DBMS

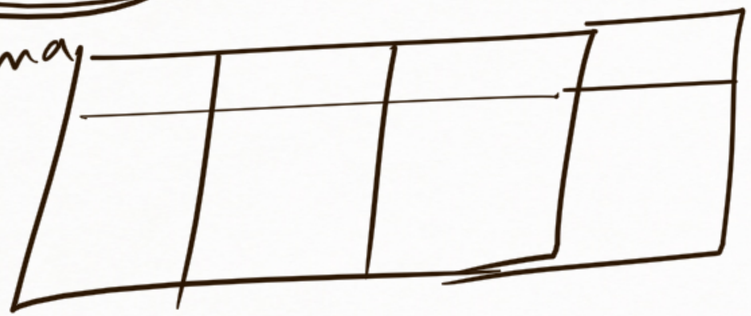
=> Different Types of SQL

1. DQL → Select, From

2. DML →  INSERT, UPDATE, DELETE

3. DDL → DROP, ALTER, CREATE

Schema of Table



Structure of Table

TCL

DCL

ER Diagrams

Requirements from Customers (Verbal info) → ER Diagrams (Visual Representation) → Actual DB (in the System in terms of tables)

Why ER Diagrams?

- High level View that we understand from Cust Req
- Readable by Non Tech Person

A discussion Prototype to make sure we are building the correct thing.

Entity Relationship Diagrams

→ Entity
→ Relationship
→ Attributes

} Main parts of ER Diagrams.

Entity Type / Entity Set

Nouns, Objects

=) Any living/Non living that you want to model.

