EXAM PROJECT FOR BIG DATA 2 : MANAGING DATABASES, SQL AND NOSQL

MAGISTERE 1

Members of the group :

Nicolas Guillotin

Yann Sasse

Ulrich Segodo

Berich Zinsou-Daho

Teacher :

Adrian Chifu

AIX-MARSEILLE SCHOOL OF ECONOMICS,
APRIL 2022

Table of contents

Introduction

As part of the evaluation of the Big Data 2 module for the first year of Magistere, we are asked to carry out a project on database management in SQL and NoSQL. In a preliminary step, we have created a context related to the functioning of a restaurant, a situation that we have modeled by creating 22 tables and 59 attributes referring to it. This was the basis for the main work on the handling of the tables.

The work itself is divided into three (03) parts. In part one, we create the database model including the data dictionary, the conceptual and logical data models, then the tables in which we enter the data using mysql. The second part is about creating and running queries ; for these queries, we also provide SQL solutions. Finally, in part three, we first choose in our model, three (03) representative tables, for which we provide the .json file containing our data and the script used to insert it. After that, we choose one (01) of the tables and we provide for it a python script containing the connection to the database and five (05) queries with their corresponding results.

Part 1 : Model

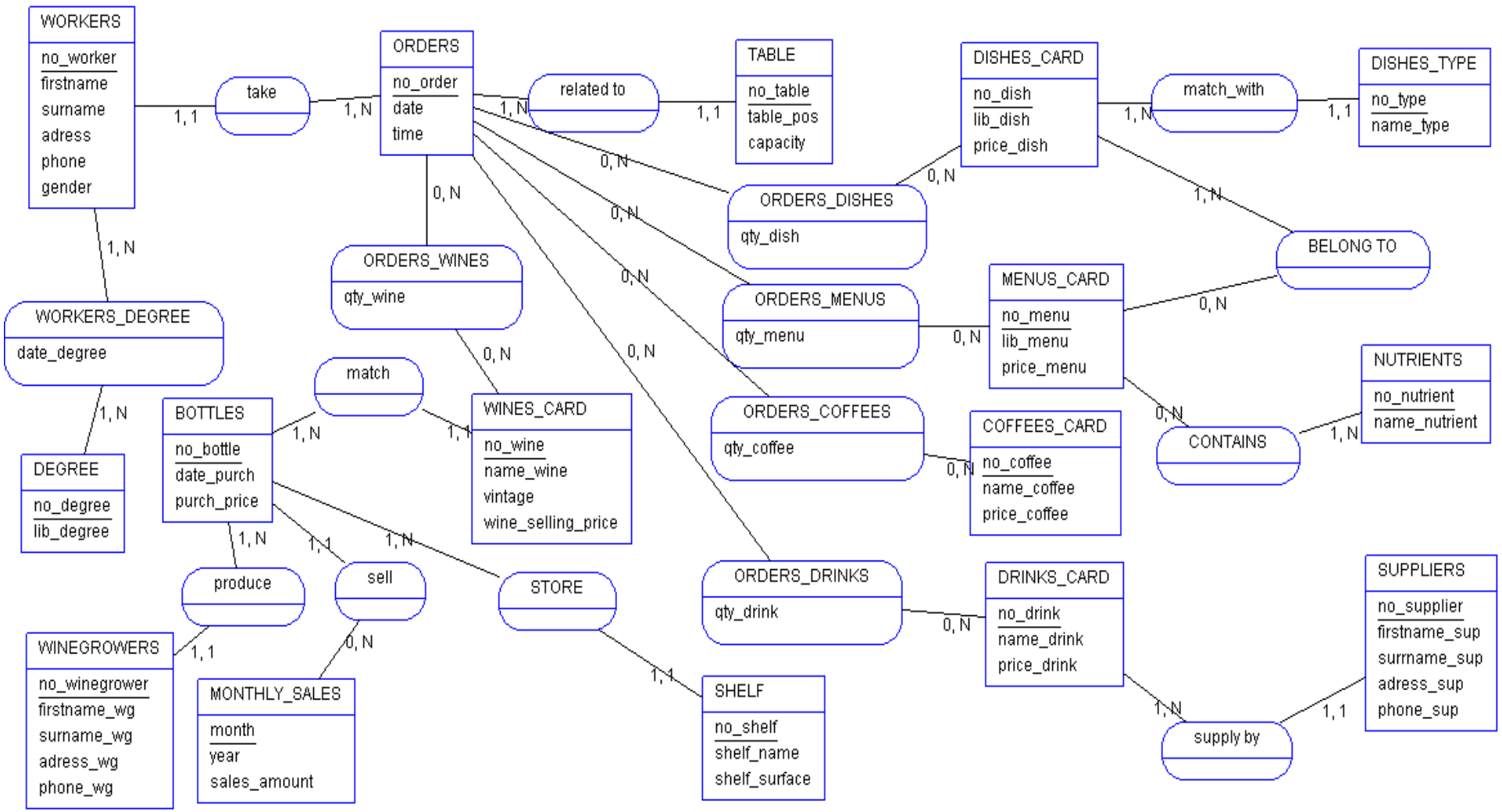
1-1. Data Dictionary

Attributs	**Description**	**Type**
no_worker	Worker ID	Numeric
Firstname	First name of worker	Text
Surname	Surname of worker	Text
Adress	Adress of worker	Text
Phone	Worker's phone number	Numeric
Gender	Gender of the worker	Text
no_degree	Degree ID	Numeric
Lib_degree	Title of the degree	Text
Date_degree	Date of graduation	Date
No_table	Table ID	Numeric
Table_pos	Position of the table in the restaurant	Text
Capacity	Capacity of the table	Numeric
no_order	Order ID	Numeric
Date	Date the order was placed	Date
Time	Time the order was placed	Time
No_dish	Dish ID	Numeric
Lib_dish	Name of the dish	Text
Price_dish	Price of the dish	Numeric
Qty_dish	Number of dishes in the order	Numeric
No_type	Type ID	Numeric
Name_type	Name of the type of dish	Text
No_menu	Menu ID	Numeric
Lib_menu	Name of the menu	Text
Price_menu	Price of the menu	Numeric
Qty_menu	Number of menus in the order	Numeric
No_nutrients	Nutrient ID	Numeric
Name_nutrients	Name of the nutrient	Text
No_wine	Wine ID	Numeric

