MBA USP ESALQ

DATA ENGINEERING I

Prof. Dr. Jeronymo Marcondes

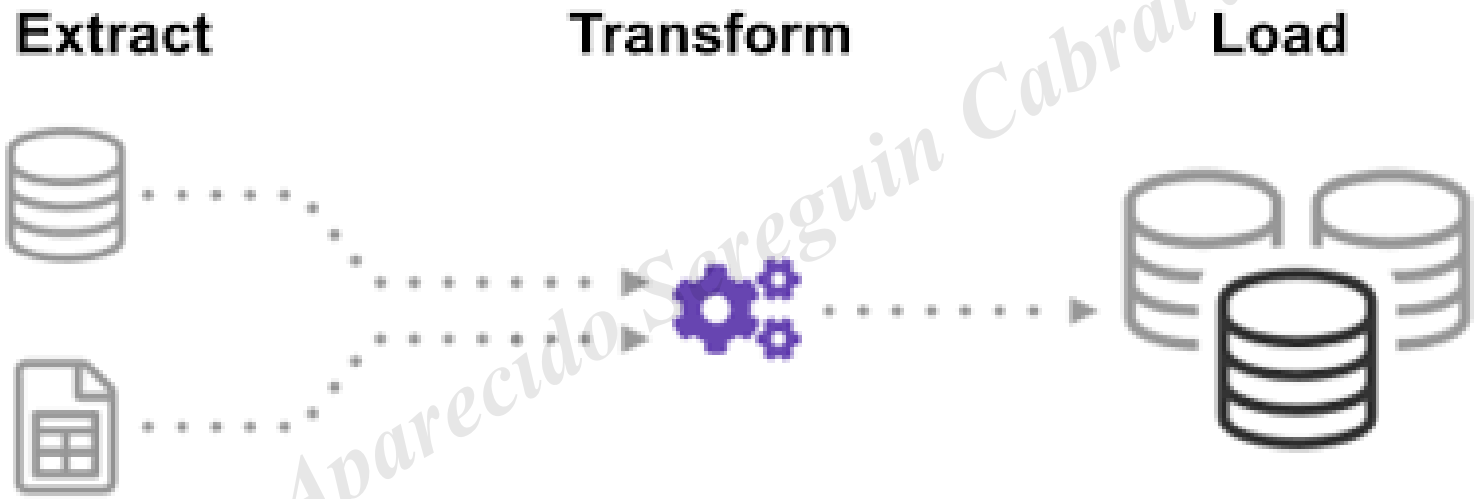
Introduction

Why to study Data Engineering?

Data Engineering x Data Science.

What is Data Engineering?

ETL



Requirements of large companies

A professional of data science must:

- Know about the structures of databases.
- Know how a process of the ETL works.
- Understand about modeling of databases.
- Understand how the use of data in the production works.
- **Know SQL.**

Our objective

Introduction to the data structure.

Relational database.

SQL.

ERD Model – construction and interpretation.

The model and the relational algebra.

Data

- Data x Information.

- What is a database?

It is a data collection, which describes, typically, the activities and relationships of one or more organizations.

Example: MBA USP.

DBMS

Database Management Systems:

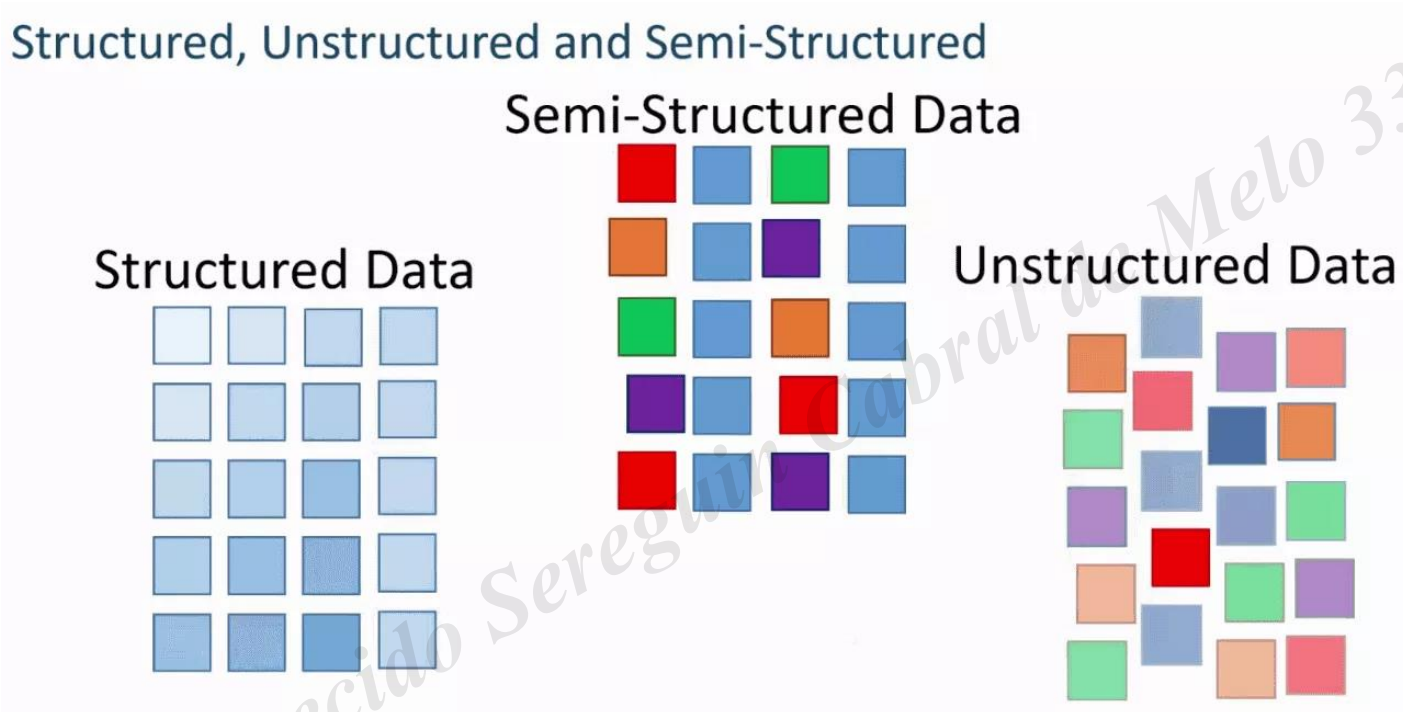
Software planned to help the maintenance, organization, and collection of existing data in a database.