=) House | OFFICE | Vendar | Person

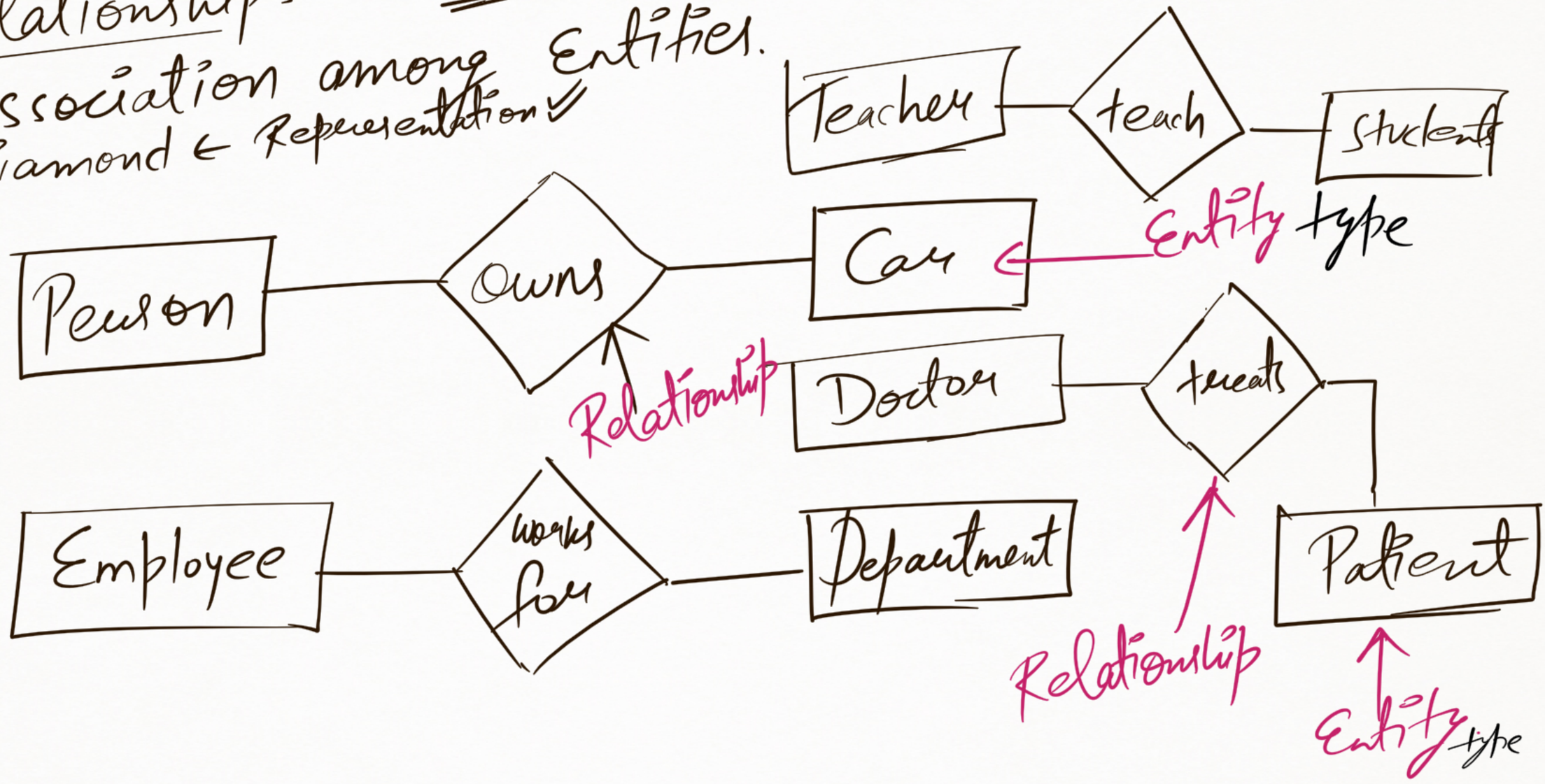
=) Teacher | Client | Store | Patient

Student | Product | Bamboo NB | Doctor

Relationships → Verbs

=) Association among Entities.

