CASE METHOD: PLYTIX

DATABASES
UNIVERSITY OF MÁLAGA

GROUP N

FAZELLO, SIMONE
FERNÁNDEZ MÉNDEZ, SALVADOR
LEE, CHAEEUN

24/25

1. INTRODUCTION

This memory describes the workflow and the decisions made during the development of the Method Case project for the Databases´s course, offered in the first semester, second year of the academic curriculum for the Degree in ‘Ingeniería Informática’ at the Universidad of Málaga.

2. DISTRIBUTION OF TASKS

We met in a working room below the library of the ‘Escuela de Ingeniería Informática’. Firstly, we decided to create a Github´s repository so as to share our work. Then, the EER model was created in SQL Developer Data Modeler (more details in the next section) by all three of us in collaboration, as well as engineered into a Relational model.

Once we finished the meeting, we agreed on this distribution of the remaining tasks:

- Simone Fazello would apply the algorithm seen in class to obtain the Relational model by hand.

- Salvador Fernández would review the EER model and make any necessary changes. He will also be in charge of exporting the project properly to meet the optional goals.

- Chaeeun Lee would write the memory of the project.

3. DEVELOPMENT OF EER MODEL

- NOTE: we made these decisions based on the provided material, i.e., the slides and the audio of the presentation of Plytix.

  Plytix´s business plan relies on offering a platform for small to medium sized enterprises, so that they can store their products' information.

  That's why we started the EER model by defining the entity ‘Product’ . All characteristics defined in the presentation were added as attributes. We decided that the primary key should be **Name**, as it would be easier to operate with, whereas the SKU is more abstract and does not indicate any characteristic of the product. However, we decided that the **SKU** should be mandatory because it is a good practice for keeping the products located.

  In addition, this focus on the product allowed us to easily define the following relationships and constraints:

  - A product is **similar** to another product when they usually are bought together
  - A product belongs to a **category** in order to group a certain group of products.
  - A product has **attributes**, which are also related to an **account**. Each account cannot have more than 5 defined attributes (check constraint)
  - A product contains an **asset**, which is also related to an account.
  - Finally, a product is created by an **account**

We then defined the entity ‘User’, which has two subentities: ‘Owner’ (owner of account) and ‘Agent’ (can emulate other accounts). An owner **manages** an account, and its PK is Payment Access, as the owner pays for the account. Also, the user **has access** to an account. The PK for User is Email.

Lastly, each account **has different functionalities** depending on their ‘Payment Plan’. The PK for said entity is Type (3 different values, check constraint) and their attributes are those specified on the Plytix’s presentation.