UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Database Management Systems, A.Y. 2018/2019
Master Degree in Computer Engineering
Master Degree in Telecommunication Engineering

Homework 3 – Logical Design

Deadline: May 19, 2019

Group Four++	Project AutoChef	
Last Name	First Name	Student Number
Bordignon	Benvenuto	1210175
Di Nardo Di Maio	Raffaele	1204879
Fabris	Cristina	1205722
Perin	Matteo	1205718
Pistilli	Davide Dravindran	1204880

Variations to the Conceptual Design

Figure 1 shows the modified version of the Entity Relationship (ER) schema. The main difference with the previous schema is that a unique internal identifier has been added to "Recipe", thus allowing the removal of the external attribute "Username" derived from "User". This identifier is, therefore, the only one needed for the instances of the entity "Recipe".

Transformation of the Entity-Relationship Schema

Figure 1 shows the entity relationship schema and *Figure 2* shows the transformed entity relationship schema. The following sections describe in detail how to obtain the transformed schema.

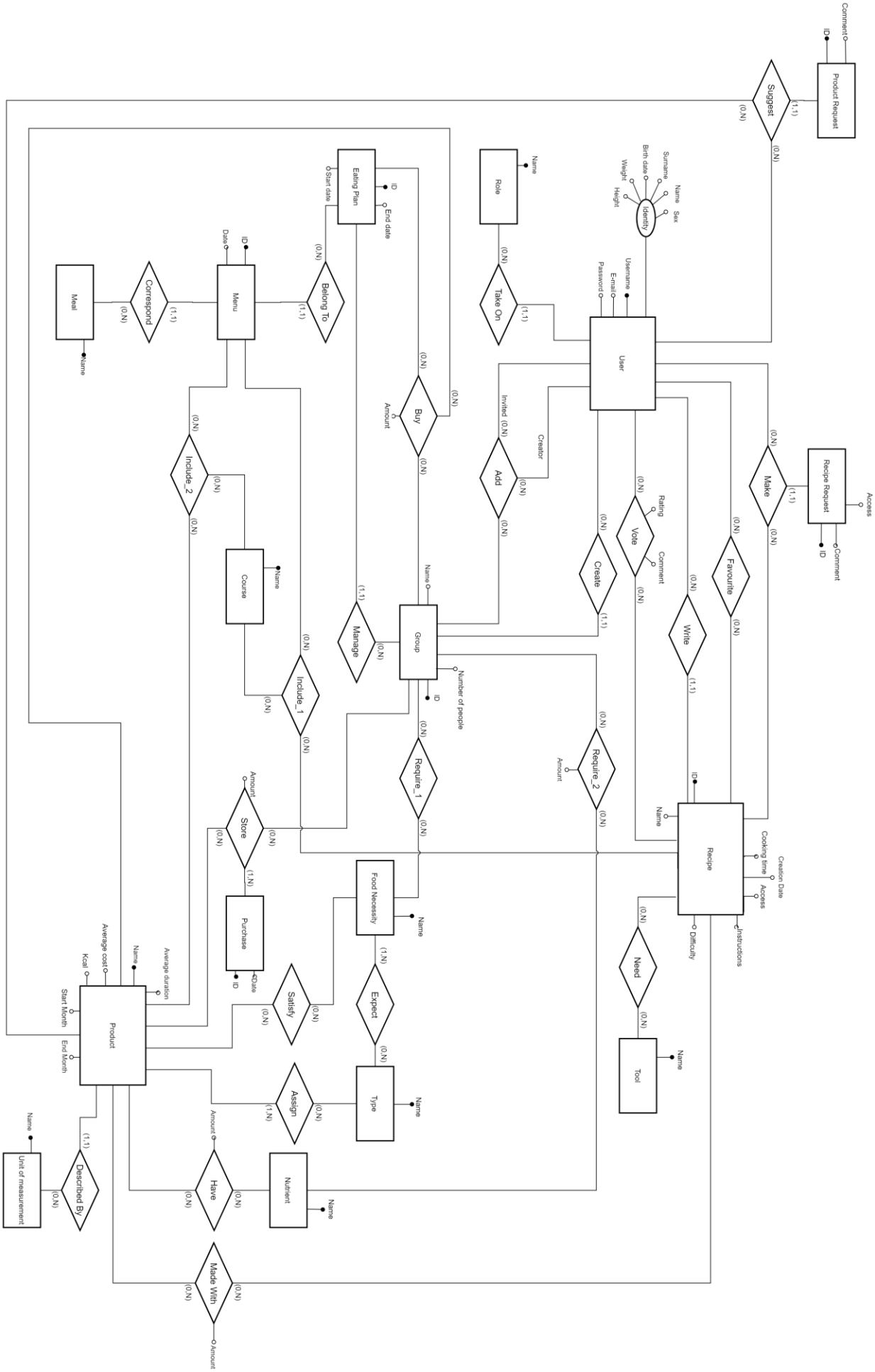

Figure 1: Variations to the ER schema

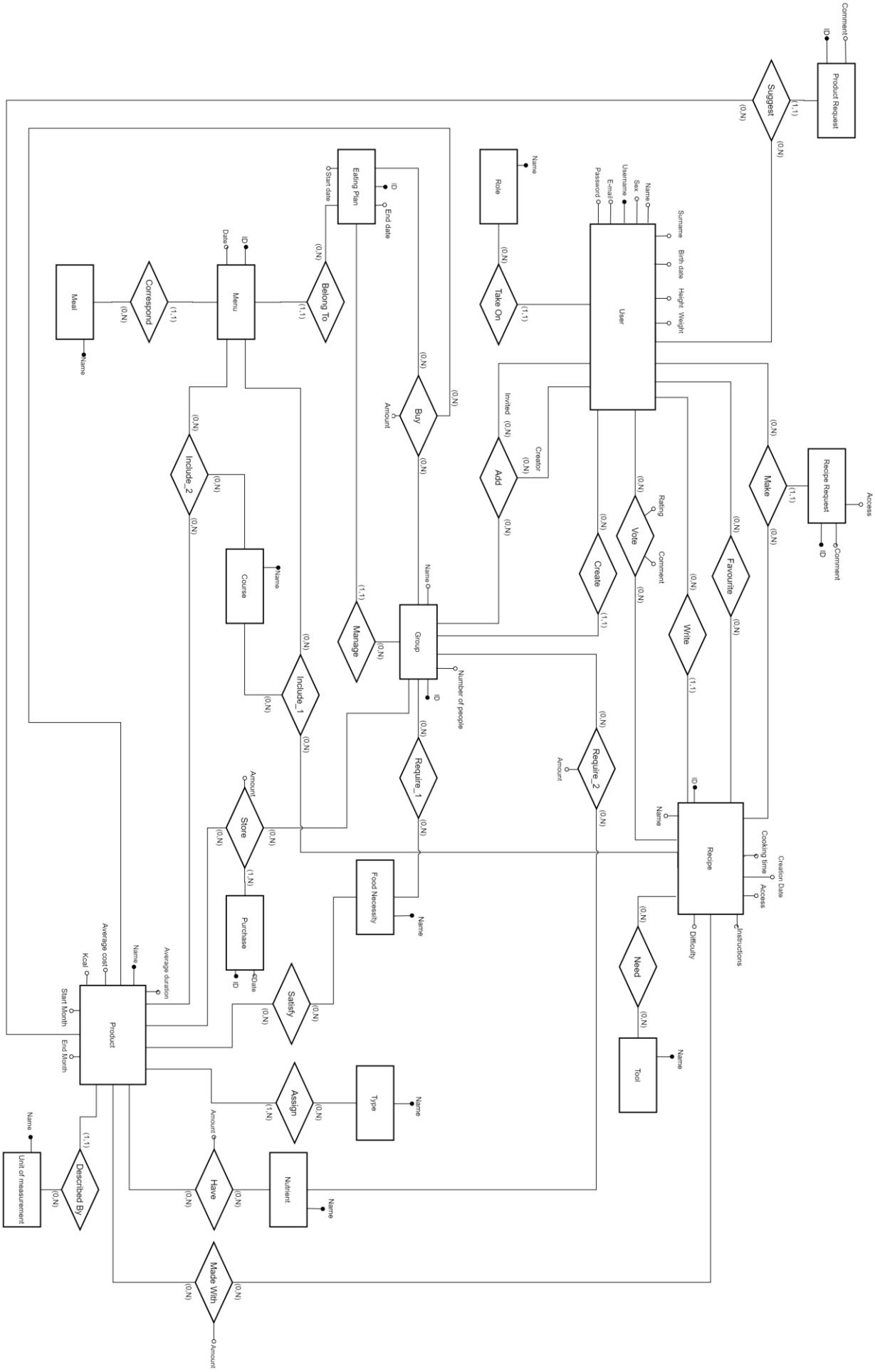

Figure 2: Transformation of the ER schema

1. Redundancy Analysis

Intensional redundancy: the schema does not contain any intensional redundancy. The two relationships "Include_1" and "Include_2" do not represent a case of redundancy since they are needed to allow the addition of just a product to a menu (for example, if a course is only composed by a product which does not need to be cooked) instead of a full-fledged recipe. Otherwise, it would be necessary to add a recipe for every single product to express the same idea, decreasing the database performance.