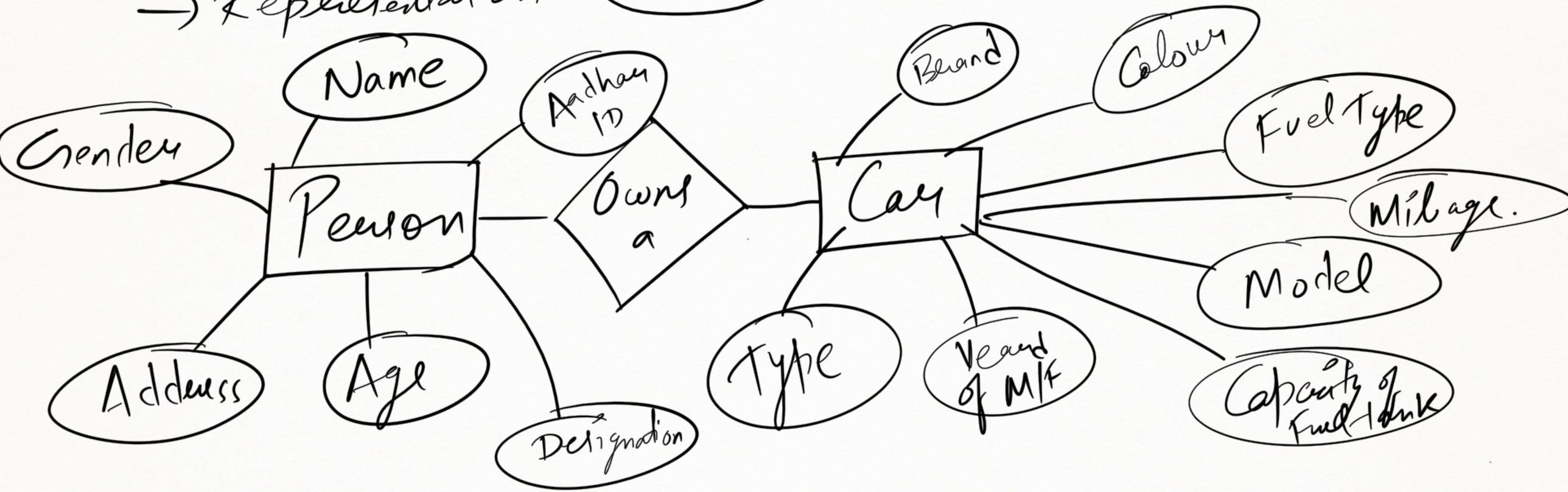
=) Diamond ← Representation



Attributes => Noun which describe Entities

→ Characteristics or property of Entity

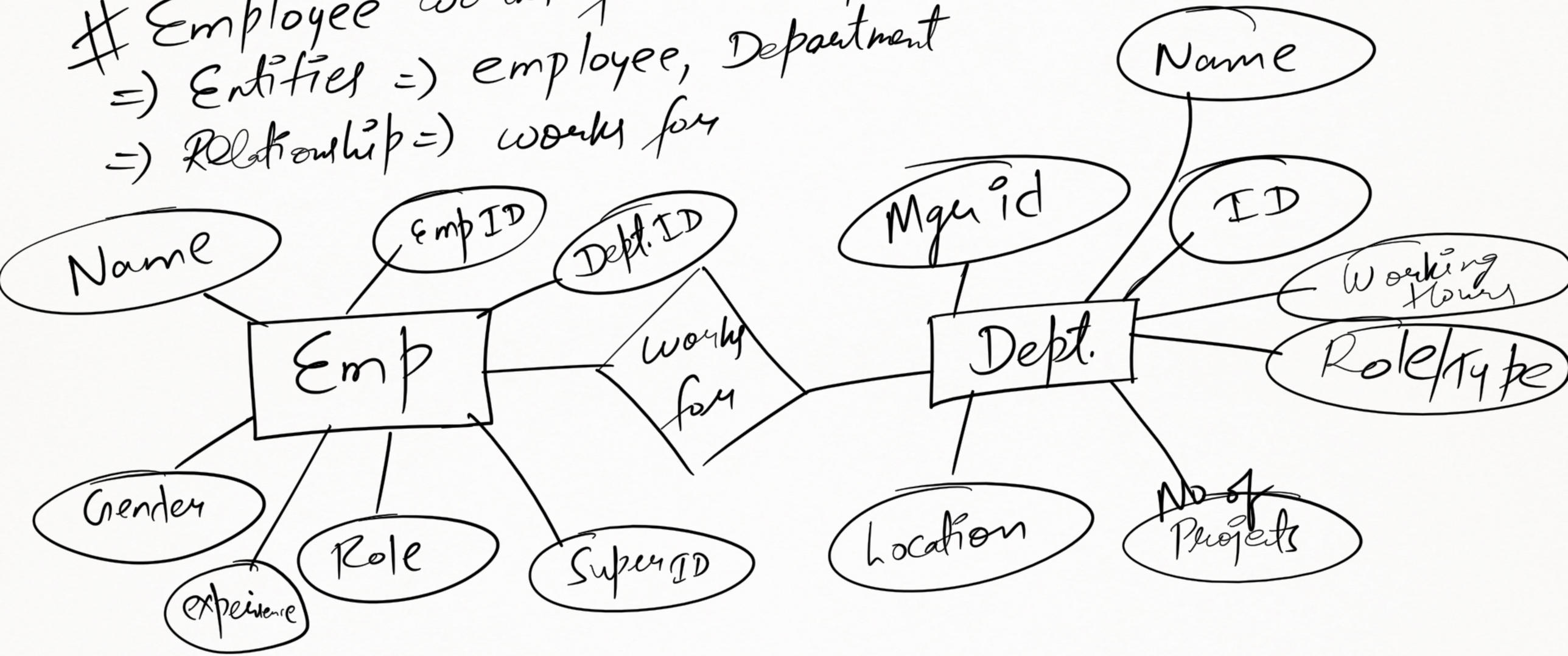
→ Representation = Ellipse/Oval

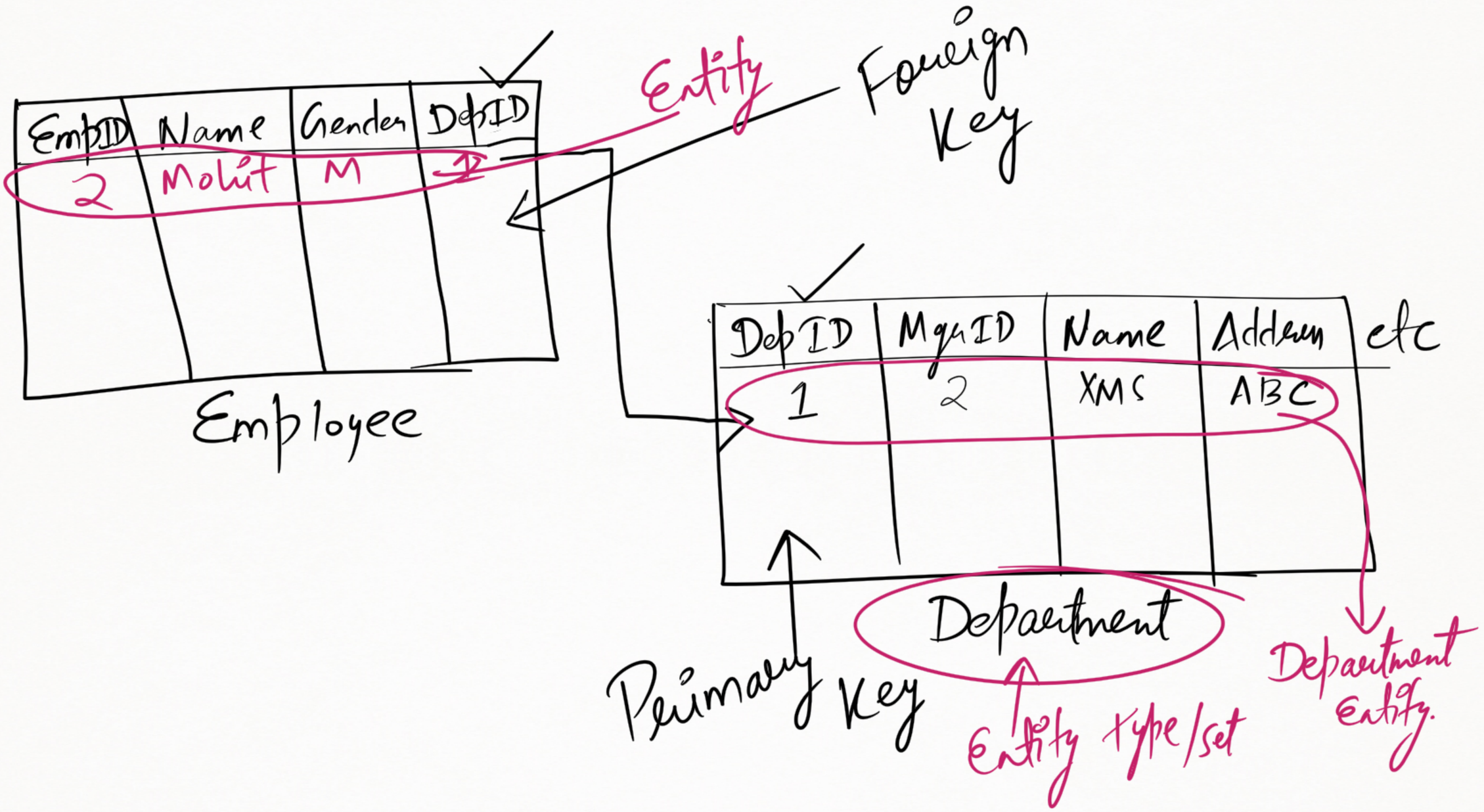


Employee works for a Department

=) Entities =) employee, Department

=) Relationship =) works for




EmpID
Name
Gender
DepID
2
Molut
M