Name_wine	Title of the wine	Text
Vintage	Vintage of the wine	Numeric
Wine_selling_price	selling price of the wine	Numeric
Qty_wine	Number of wines in the order	Numeric
No_bottle	Bottle ID	Numeric
Date_purch	Date of purchase of the bottle of wine	Date
Purch_price	Purchase price of wine	Numeric
No_shelf	Shelf ID	Numeric
Shelf_name	Name of the shelf	Text
Shelf_surface	Area occupied by the shelf	Numeric
No_winegrower	Winegrower ID	Numeric
Firstname_wg	First name of the winegrower	Text
Surname_wg	Surname of the winegrower	Text
Adress_wg	Adress of the winegrower	Text
Phone_wg	Winegrower's phone number	Numeric
Month	Month in which a bottle of wine was sold	Entier
Year	Year in which a bottle of wine was sold	Entier
Sales_amount	monthly amount of wine sales	float
No_coffee	Coffee ID	Numeric
Name_coffee	Name of the coffee	Text
Price_coffee	Price of the coffee	Numeric
Qty_coffee	Number of coffee in the order	Numeric
No_drink	Drink ID	Numeric
Name_drink	Name of the drink	Text
Price_drink	Price of drink	Numeric
Qty_drink	Number of drink in the order	Numeric
No_supplier	Supplier ID	Numeric
Firstname_sup	First name of the supplier	Text
Suname_sup	Surname of the supplier	Text
Phone_sup	Supplier's phone number	Numeric
Adress_sup	Adress of the supplier	Text

1-2. Conceptual Data Model (CDM)

1-2-1. Preliminary CDM


WORKERS
no_worker
firstname
surname
adress
phone
gender
take
ORDERS
no_order
date
time
related to
TABLE
no_table
table_pos
capacity
DISHES_CARD
no_dish
lib_dish
price_dish
match_with
DISHES_TYPE
no_type
name_type
ORDERS_DISHES
qty_dish
BELONG TO
ORDERS_WINES
qty_wine
ORDERS_MENUS
qty_menu
MENUS_CARD
no_menu
lib_menu
price_menu
WORKERS_DEGREE
date_degree
match
WINES_CARD
no_wine
name_wine
vintage
wine_selling_price
ORDERS_COFFEES
qty_coffee
COFFEES_CARD
no_coffee
name_coffee
price_coffee
CONTAINS
NUTRIENTS
no_nutrient
name_nutrient
DEGREE
no_degree
lib_degree
BOTTLES
no_bottle
date_purch
purch_price
produce
sell
STORE
ORDERS_DRINKS
qty_drink
DRINKS_CARD
no_drink
name_drink
price_drink
SUPPLIERS
no_supplier
firstname_sup
surrname_sup
adress_sup
phone_sup
supply by
WINEGROWERS
no_winegrower
firstname_wg
surname_wg
adress_wg
phone_wg
MONTHLY_SALES
month
year
sales_amount
SHELF
no_shelf
shelf_name
shelf_surface

1-2-2. Normalisation

The first normal form (1NF) is respected for all attributes except adress, adress_sup and adress_wg which do not have atomic values. Since the adress will be composed of the street, city, postal code, and country. We will therefore create a table with these attributes. As far as 2FN and 3FN are concerned, they are respected.

Our new data dictionary is then :