Extensional redundancy: The schema contains a many-to-many cycle between "Food Necessity", "Type" and "Product". This cycle can be removed by deleting the relationship "Expect" between the entities "Food Necessity" and "Type" since it only brings redundant information and it does not help improving performance because this relationship would have not been used regardless.

2. Removal of Multi-Valued Attributes

The schema does not contain any multivalued attribute.

3. Removal of Composite Attributes

The schema contains the composite attribute "Identity" in the entity "User". Since the attribute has cardinality (1, 1), we can directly associate all component attributes to the entity.

4. Removal of IS-A Relations and Generalizations

The schema does not contain IS-A relations or generalizations.

5. Choice of Principal Identifiers

The schema does not contain external identification cycles since no external identifiers are used.

6. Specification of Additional External Constraints

No external constraint is added to the schema.

7. Variations to the Data Dictionary

The following section lists the variation to the entities table of the data dictionary. The relationship table was not modified.

Entities Table

User	User of AutoChef	• E-mail, e-mail of the user, text. • Birth date, birthday of the user, date. • Height, height of the user, number. • Name, name of the user, text. • Sex, sex of the user, text. • Surname, surname of the user, text. • Weight, weight of the user, number. • Password, personal password for the authentication of the user, text. • Username, unique name that identifies a user in the	Username

		database, text.	
Recipe	Set of instructions that users can follow to transform a set of products into courses.	• Access, current state of the recipe (private or public), text. • Cooking time, minutes needed to complete the recipe, integer. • Creation Date, date in which the author created the recipe, date. • Difficulty, difficulty of the recipe (from 1 to 5), integer. • Instructions, steps to follow in order to prepare the recipe, text. • Name, name of the recipe, text. • ID, identifier of the recipe, integer.	ID

Relationships Table

No relationships were modified.

Analysis of Database Load

In this section we report the analysis of the database performance in order to justify the presence of redundancies in the entity relationship schema. We decided to consider two different operations that involve the redundant attributes "Start date" and "End date" (in the entity "Eating Plan"). These can be calculated by looking at all instances of the entity menu which are related to a specific eating plan and then by searching the menu that has the first date and the menu that has the last date.
In our analysis we consider the following operations:

- O1: Insert a new menu and associate it to an eating plan.
- O2: Show an instance of an eating plan, taking into consideration start date and end date.