1
Employee
Entity
Foreign Key
DepID
MgrID
Name
Addrss
etc
1
2
XMS
ABC
Department
Primary key
Entity type/set
Department Entity.

Schema : Structure of the table

Person(Age, Name, Address) ---} Schema

Blueprint of table

is a table

DESC Person

Columns	NULL	KEY	DEFAULT
col-1 col-2			

Schema

Every table will have its own BP/Schema.

Client(client_id, client_name, branch_id)

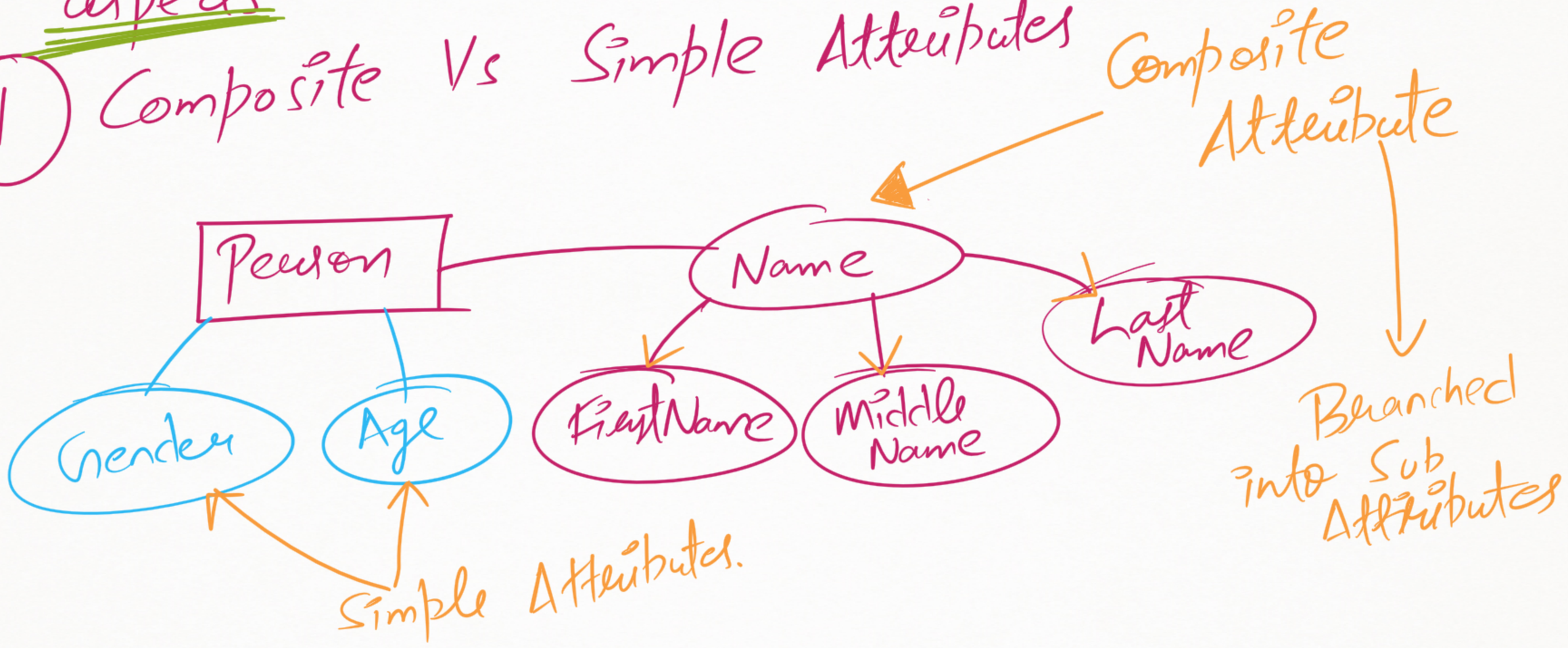
```
mysql> DESC client;
+-------------+-------------+------+-----+---------+-------+
| Field.      | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| client_id ✓ | int         | NO   | PRI | NULL    |       |
| client_name✓| varchar(40) | YES  |     | NULL    |       |
| branch_id ✓ | int         | YES  | MUL | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
3 rows in set (0.01 sec)
```