Example: MySQL.

Data Structures

The data we can use are divided into:

- Structured Data.
- Semi-structured Data.
- Unstructured data.



Source: https://www.astera.com/pt/tipo/blog/dados-semiestruturados-e-n%C3%A3o-estruturados-estruturados/

Structured the data that have well-defined formats, such as those extracted from spreadsheets or relate databases in SQL.

Semi-structured – Similar to structured data, but not obedient in the totality regarding the form. In this line are the records of languages based on HTML and XML.

Unstructured or NoSQL - do not have a specific format, these are data collected in their original form, such as a text, a video, an email fragment, a system log or a simple photo.

Structured Data

CPF	Name	Grade
x	Zé das couves	10
y	Maria das desgraças	2
h	Silvio Santos	5

Semi-structured data

```
[
 {
   "CPF": "x",
   "Name":  "Zé das couves",
   "Grade":  "10",
   "Telephone":    "It's none of your business"
 },
 {
   "CPF": "y",
   "Name":  "Maria das desgraças",
   "Grade":  "2"
 },
 {
   "CPF": "h",
   "Name":  "Silvio Santos",
   "Grade":  "5",
   "Income":  "Muito alta"
 }
]
```

Unstructured Data

Relational DBMS

Our focus will be on Relational DBMS

Advantages in the use of a DBMS:

- Independence.
- Efficiency.
- Integrity and security.
- Simplified data management.
- Access control.

Data model

- Data "stored" in the database according to the model. The DBMS will allow us to look at this model and make consultations according to the pre-established logic.
- The description of the data in terms of models is called **SCHEMA**. As example below:

Students (CPF: string, Name: string, Grade: Integer)

Type of data

- The types of data are classified in different categories and allow N formats. We will present only the most common ones.
- Integer Example: 1, 2, etc.
- Float. Example: 0.10, 10.25, etc.
- String. Example: “Good morning", “my name is", etc.
- Date. Example: 2021-01-01.
- VARCHAR and CHAR case.