Both O1 and O2 are online operations because each menu must be associated immediately with an instance of an eating plan and the data contained in it must be displayed when asked.
Table 1 reports the description of all considered operations:

Operation	Description	Frequency	Type
O1 add a new menu	Insert a new menu and associate it to an eating plan.	21/week	online
O2 show all data related to an instance of an eating plan	Show an instance of an eating plan, taking into consideration start date and end date.	7/week	online

Table 1: Frequency table

Table 2 reports the volume of data related to operation O1 with redundancy. In this case, each menu must be within Start date and End date of its corresponding eating plan:

Concept	Construct	Access	Type	Average access
Menu	Entity	1	W	1X21X2=42
Belong to	Relationship	1	W	1X21X2=42
Eating plan	Entity	1	R	1X21X1=21
Total Access			105	

Table 2: Volume of data related to operation O1 with redundancy

Table 3 reports the data volume of operation O1 without redundancy:

Concept	Construct	Access	Type	Average access
Menu	Entity	1	W	1X21X2=42
Belong to	Relationship	1	W	1X21X2=42
Total access			84	

Table 3: data volume of operation O1 without redundancy

Table 4 reports the data volume of operation O2 with redundancy:

Concept	Construct	Access	Type	Average access
Eating Plan	Entity	1	R	1X7X1
Total access			7	

Table 4: data volume of operation O2 with redundancy

Table 5 reports the data volume of operation O2 without redundancy:

Concept	Construct	Access	Type	Average access
Menu	Entity	1	R	1X7X1=7
Belong to	Relationship	21	R	21X7X1=147
Total access			154	

Table 5: data volume of operation O2 without redundancy.

In *Table 6* we finally compare the database performance both with and without the redundant attributes. We can see that, while operation O1 suffers a bit from the introduction of redundancy, operation O2's performance improves by a huge margin. This means that it is better to keep the redundant attributes.

operation	With redundancy	Without redundancy
O1	105	84
O2	7	154
total	112	238

Table 6: performance comparison with and without redundancy

Relational Schema

Figure 3 shows the relational schema. This was derived directly from the transformed ER schema shown in *Figure 2* by using only immediate transformations.