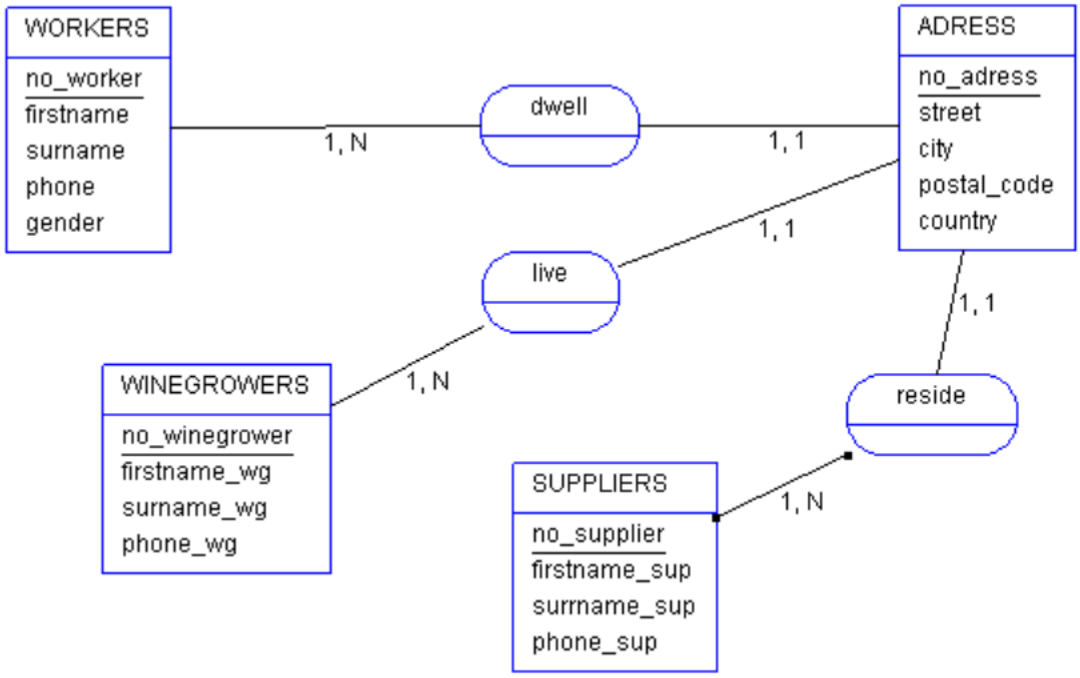
Attributs	**Description**	**Type**
no_worker	Worker ID	Numeric
Firstname	First name of worker	Text
Surname	Surname of worker	Text
Phone	Worker's phone number	Numeric
Gender	Gender of the worker	Text
no_degree	Degree ID	Numeric
Lib_degree	Title of the degree	Text
Date_degree	Date of graduation	Date
No_table	Table ID	Numeric
Table_pos	Position of the table in the restaurant	Text
Capacity	Capacity of the table	Numeric
no_order	Order ID	Numeric
Date	Date the order was placed	Date
Time	Time the order was placed	Time
No_dish	Dish ID	Numeric
Lib_dish	Name of the dish	Text
Price_dish	Price of the dish	Numeric
Qty_dish	Number of dishes in the order	Numeric
No_type	Type ID	Numeric
Name_type	Name of the type of dish	Text
No_menu	Menu ID	Numeric
Lib_menu	Name of the menu	Text
Price_menu	Price of the menu	Numeric

Qty_menu	Number of menus in the order	Numeric
No_nutrients	Nutrient ID	Numeric
Name_nutrients	Name of the nutrient	Text
No_wine	Wine ID	Numeric
Name_wine	Title of the wine	Text
Vintage	Vintage of the wine	Numeric
Wine_selling_price	selling price of the wine	Numeric
Qty_wine	Number of wines in the order	Numeric
No_bottle	Bottle ID	Numeric
Date_purch	Date of purchase of the bottle of wine	Date
Purch_price	Purchase price of wine	Numeric
No_shelf	Shelf ID	Numeric
Shelf_name	Name of the shelf	Text
Shelf_surface	Area occupied by the shelf	Numeric
No_winegrower	Winegrower ID	Numeric
Firstname_wg	First name of the winegrower	Text
Surname_wg	Surname of the winegrower	Text
Phone_wg	Winegrower's phone number	Numeric
month	Month in which a bottle of wine was sold	Entier
Year	Year in which a bottle of wine was sold	Entier
Sales_amount	monthly amount of wine sales	Float
No_coffee	Coffee ID	Numeric
Name_coffee	Name of the coffee	Text
Price_coffee	Price of the coffee	Numeric
Qty_coffee	Number of coffee in the order	Numeric
No_drink	Drink ID	Numeric
Name_drink	Name of the drink	Text
Price_drink	Price of drink	Numeric
Qty_drink	Number of drink in the order	Numeric
No_supplier	Supplier ID	Numeric
Firstname_sup	First name of the supplier	Text
Suname_sup	Surname of the supplier	Text
Phone_sup	Supplier's phone number	Numeric

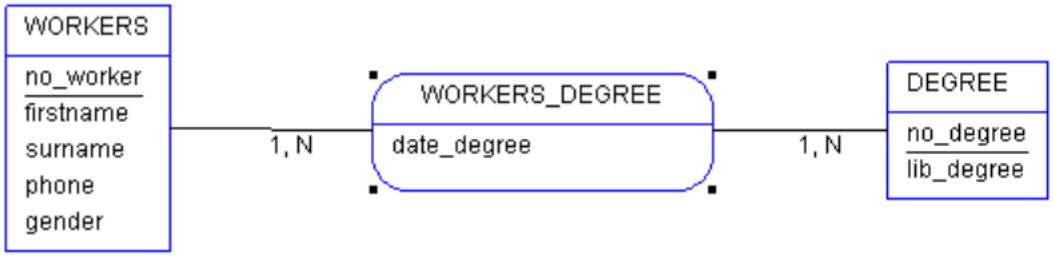
No_adress	Adress ID	Numeric
Street	Street	Text
City	City	Text
Postal_code	Postal code	Numeric
Country	Country	Text

1-2-3. Hypothesis

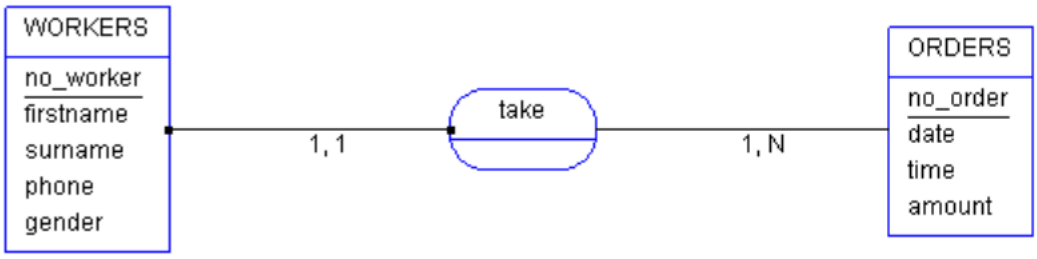
1. A worker lives at one and only one address. An address can correspond to one or more workers. The same applies to the supplier and the winegrower