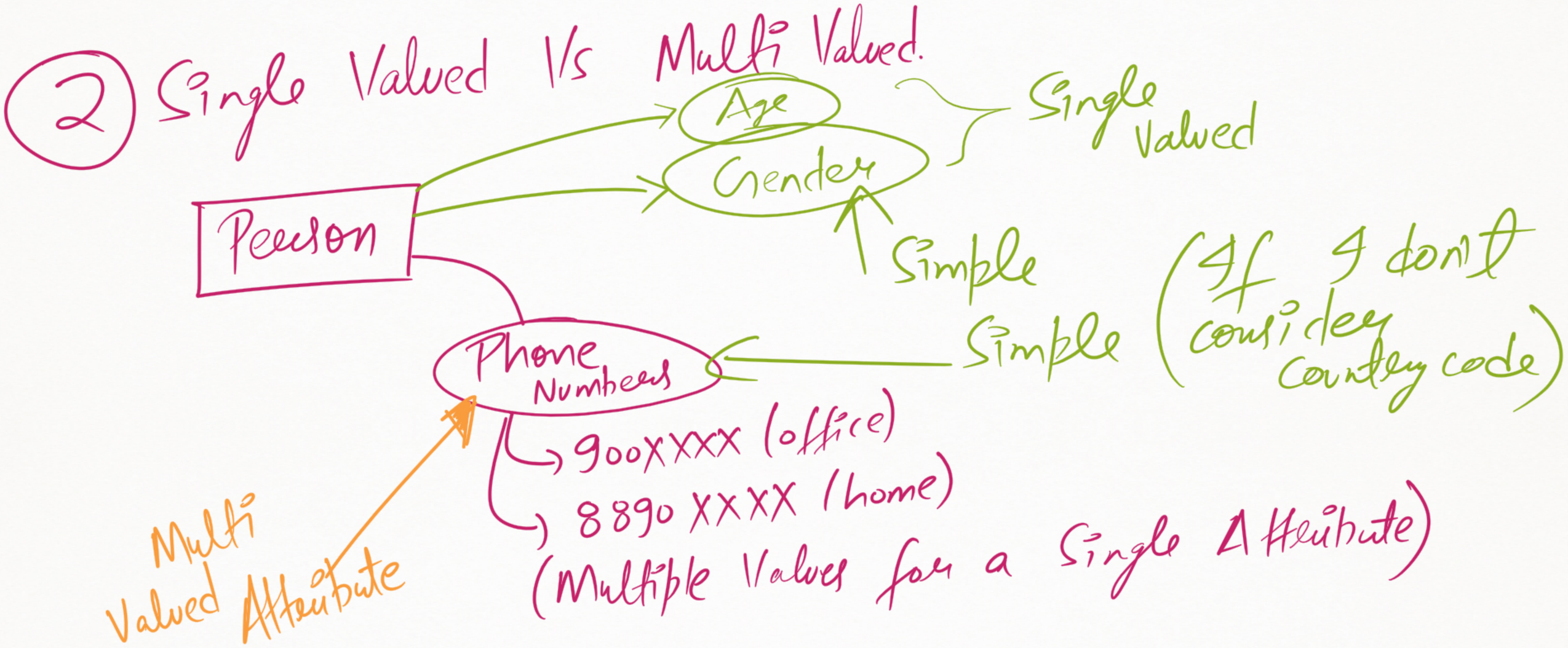
Theoretical Schema

Practical Schema

Types of Attributes based on different aspects

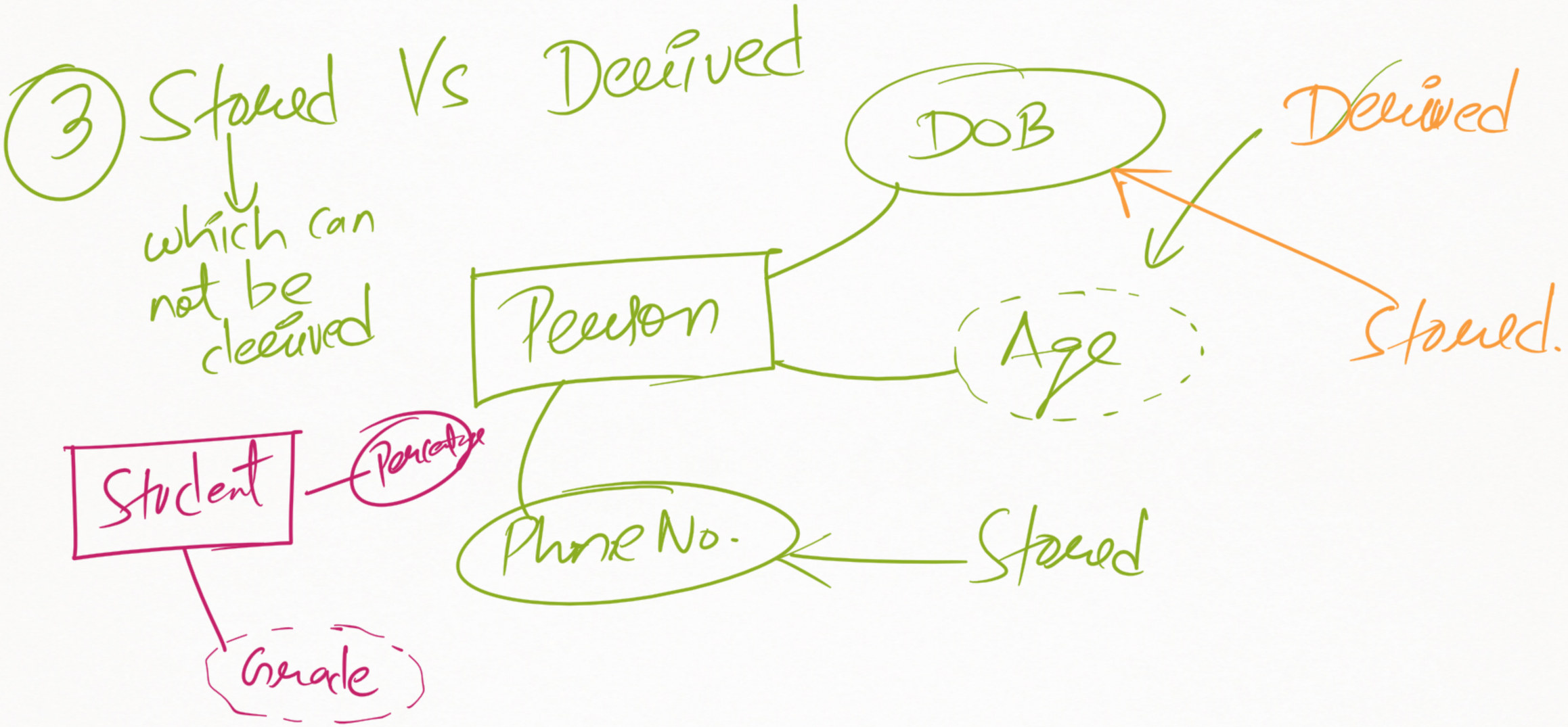
1 Composite Vs Simple Attributes

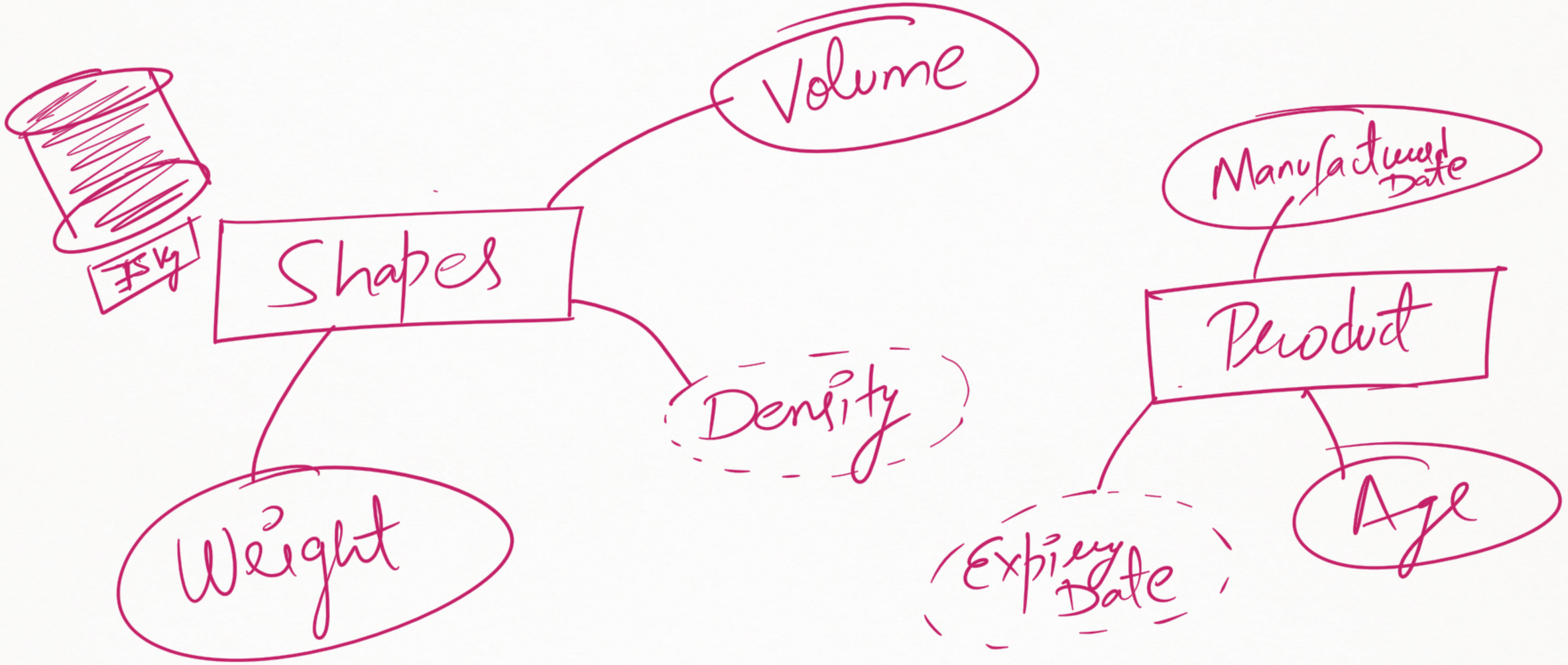



2 Single Valued V/s Multi Valued.
Age
Gender
Single Valued
Person
Simple
Simple
Phone Numbers
(If I don't consider country code)
900XXXX (office)
8890 XXXX (home)
(Multiple Values for a Single Attribute)
Multi Valued Attribute

Single Vs Simple

→ they both can come together, it is not Single Vs Simple, it is more like Single and Simple