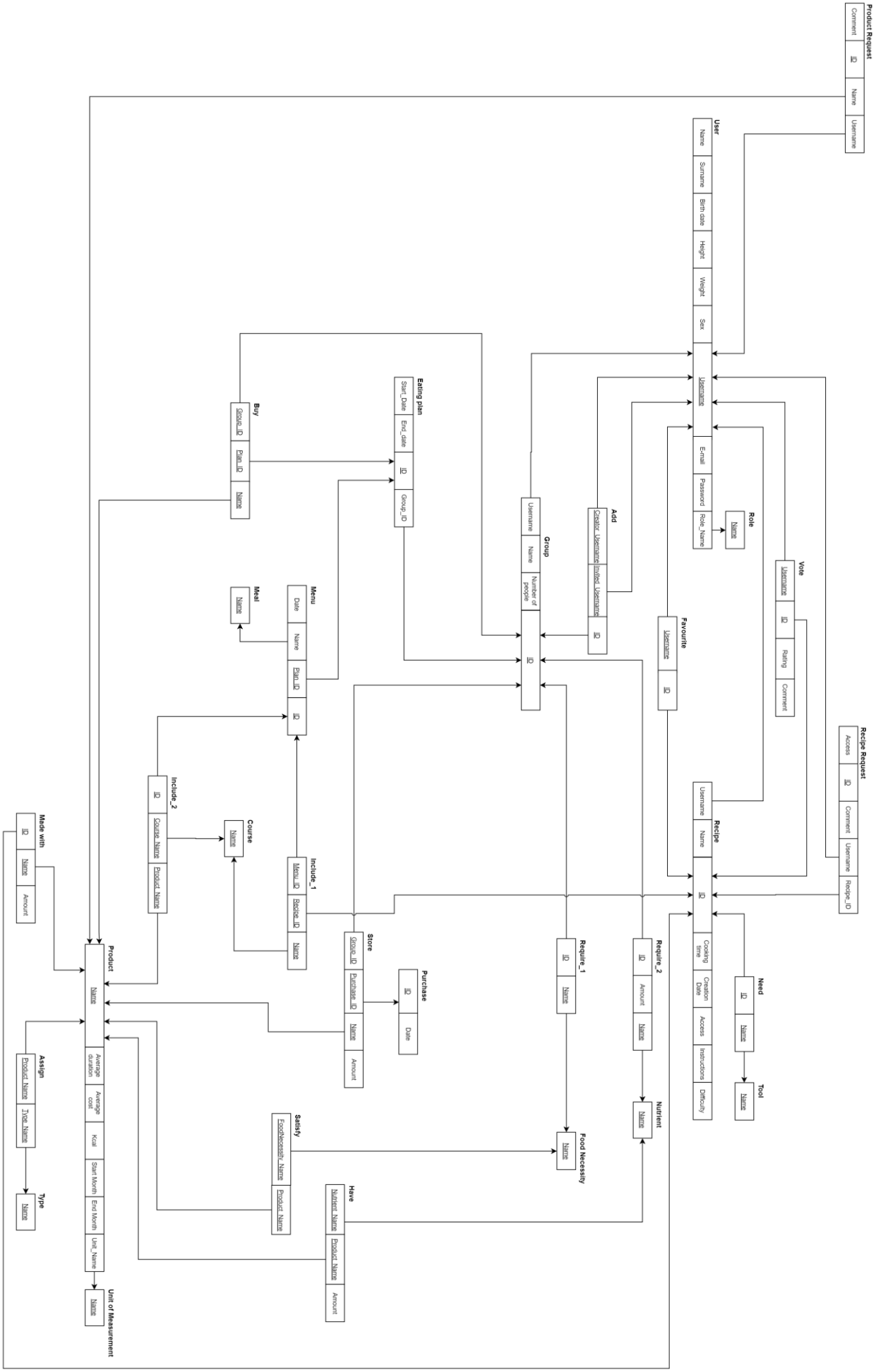

Figure 3: Relational Model

Normalization of the Relational Schema

The relational schema depicted in *Figure 3* is already normalized up to the Boyce-Codd normal form:

- **First Normal Form:** Composite and multivalued attributes have been removed during the transformation of the ER Schema and this guarantees the 1NF.
- **Second Normal Form:** By analysing the relations with a primary key composed by more than one attribute we can see that every non-prime attribute is totally dependent on the key. In fact, the only relations containing a key with more than one attribute along with non-prime attributes are the ones containing the attribute Amount, which in each instance is fully functional dependent on the respective key attributes and the relations Vote and Menu, which attributes (Rating, Comment and Date, Name respectively) have both full functional dependence on the key. This ensures the 2NF.
- **Third Normal Form:** The 3NF is achieved, since there are no transitive dependencies in the relational schema. In fact, there are no functional dependencies between the attributes of each relation.

Data Dictionary

Relation	Attribute	Description	Domain	Constraints
Product request	ID	Identifier of the product request	Integer	Primary key
	Comment	Description of the product that a user wants to add	Text	Not NULL
	Name	Name of the product	Text	Foreign key to Product, Not NULL
	Username	Unique name that identifies a user in the database	Text	Foreign key to User, Not NULL
Role	Name	Name that identifies the role of a user (Subscriber, Moderator, Banned)	Text	Primary key
User	E-mail	E-mail of the user	Text	Not NULL
	Birth Date	Birthday of the user	Date	Not NULL
	Height	Height of the user	Real	Not NULL
	Name	Name of the user	Text	Not NULL
	Sex	Sex of the user	Text	Not NULL
	Surname	Surname of the user	Text	Not NULL
	Weight	Weight of the user	Real	Not NULL
	Password	Personal password for user authentication	Text	Not NULL
	Username	Unique name that identifies a user in the database	Text	Primary key
	Role_name	Name that	Text	Foreign key to Role,