Data model

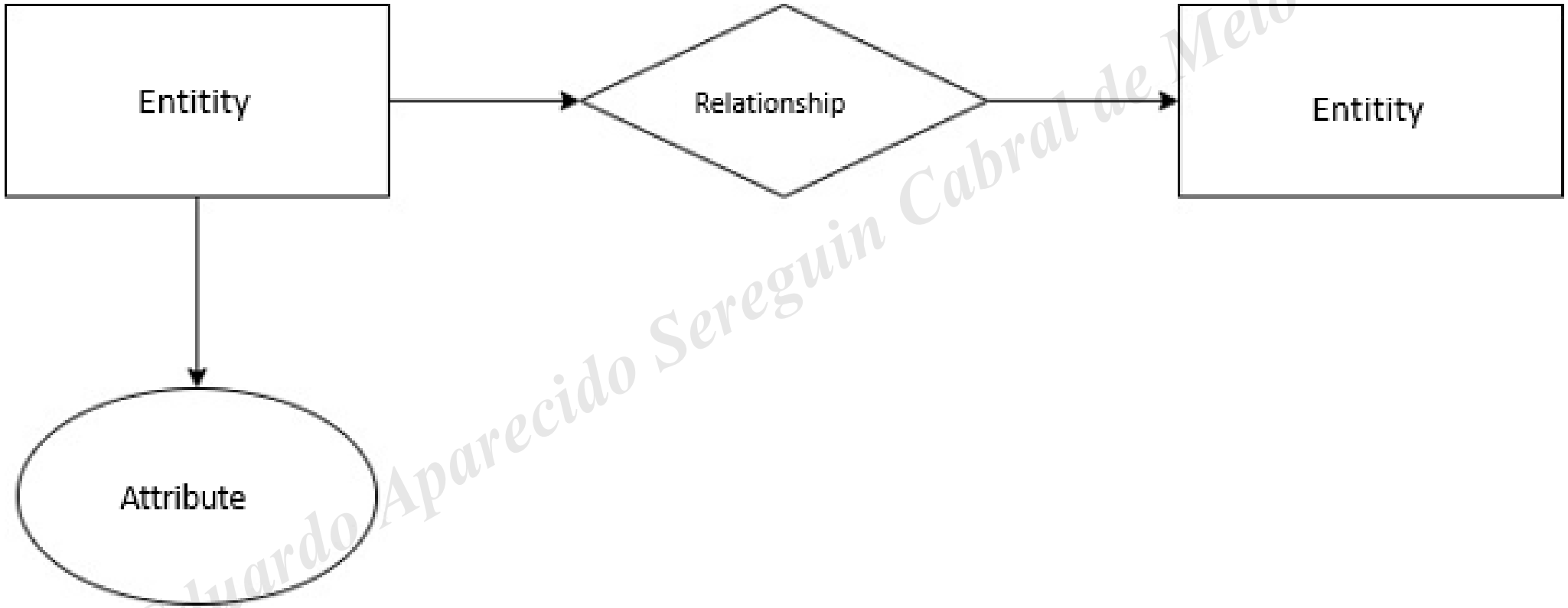
Students (CPF: string, Name: string, Grade: Integer)

- This indicates to us that it is a table with three fields.
- Relational model implies that each register is unique.
- Integrity Constraints!
-

Level of Abstraction

Conceptual model

- The highest level.
- Closer to the reality of the business.
- It describes the relationships between the entities present in a database.



ERD Definitions

- Entity: Something that can be defined and can have data stored about it - such as a person, an object, concept or event. Think on entities as nouns. Examples: a customer, student, car or product.