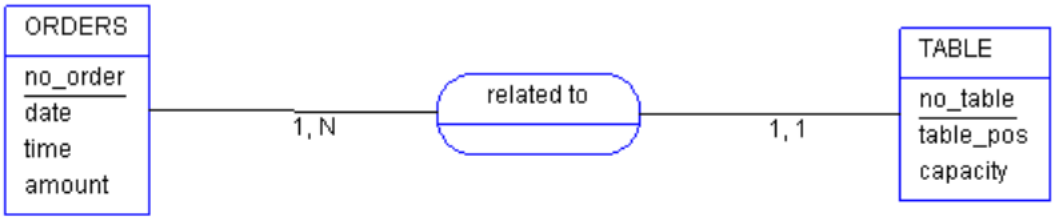
2. A worker may have one or more diplomas at given dates. A degree can be obtained by one or more workers



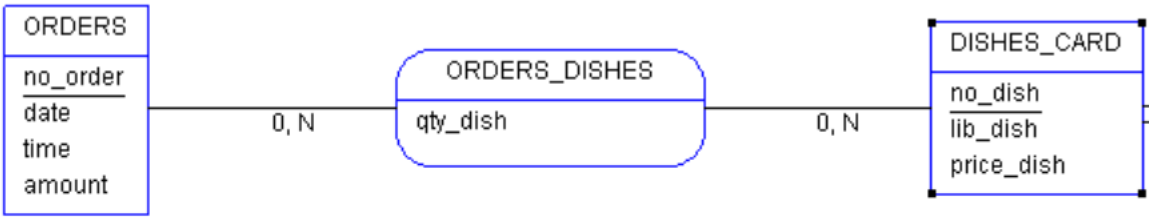
3. A worker can take one or more orders. An order is taken by one and only one worker



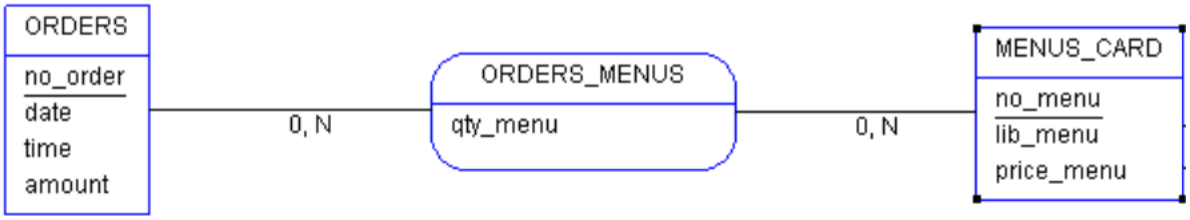
4. An order corresponds to one and only one table. A table can place one or more orders.



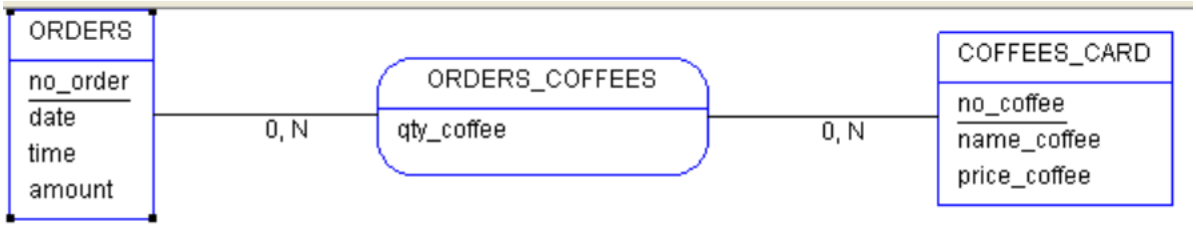
5. An order may contain zero or more dishes that are on the dish menu. A dish can be found in zero or more orders.



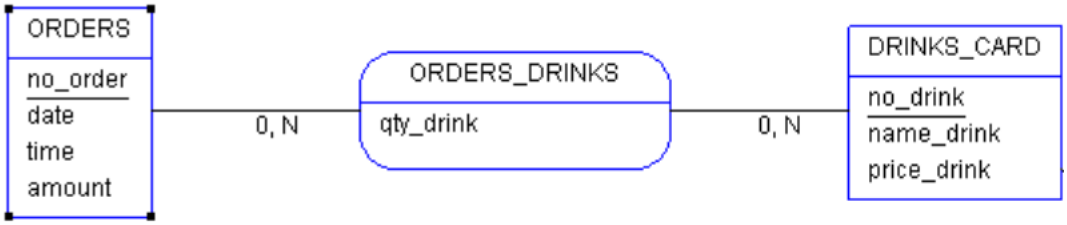
6. An order can contain zero or more menus that are on the menu card. A menu can be found in zero or more orders.



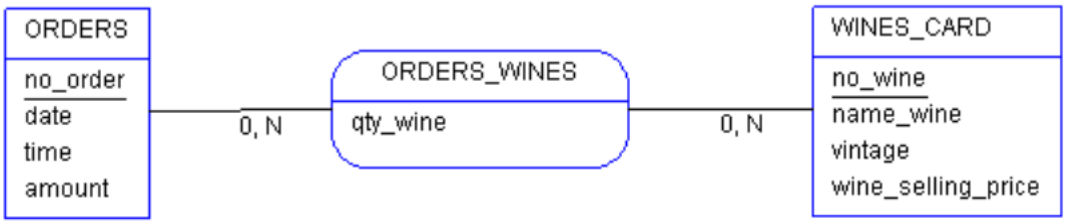
7. An order may contain zero or more coffees that are on the coffee menu. A coffee can be found in zero or more orders.



