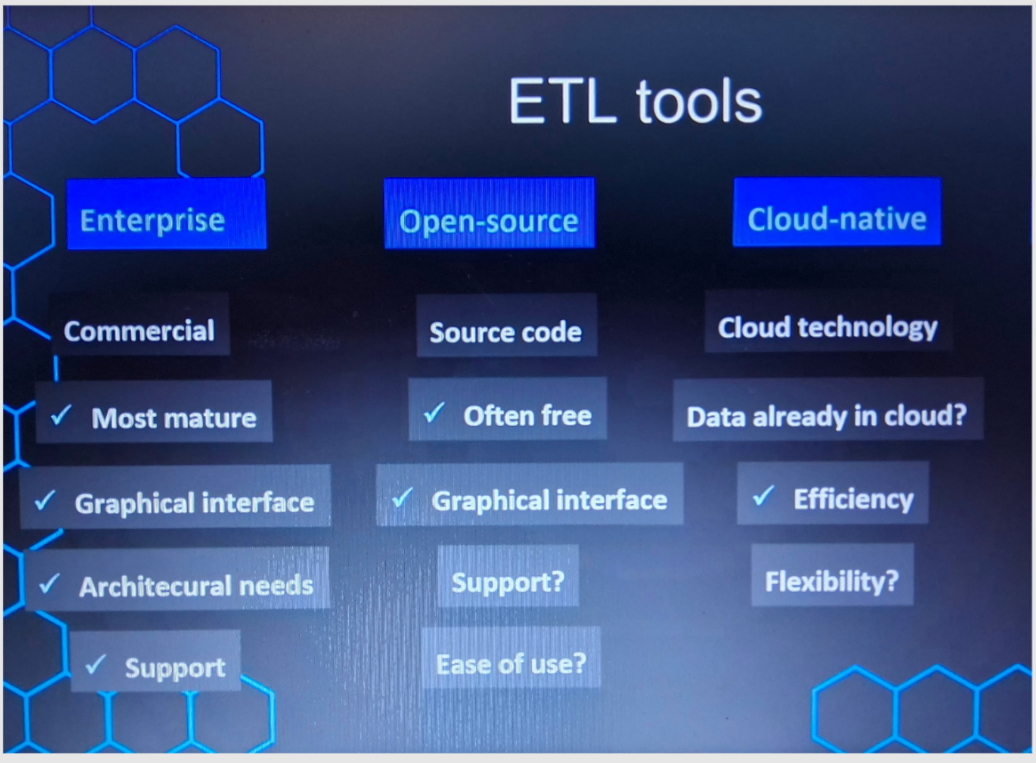
Type 2 : New row

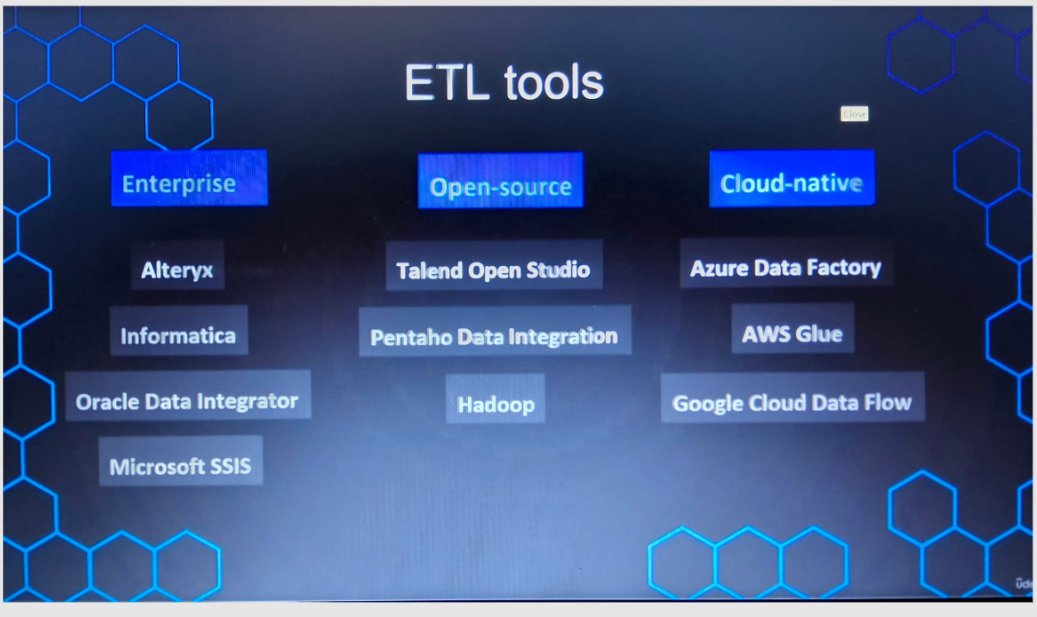
- Changes reflected with history

Type 3 : Additional attributes

- instead of adding a row, we add a column
- Typically used for significant changes at a time

ETL





- ETL v.s. ELT

- Query optimization and database internals