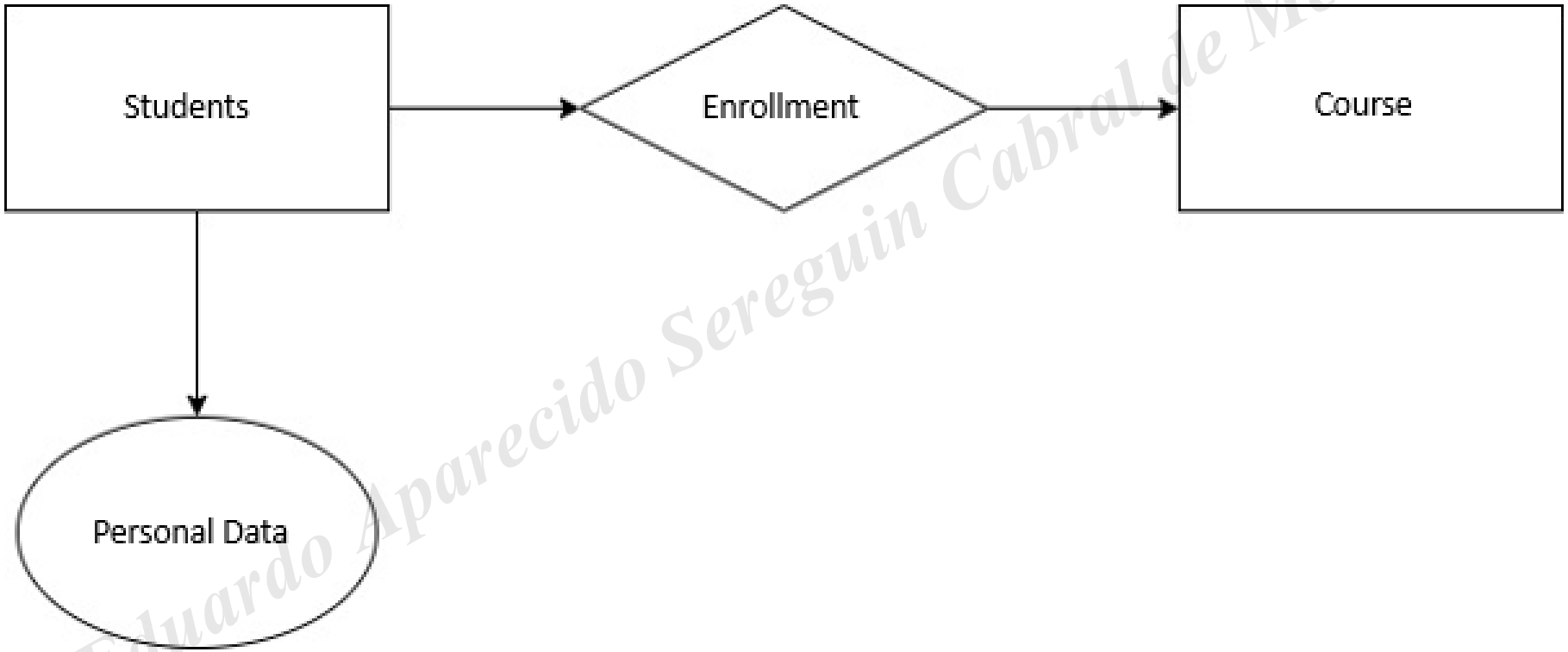
ERD Definitions

- Relationship: How entities act upon each other or are associated with each other. Think on relationships as verbs. For example, the student can sign up in a course. The two entities would be the student and the course, and the relationship described is the act of signing up, so, connecting the two entities.

ERD Definitions

- Attribute: The property or characteristic of an entity, often represented by an oval or circle.

ERD Example

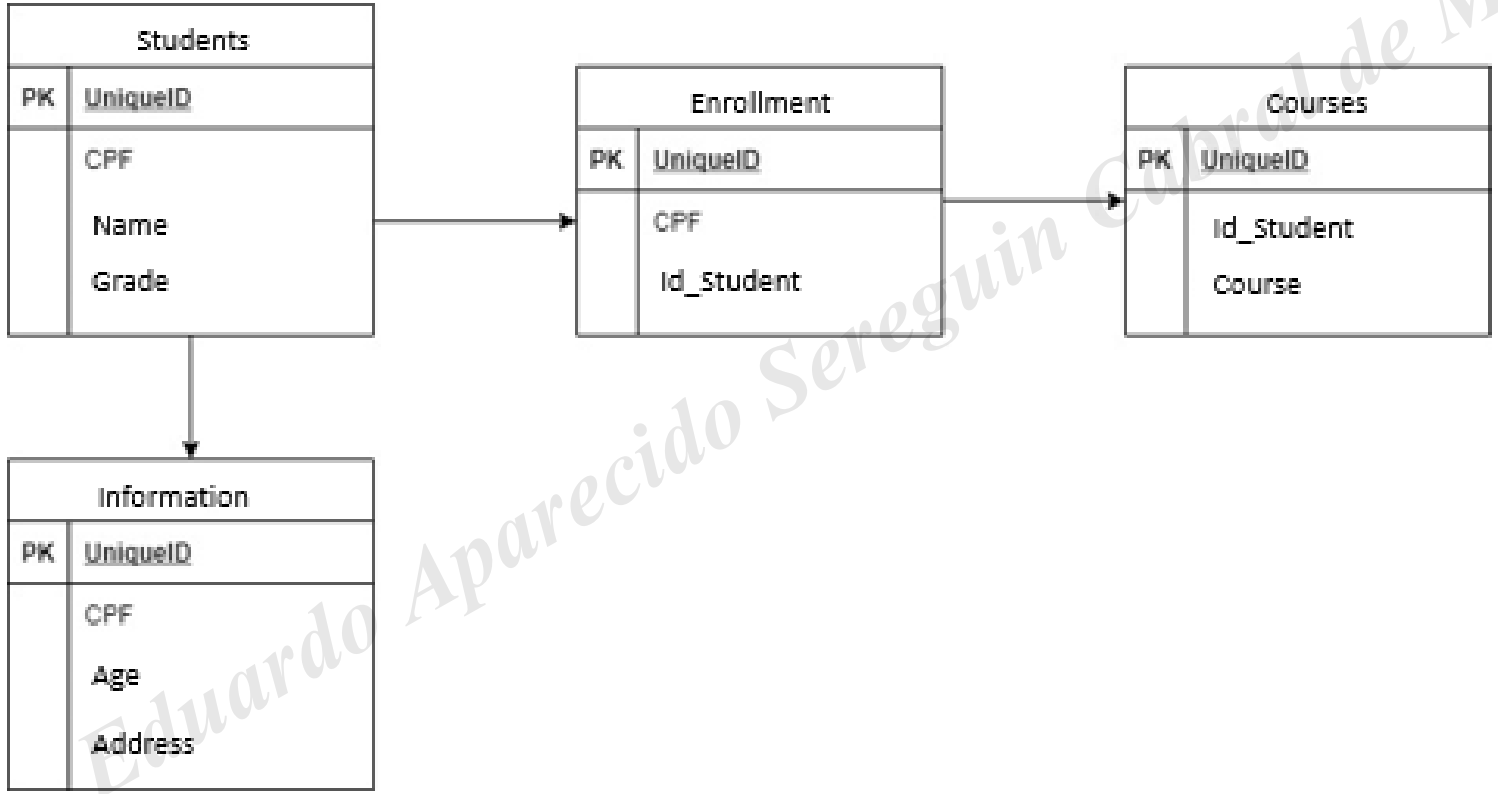


Level of Abstraction

Logical Model

- As the data will be arranged effectively in tables in the database.
- It takes into account limitations of the bank and DBMS.
- Defines the primary and foreigner keys, and integrity constraints.