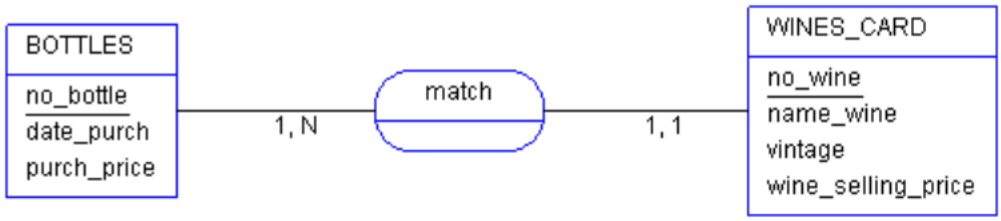
8. An order can contain zero or more drinks that are on the drinks menu. A drink can be found in zero or more orders.



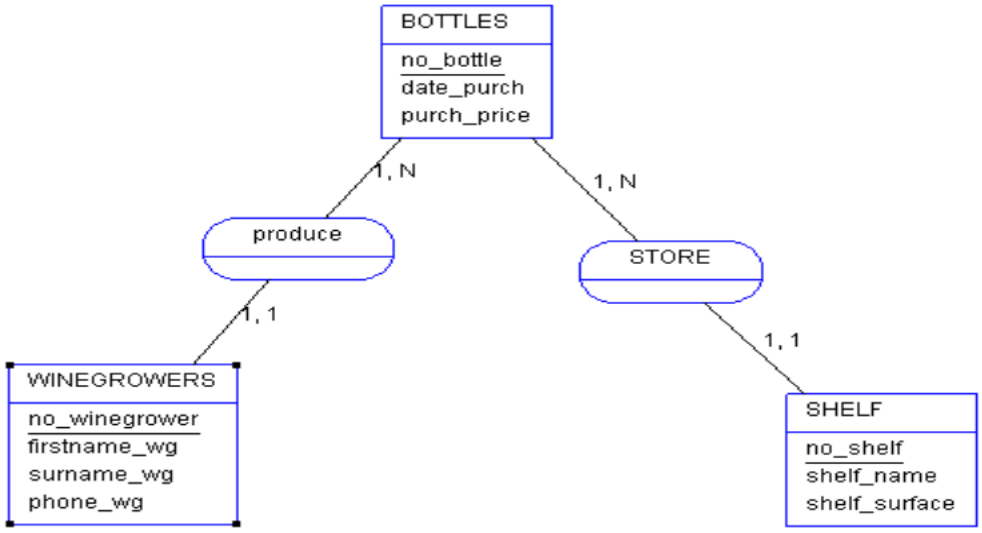
9. An order may contain zero or more wines that are on the wine list. A wine can be found in zero or more orders.



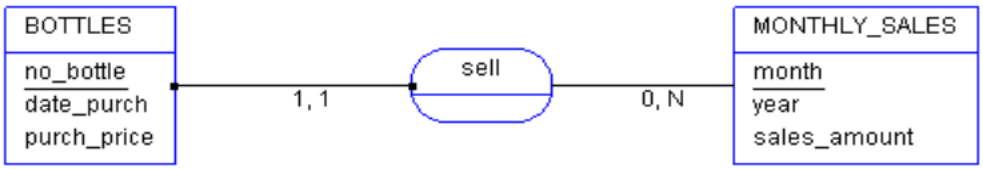
10. A wine on the wine list corresponds to one or more bottles in stock and a specific bottle corresponds to one and only one wine description.



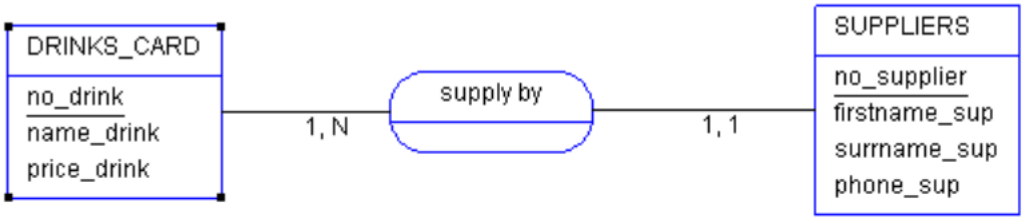
11. A bottle is supplied by one and only one winegrower and a winegrower can supply one or more bottles. A bottle is stored in one and only one shelf and in one shelf can be stored one or more bottles



12. One bottle can be sold zero or more per month. A sale corresponds to one and only one bottle

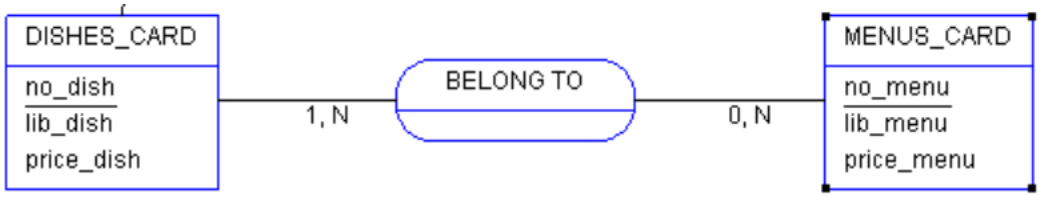


13. A drink is supplied by one and only one supplier. A supplier may provide one or more beverages



14. A menu contains at least one nutrient. A nutrient can be found in zero or more menus.

15. A menu can contain one or more dishes. A dish can be found in zero or more menus



16. There is only one type of dish for each course, either a starter, a main course or a dessert. Conversely, a starter may have one or more different dishes.

1-2-4. Model's limitations

A certain number of elements involved in the daily management of a restaurant are not taken into account by our model. We preferred to restrict the model to the elements that are there so as not to make the work more cumbersome.