		identifies the role of a user (Subscriber, Moderator, Banned)		Not NULL
Recipe Request	Access	Current state of the recipe (private or public)	Text	Not NULL
	Comment	Details of the request	Text	Not NULL
	ID	Identifier of the recipe request	Integer	Primary key
	Username	Unique name that identifies a user in the database	Text	Foreign key to User, Not NULL
	Recipe_ID	Identifier of a recipe	Integer	Foreign key to Recipe, Not NULL
Recipe	Access	Current state of the recipe (private or public)	Text	Not NULL
	Cooking Time	Minutes needed to complete the recipe	Integer	Not NULL
	Creation Date	Date in which the author created the recipe	Date	Not NULL
	Difficulty	Difficulty of the recipe (from 1 to 5)	Integer	Not NULL
	Instructions	Steps to follow in order to prepare the recipe	Text	Not NULL
	Name	Name of the recipe	Text	Not NULL
	Username	Username of the author, from the entity User	Text	Foreign key to User, Not NULL
	ID	Identifier of a recipe	Integer	Primary key
Need	ID	Identifier of a recipe	Integer	Foreign key to Recipe, Not NULL, Primary key with Name
	Name	Name of the tool needed for the recipe	Text	Foreign key to Tool, Not NULL, Primary key with ID
Tool	Name	Name of the tool	Text	Primary key
Favourite	Username	Unique name that identifies a user in the database	Text	Foreign key to User, Not NULL, Primary key with ID
	ID	Identifier of a recipe	Integer	Foreign key to Recipe, Not NULL, Primary key with Username

Vote	Rating	Number of stars (1-5)	Integer	Not NULL
	Username	Unique name that identifies a user in the database	Text	Foreign key to User, Not NULL, Primary Key with ID
	ID	Identifier of a recipe	Integer	Foreign key to Recipe, Not NULL, Primary key with Username
	Comment	A short review of the recipe	Text	Not NULL
Add	Creator_Username	Unique name that identifies a user (the creator of the group) in the database	Text	Foreign key to User, Not NULL, Primary key with Invited_Username with ID
	Invited_Username	Unique name that identifies a user in the database	Text	Foreign key to User, Not NULL, Primary key with Creator_Username and ID
	ID	Identifier of the group	Integer	Foreign key to Group, Not NULL, Primary key with Creator_Username and Invited_Username
Buy	Group_ID	Identifier of the group	Integer	Foreign key to Group, Not NULL, Primary key
	Plan_ID	Identifier of the eating plan	Integer	Foreign key to Eating Plan, Not NULL, Primary key
	Name	Name of the product	Text	Primary key
Menu	Date	Day in which the menu will be prepared	Date	Not NULL
	Name	Name that identifies the meal	Text	Foreign key to Meal
	Plan_ID	Identifier of the eating plan	Integer	Foreign key to Eating Plan, Not NULL, Primary key with ID
	ID	Identifier of the menu		Primary key
Include_1	Menu_ID	Identifier of the menu	Integer	Foreign key to Menu, Not NULL, Primary key with Recipe_ID and Name
	Recipe_ID	Identifier of the recipe	Integer	Foreign key to Recipe, Not NULL, Primary key with Menu_ID and Name
	Name	Name of the	Text	Foreign key to

		course		Course, Not NULL, Primary key with Menu_ID and Recipe_ID
Include_2	ID	Identifier of the menu	Integer	Foreign key to Menu, Not NULL, Primary key with Course_Name and Product_Name
	Course_Name	Name of the course	Text	Foreign key to Course, Not NULL, Primary key with ID and Product_Name
	Product_Name	Name of the product	Text	Foreign key to Product, Not NULL, Primary key with ID and Course_Name
Unit of Measurement	Name	Name of the unit of measurement	Text	Primary key
Nutrient	Name	Name of the nutrient	Text	Primary key
Made with	ID	Identifier of the recipe	Text	Foreign key to Recipe, Not NULL, Primary key with Name
	Name	Name of the product	Text	Foreign key to Product, Not NULL, Primary key with ID
	Amount	Quantity of a product needed for the recipe	Real	Not NULL
Require_1	ID	Identifier of the group	Integer	Foreign key to Group, Not NULL, Primary key with Name
	Name	Name of the allergy or of the food preference	Text	Foreign key to Food Necessity, Not NULL, Primary key with ID
Require_2	ID	Identifier of the group	Integer	Foreign key to Group, Not NULL, Primary key with Name
	Name	Name of the nutrient	Text	Foreign key to Nutrient, Not NULL, Primary key with ID
	Amount	Quantity of nutrient needed by a group	Real	Not NULL
Food Necessity	Name	Name of the allergy or the food preference	Text	Primary key
Satisfy	FoodNecessity_Name	Name of the allergy or the food preference	Text	Foreign key to Food Necessity, Not NULL, Primary key with Product_Name
	Product_Name	Name of the	Text	Foreign key to