Logical Schema



Level of Abstraction

Physical Model

- Implementation How to insert the data and create tables and all the schema.
- The lowest level.
- How the data will be stored.
- Restoration methods, backup.

Query

Given the existence of a database, we may ask:

- How many students are enrolled in a course?
- How many courses are active?
- What is the average age of the students?
- What is the average age of students in a certain course?

SQL enters.

SQL

- DML – data manipulation language.
- Universally accepted.
- Proper to use relational algebra

Introduction to SQL

SQL

- Structured Query Language.
- Origin – IBM.
- We do not need the way to get in the result – we define the result.
- Declarative language.

Important Aspects

- DML – data manipulation.
- DDL – data definition.
- Remote Database Access
- Transactions management.
- Safety.

Basic form of a query

SELECT [*DISTINCT*] **list of selection**

FROM **list of origin**

WHERE qualifi**cation**

Table: ***Students***

CPF	Name	Grade
x	Zé das couves	10
y	Maria das desgraças	2
h	Silvio Santos	5

Components

list of selection – columns that must appear in the result.

list of origin – which tables will be consulted.

Qualification – conditions to be imposed in the consultation.

Example 1

- How to obtain a table with CPF and grades?

SELECT **CPF, Grade**

FROM **Students**

Observations

- The field's name has to be accurate.
- SQL is case insensitive.
- Separate the name from the columns by commas.

List of origin and Alias

- Alias is the "nickname". It is very used in SQL.

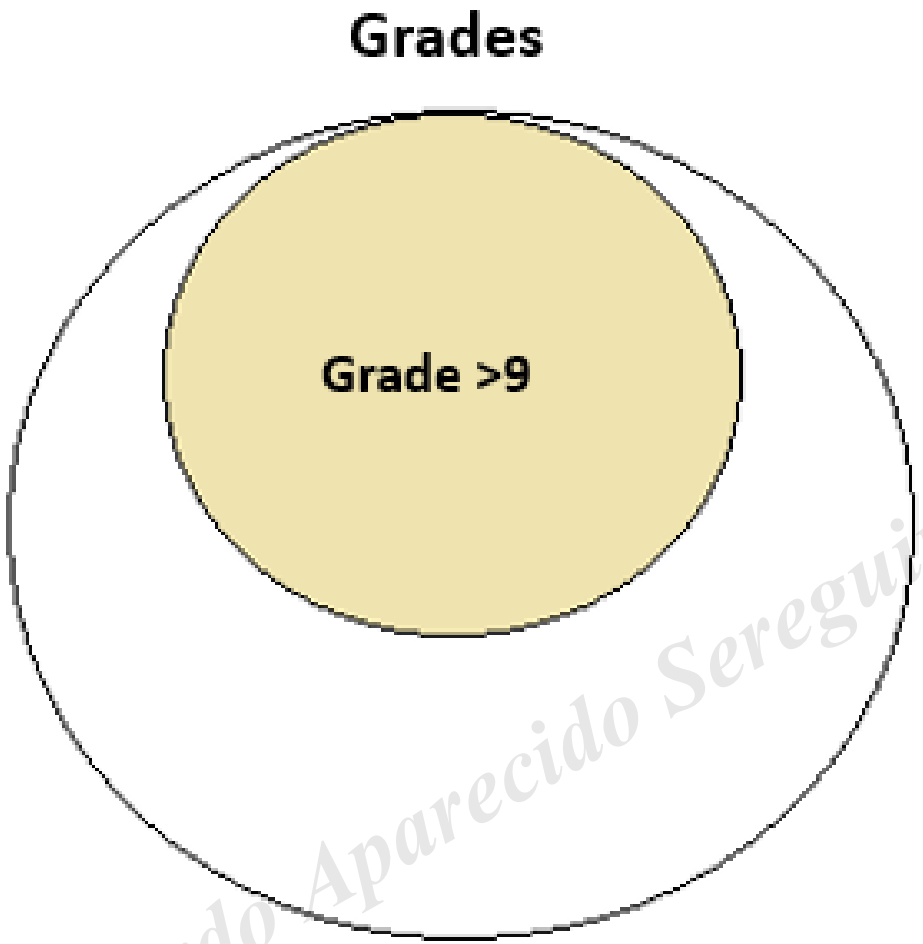
- You can use it to facilitate the understanding of your query.

SELECT A.**CPF,** A.**Grade**

FROM **Students** A

Qualification

- The qualification are the "where" clauses.
- These are a Boolean combination of conditions in the form of expressions.
- In terms of algebra, they are definitions of subsets.



Comparison Operators

Operator	Meaning
=	Equal to
> (Maior que)	Greater than
< (Menor que)	Less than
>= (Maior ou igual a)	Greater than or equal to
<= (Menor ou igual a)	Less than or equal to
<> (Diferente de)	It's different to

Example 2

- How to obtain a table with CPF and grades greater than 9?

```
SELECT CPF, Grade
FROM Students
WHERE Grade > 9
```

More than a clause

- In this case, we need to define how the relationship is between the clauses.
- Suppose that we have 2 conditions: condition-1 and condition-2.
- AND => the two conditions has to be true at the same time.
- OR => one of the two has to be true.