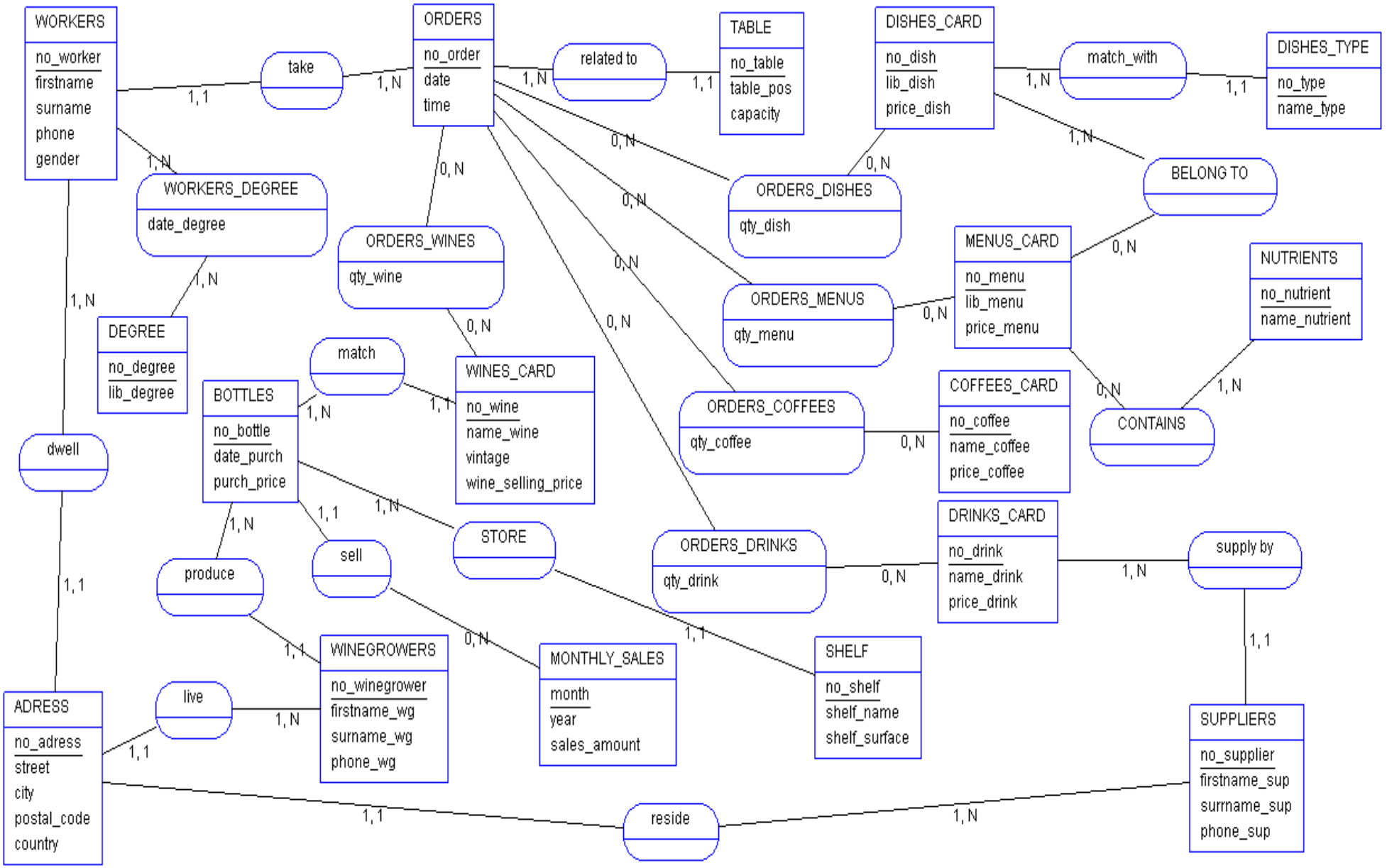
We do not take into account the possibilities of payment of invoices (cash or credit card).

In the personnel management we do not take into account the potential experience of the employee, and other personal information/data.