3 Stored Vs Derived
which can not be derived
DOB
Derived
Person
Age
Stored.
Student
Percentage
Phone No.
Stored
Grade


Volume
Shapes
Density
Weight
Manufactured Date
Product
Age
Expiry Date

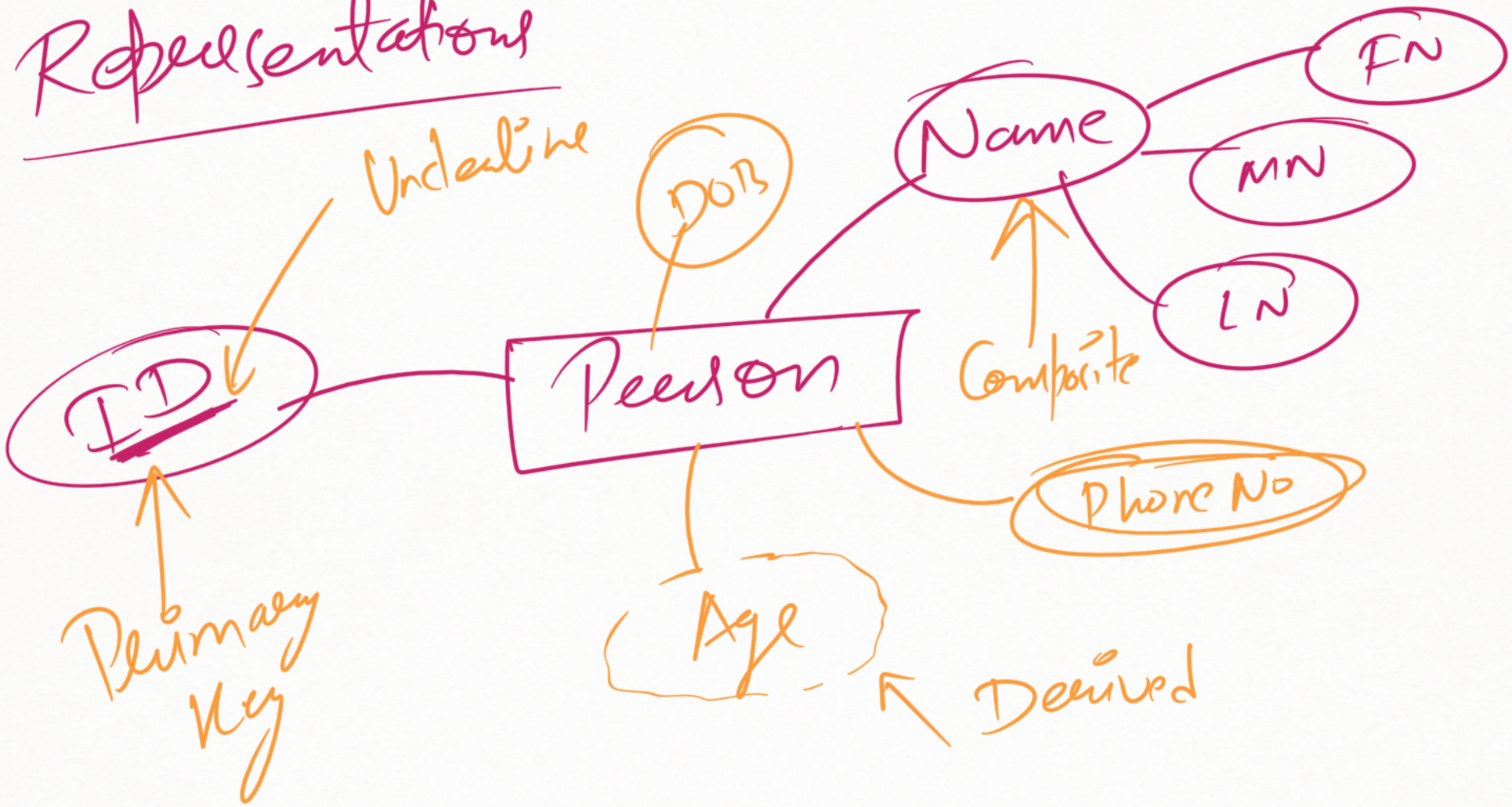
#④ Complex Attributes → Composite + Multivalued

→ Simple + Multivalued

Non Complex

↳ Simple + Single

Representations



#SQL

1> Group By

Name	Salary	Branch
a	50	X
b	55	Y
c	40	X
d	165	Y

Happy

Group By Branch

```
Select Branch
From Happy
GROUP BY Branch
```

X
Y

```
mysql> SELECT branch_id FROM employee GROUP BY branch_id;
+-----------+
| branch_id |
+-----------+
|         1 |
|         2 |
|         3 |
+-----------+
3 rows in set (0.01 sec)
```

Aggregate functions

- SUM ( ) ✓ →
- COUNT( ) →
- AVG( ) →
- Min( ) →
- Max( ) →

GROUP BY WHEN COMBINED WITH AGGREGATE FUNCTIONS

||

V.V. Interesting.

Order By

How do we want to sort the results?

```
mysql> SELECT emp_id,salary FROM employee ORDER BY salary;
+--------+--------+
| emp_id | salary |
+--------+--------+
|    104 |  55000 |
|    103 |  63000 |
|    107 |  65000 |
|    105 |  69000 |
|    108 |  71000 |
|    102 |  75000 |
|    106 |  78000 |
|    101 | 110000 |
|    100 | 250000 |
+--------+--------+
9 rows in set (0.00 sec)
```

← Min

← Max.

By default Ascending order

```
mysql> SELECT emp_id,salary FROM employee ORDER BY salary ASC;
+--------+--------+
| emp_id | salary |
+--------+--------+
|    104 |  55000 |
|    103 |  63000 |
|    107 |  65000 |
|    105 |  69000 |
|    108 |  71000 |
|    102 |  75000 |
|    106 |  78000 |
|    101 | 110000 |
|    100 | 250000 |
+--------+--------+
9 rows in set (0.00 sec)
```

Ascending Order

```
mysql> SELECT emp_id,salary FROM employee ORDER BY salary DESC;
+--------+--------+
| emp_id | salary |
+--------+--------+
|    100 | 250000 |
|    101 | 110000 |
|    106 |  78000 |
|    102 |  75000 |
|    108 |  71000 |
|    105 |  69000 |
|    107 |  65000 |
|    103 |  63000 |
|    104 |  55000 |
+--------+--------+
9 rows in set (0.00 sec)
```

DESCENDING Order.