Operator	Meaning	Example
AND	and	Condition-1 AND condition-2
OR	or	Condition-1 OR condition-2

Example 3

CPF	NAME	GRADE	AGE
XX	JOÃO	10	20
YY	PEDRO	7	30

- How to obtain all records with grades greater than 6 **AND** age greater than 25?

```
SELECT *
FROM Students
WHERE Grade > 6 AND Age > 25
```

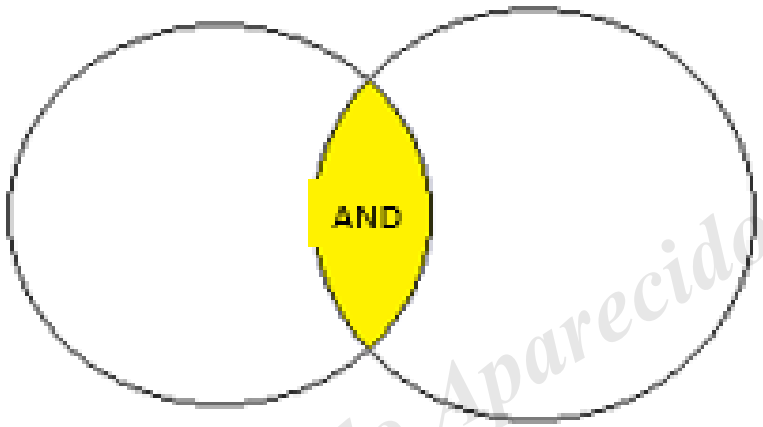
Example 4

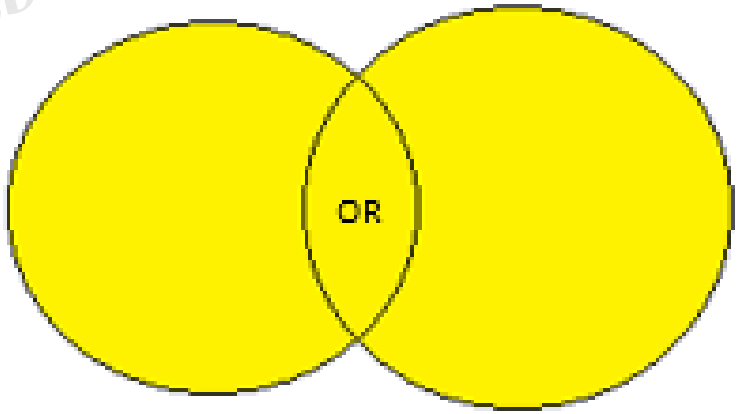
CPF	NAME	GRADE	AGE
XX	JOÃO	10	20
YY	PEDRO	7	30

- How to obtain all records with grades greater than 9 **OR** age greater than 25?

```
SELECT *
FROM Students
WHERE Grade > 9 OR Age > 25
```

Venn Diagram





Case of repeated lines

PIS	NAME
xxx	Pedro
xxx	Pedro

- Problem in the selection when there are lines with some column of repeated values.
- Use of DISTINCT.

SELECT *DISTINCT* **NAME**

FROM **Students**

Aggregation

How to add values per operations.

Data summarization operations:

- Average.
- Minimum.
- Maximum.
- Etc.

COUNT

- It counts the number of records under certain conditions.
- The following logic is presented:

SELECT COUNT**([Counted Field]), Grouped Fields**

FROM **Table**

GROUP BY **Grouped Fields**

Example 5

CPF	NAME	STATE
XXX	JOÃO	SP
YYY	PEDRO	SP
HHH	MARIANA	AL
JJJ	FLAVIA	RJ

```
SELECT COUNT(CPF)
FROM Students
```

4

Example 6

CPF	NAME	STATE
XXX	JOÃO	SP
YYY	PEDRO	SP
HHH	MARIANA	AL
JJJ	FLAVIA	RJ

```
SELECT COUNT(CPF), STATE
FROM Students
GROUP BY STATE
```

2	SP
1	AL
1	RJ

Example 7

CPF	NAME	STATE
XXX	JOÃO	SP
YYY	PEDRO	SP
HHH	MARIANA	AL
JJJ	FLAVIA	RJ

SELECT COUNT(CPF) THE count, STATE

FROM Students

GROUP BY STATE

ORDER BY count

1	AL
1	RJ
2	SP

SUM

- It sums the number of records under certain conditions.
- The following logic is presented:

SELECT SUM**([Summed Field]), Grouped Fields**

FROM **Table**

GROUP BY **Grouped Fields**

Example 8

CPF	NAME	STATE	AGE
XXX	JOÃO	SP	20
YYY	PEDRO	SP	30
HHH	MARIANA	AL	30
JJJ	FLAVIA	RJ	40

```
SELECT SUM(AGE)
FROM Students
```