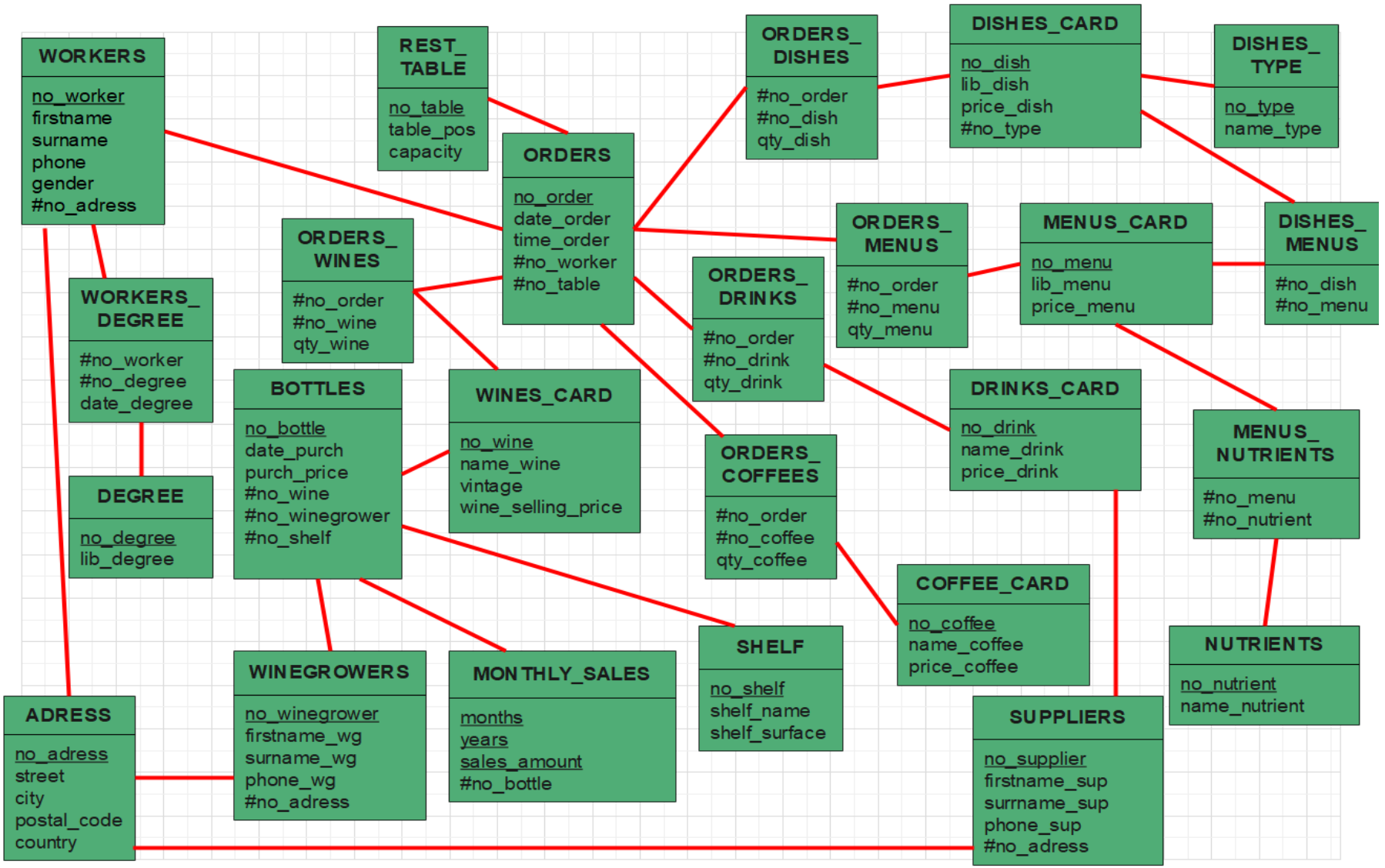
In the conceptual data model, some of our relationships are limited so as not to enlarge the scope of the model. For example, the relationship between suppliers and beverages was limited by not considering the fact that a beverage can be supplied by several suppliers, given that the manufacturing license could change from one supplier to another. The same applies to the relationship between wine producers and the bottles they produce.

We have a somewhat limited wine storage management: we did not want to extend it further because it was not necessarily the main purpose of the work.

1-2-5. Final CDM



1-3. Transformation of the CDM into the Logical Data Model (LMD)



1-4. Creation script for mysql (see .txt file)

1-5. Insertion of data into tables (see .txt file)

Part 2 : Queries and solutions

2.1. Five basic queries

```
 9   -- 1: Display all the names of the workers.
10   SELECT firstname, surname
11       FROM workers;
12
13   -- 2: Display all the properties of the shelf.
14   SELECT *
15       FROM shelf;
16
17   -- 3: Display all the names of the drinks.
18   SELECT name_drink
19       FROM drinks_card;
20
21   -- 4: Display the differents degrees of the workers.
22   SELECT lib_degree
23       FROM degree;
24
25   -- 5: Display the number of places of each table.
26   SELECT capacity, no_table
27       FROM rest_table;
```

2.2. Five WHERE clause queries

```sql
-- 1: Display the name of the drinks whose price is higher than 1€.
SELECT name_drink
    FROM drinks_card
    WHERE price_drink > 1;

-- 2: Display the bottle numbers whose sales amount are higher than 200€.
SELECT no_bottle
    FROM monthly_sales
    WHERE sales_amount > 200;

-- 3: Display the type of the dish #2.
SELECT name_type
    FROM dishes_types
    WHERE no_type = 2;

-- 4: Display the number and the name of menus whose price is between 15€ and 20€.
SELECT no_menu, lib_menu
    FROM menus_card
    WHERE price_menu
    BETWEEN 15 AND 20;

-- 5: Display bottles sold in March 2020.
SELECT no_bottle
    FROM monthly_sales
    WHERE years=2020 AND months=03;
```

2.3. Five ORDER BY queries

```sql
-- 1 : Display the list of dishes sorted by prices in descending order.
SELECT *
    FROM dishes_card
    ORDER BY price_dish DESC;

-- 2 : Display the list of menus sorted by menu number in ascending order.
SELECT *
    FROM menus_card
    ORDER BY no_menu;

-- 3 : Display the name of the coffees sorted by prices in ascending order.
SELECT name_coffee, price_coffee
    FROM coffee_card
    ORDER BY price_coffee;

-- 4 : Display the phone of the workers sorted by workers' firstname.
SELECT phone, surname
    FROM workers
    ORDER BY surname;

-- 5 : Display the order number sorted by order's date.
SELECT no_order, date_order
    FROM orders
    ORDER BY date_order;
```

2.4. Five Multi-Table queries

```sql
-- 1 : Display the city of the worker #5.
SELECT city
    FROM adress a, workers w
    WHERE a.no_adress=w.no_adress AND no_worker=5;

-- 2 : Display the date degree of all the male workers.
SELECT date_degree, gender
    FROM workers w, workers_degree wd
    WHERE gender="male" AND w.no_worker=wd.no_worker;

-- 3 : Display the nutrients name of the menu #8, the pizza menu.
SELECT name_nutrient
    FROM menus_nutrients mn, nutrients n
    WHERE no_menu=8 AND mn.no_nutrient=n.no_nutrient;

-- 4 : Display the type name of the dish #11.
SELECT name_type
    FROM dishes_card dm, dishes_types dt
    WHERE no_dish=11 AND dm.no_type=dt.no_type;

-- 5 : Display the shelf name of the bottle #6.
SELECT shelf_name
    FROM shelf s, bottles b
    WHERE no_bottle=6 AND s.no_shelf=b.no_shelf;
```