		product		Product, Not NULL, Primary key with FoodNecessity_Name
Store	Purchase_ID	Identifier of the purchase	Integer	Foreign key to Purchase, Not NULL, Primary key with Group_ID and Name
	Name	Name of the product	Text	Foreign key to Product, Not NULL, Primary key with Purchase_ID and Group_ID
	Group_ID	Identifier of the group	Integer	Foreign key to Group, Not NULL, Primary key with Purchase_ID and Name
	Amount	Quantity of the product stored after a purchase	Real	Not NULL
Assign	Type_Name	Name of the type of food	Text	Foreign key to Type, Not NULL, Primary key with Product_Name
	Product_Name	Name of the product	Text	Foreign key to Product, Not NULL, Primary key with Type_Name
Have	Nutrient_Name	Name of the nutrient	Text	Foreign key to Nutrient, Not NULL, Primary key with Product_Name
	Product_Name	Name of the product	Text	Foreign key to Product, Not NULL, Primary key with Nutrient_Name
	Amount	Quantity of a nutrient in a product	Real	Not NULL
Product	Name	Name of the product	Text	Primary key
	Average_Duration	Average expiration time of the product	Integer	Not NULL
	Average_Cost	Average cost of the product	Real	Not NULL
	Kcal	Amount of kcal of the product	Real	Not NULL
	Start_Month	First month in which the product is available	Integer	Not NULL
	End_Month	Last month in which the product is available	Integer	Not NULL
	Unit_Name	Name of the unit	Text	Foreign key to Unit of

		of measurement		Measurement, Not NULL
Course	Name	Name of the course	Text	Primary key
Meal	Name	Name that identifies the meal	Text	Primary key
Group	Name	Name of the group	Text	Not NULL
	Number of people	Number of participants of the group	Integer	Not NULL
	ID	Identifier of the group	Integer	Primary key
	Username	Unique name that identifies a user in the database	Text	Not NULL
Eating plan	Start_Date	Date in which the plan begins	Date	Not NULL
	End_Date	Date in which the plan ends	Date	Not NULL
	ID	Identifier of the eating plan	Integer	Primary key
	Group_ID	Identifier of the group	Integer	Foreign key to Group, Not NULL
Type	Name	Name of the type of food	Text	Primary key
Purchase	ID	Identifier of the purchase	Integer	Primary key
	Date	Date in which the purchase is made	Date	Not NULL

External Constraints

1. A single user cannot write more than one recipe with the same identical name. Therefore, it must be checked that there is not a recipe with the same Name and Username when adding a tuple within Recipe.
2. A tuple must be added automatically in Purchase when a user adds items to one of their groups' pantry. Date is set to the current time and date.
3. A group cannot have two overlapping eating plans, this must be ensured by checking if there are two tuples in Eating Plan related to the same group with different IDs and with overlapping Start_Date and End_Date.
4. There cannot be two tuples in User with the same E-mail and different Username (a single email can only be used for one subscription).
5. When a user registers to AutoChef, the application automatically creates a tuple in Group with attributes set to the user's username in Username and Name and Number of People to 1. This is the user's personal group, which they can use to manage their personal eating plan. This tuple ID cannot be present in Add (a user cannot add more people to his personal group).
6. A user can only favourite recipes that he did not write in order to avoid data duplication. This is ensured by checking if there exists a tuple with the same values for Username and ID in both Recipe and Favourite.

7. Only moderators should be able to see the data inside the relations Recipe Request and Product Request. Each user should only be able to see eating plan and pantry data associated with his account or his groups, this can be done by checking that the respective IDs are linked to an ID in the relation Group or Add that has the correct Username or Invited_Username.