120

Example 9

CPF	NAME	STATE	AGE
XXX	JOÃO	SP	20
YYY	PEDRO	SP	30
HHH	MARIANA	AL	30
JJJ	FLAVIA	RJ	40

```
SELECT SUM(AGE), STATE
FROM Students
GROUP BY STATE
```

50	SP
30	AL
40	RJ

AVERAGE (AVG)

- Calculate the arithmetic average of records under certain conditions.
- The following logic is presented:

SELECT AVG**([Field]), Grouped Fields**

FROM **Table**

GROUP BY **Grouped Fields**

Example 9

CPF	NAME	STATE	AGE
XXX	JOÃO	SP	20
YYY	PEDRO	SP	30
HHH	MARIANA	AL	30
JJJ	FLAVIA	RJ	40

```
SELECT AVG(AGE), STATE
FROM Students
GROUP BY STATE
```

25	SP
30	AL
40	RJ

MIN and MAX

- Get the lowest or the highest value of the records under certain conditions.
- The following logic is presented:

SELECT MIN**([Field]), Grouped Fields**
FROM **Table**
GROUP BY **Grouped Fields**

SELECT MAX**([Field]), Grouped Fields**
FROM **Table**
GROUP BY **Grouped Fields**

Example 10

CPF	NAME	STATE	AGE
XXX	JOÃO	SP	20
YYY	PEDRO	SP	30
HHH	MARIANA	AL	30
JJJ	FLAVIA	RJ	40

```
SELECT MAX(AGE), STATE
FROM Students
GROUP BY STATE
```

30	SP
30	AL
40	RJ

Example 11

CPF	NAME	STATE	AGE
XXX	JOÃO	SP	20
YYY	PEDRO	SP	30
HHH	MARIANA	AL	30
JJJ	FLAVIA	RJ	40

```
SELECT MIN(AGE), STATE
FROM Students
GROUP BY STATE
```

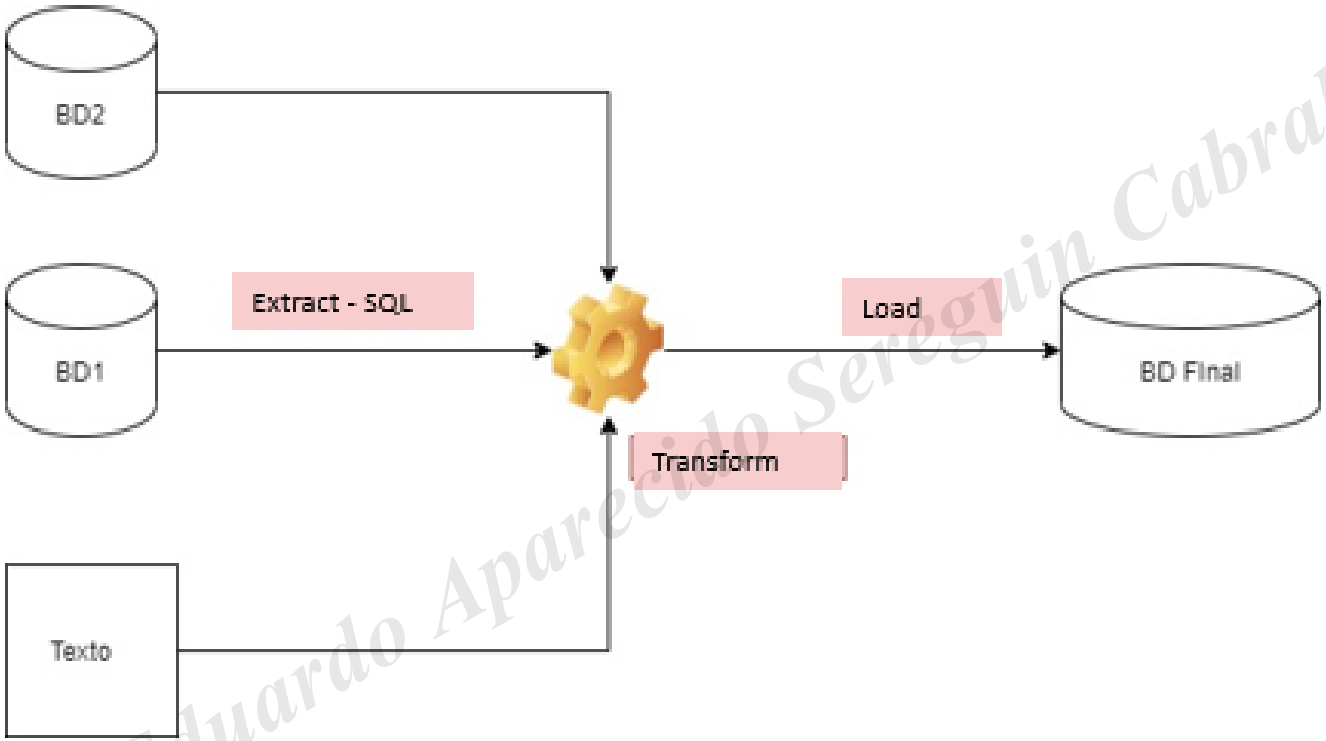
20	SP
30	AL
40	RJ

Introduction to ETL

Extract, Transformation and Load.