2.5. Five queries with Numeric expressions and functions

```sql
-- 1 Display the average price of Espresso coffees
SELECT AVG(price_coffee) AS average_price
    FROM COFFEE_CARD
    WHERE name_coffee='Espresso';

-- 2 Display the total capacity of the restaurant (total number of available places)
SELECT SUM(rest_table.capacity) AS total_capacity
    FROM rest_table;

-- 3 Display the number, the position and the capacity of the table with the highest capacity
SELECT no_table, table_pos, capacity
    FROM rest_table
    WHERE capacity = (SELECT MAX(capacity)
                      FROM rest_table);

-- 4 Display the names of drinks with a price lower than the average drinks price
SELECT name_drink
    FROM drinks_card
    WHERE price_drink < (SELECT AVG(price_drink)
                         FROM drinks_card);

-- 5 Display for wines produced after 2000, the current price and the price after a 20% discount
SELECT name_wine, wine_selling_price, wine_selling_price*0.8 AS reduced_rate
    FROM wines_card
    WHERE vintage > 2000;
```

2.6. Five GROUP BY queries

```sql
-- 1 Display the average table capacity by position
SELECT table_pos, AVG(capacity)
    FROM rest_table
    GROUP BY table_pos;

-- 2 Display the number of workers per gender
SELECT gender, COUNT(no_worker)
    FROM workers
    GROUP BY gender;

-- 3 Display the number of workers with 4 degrees at least
SELECT w.no_worker, COUNT(no_degree)
    FROM workers w, workers_degree wd
    WHERE w.no_worker=wd.no_worker
    GROUP BY(no_worker)
    HAVING COUNT(no_degree)>=4;

-- 4 Display for each position, the position and the total capacity
SELECT table_pos, SUM(capacity)
    FROM rest_table
    GROUP BY table_pos;

-- 5 Display for each month regardless of the year, the total amount of sales
SELECT months, SUM(sales_amount)
    FROM monthly_sales
    GROUP BY months;
```

2.7. Five nested queries

```sql
-- 1. Show labels for dishes cheaper than cassoulet
SELECT lib_dish
    FROM dishes_card
    WHERE price_dish < (SELECT price_dish
                        FROM dishes_card
                        WHERE lib_dish = 'cassoulet');

-- 2. Show most expensive wine label
SELECT name_wine
    FROM wines_card
    WHERE wine_selling_price = (SELECT MAX(wine_selling_price)
                                FROM wines_card);

-- 3. Display the list of workers who graduated in 2020
SELECT *
    FROM workers
    WHERE no_worker in (SELECT no_worker
                        FROM workers_degree
                        WHERE date_degree LIKE '2020%');

-- 4. Display the list of workers who live in the same street as suppliers
SELECT surname, firstname
    FROM workers w
    JOIN adress a
    ON(w.no_adress = a.no_adress)
    AND a.street in (SELECT DISTINCT(street)
                        FROM adress a
                        JOIN suppliers s
                        ON(a.no_adress = s.no_adress));

-- 5. Display the list of workers who do not live in the same street as winegrowers
SELECT surname, firstname
    FROM workers w
    JOIN adress a
    ON(w.no_adress = a.no_adress)
    AND a.street NOT IN (SELECT DISTINCT(street)
                            FROM adress a
                            JOIN winegrowers wg
                            ON(a.no adress = wg.no adress));
```

Part 3 : mongoDB

3-1. Creation of .json files

For the three (03) representative entities, we choose **workers, suppliers and winegrowers**. As these tables contain foreign keys, we proceeded to denormalization in order to have more information in each .json file.

- Importing .json files

To import the three (03) .json files, we created a directory named **projet** with the mkdir command in which we created **three (03) collections** : **workers, winegrowers and suppliers**.

```
> show dbs
admin      0.000GB
config     0.000GB
local      0.000GB
projet     0.000GB
workers    0.000GB
workersdb  0.000GB
> use projet
switched to db projet
> show collections
suppliers
winegrowers
workers
```

For the pymongo script, we choose the workers entity ; we put for it the connection to the database and the ScriptMongo.py file.

3-2. MongoDB queries

```
# 1: Display the male workers.

result1 = db.workers.find({"gender" : "male"})
for document in result1:
    print(document)

# 2 : Count the number workers by gender.

result2 = db.workers.aggregate([{"$group": {"_id": "$gender", "total": { "$sum" : 1}}}])
for r in result2:
    print(r["total"], " ", r["_id"])


# 3 : Display the cities of the workers distinctly
db.workers.distinct("adress.city")


# 4 : Sort the workers by their city.
result4 = db.workers.find().sort([("adress.city", pymongo.ASCENDING)])
print(list(result4))

# 5 : Display the workers who live in the street "Rue des pucelles".
result5 = db.workers.find({"adress.street" : "RUE DES PUCELLES"})
print(list(result5))
```

Conclusion

This project consisted in creating a database named project_restaurant, related to the management of orders, staff, and supply of a restaurant. We inserted data, and executed SQL and NoSQL queries to obtain specific information from the database. We went through the preliminary steps of creating tables and attributes, normalization, and designing conceptual and logical data models based on defined assumptions. During the process, we were able to highlight the limits of the model with regard to the complexity of the relationships between tables, and the non-inclusion of non-negligible aspects in the management of a restaurant.