The integration of ETL data is a three-step process in which the data is extracted from one or more data sources, converted to the necessary state and loaded into a database or data warehouse in cloud.

Structure of an ETL



Frameworks for ETL

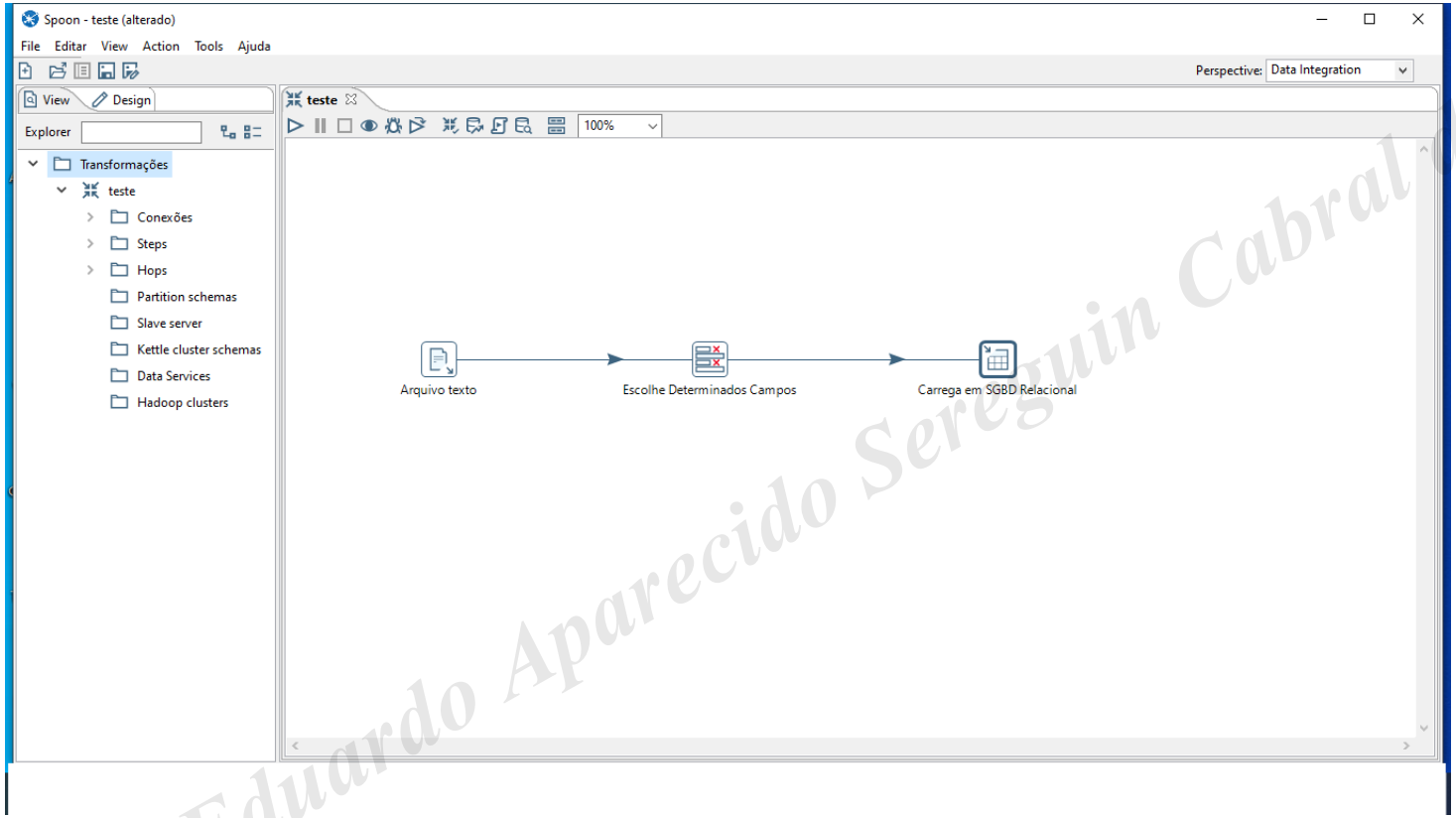
Apache Airflow

python™

pentaho®

Informatica

Example Pentaho



Create Table

- Create a Table with certain fields.
- The following logic is presented:

```
CREATE TABLE ADDRESS
(
Id_Student INTEGER,
Address Varchar(50)
)
```

Insert

- Insert certain fields.
- The following logic is presented:

```
INSERT INTO ADDRESS (Id_Student, Address)
VALUES (2, "RUA PEIXOTO DA SILVA")
```