BS DEGREE IN INFORMATICS ENGINEERING Academic year: 2023/24 - 2nd year, 2nd term Subject: File Structures and Databases First Assignment's Report: Relational Design and Impl.	uc3m Universidad Carlos III de Madrid

Lecturers:	**Francisco Javier Calle Gómez**		
Group:	**89**	**Lab User**	**fsdb279**
Student:	**Ines Guillén Peña**	**NIA:**	**100495752**
Student:	**Salvador Ayala Iglesisas**	**NIA:**	**100495832**
Student:	**Eduardo Alarcón Navarro**	**NIA:**	**100472175**

1 Introduction

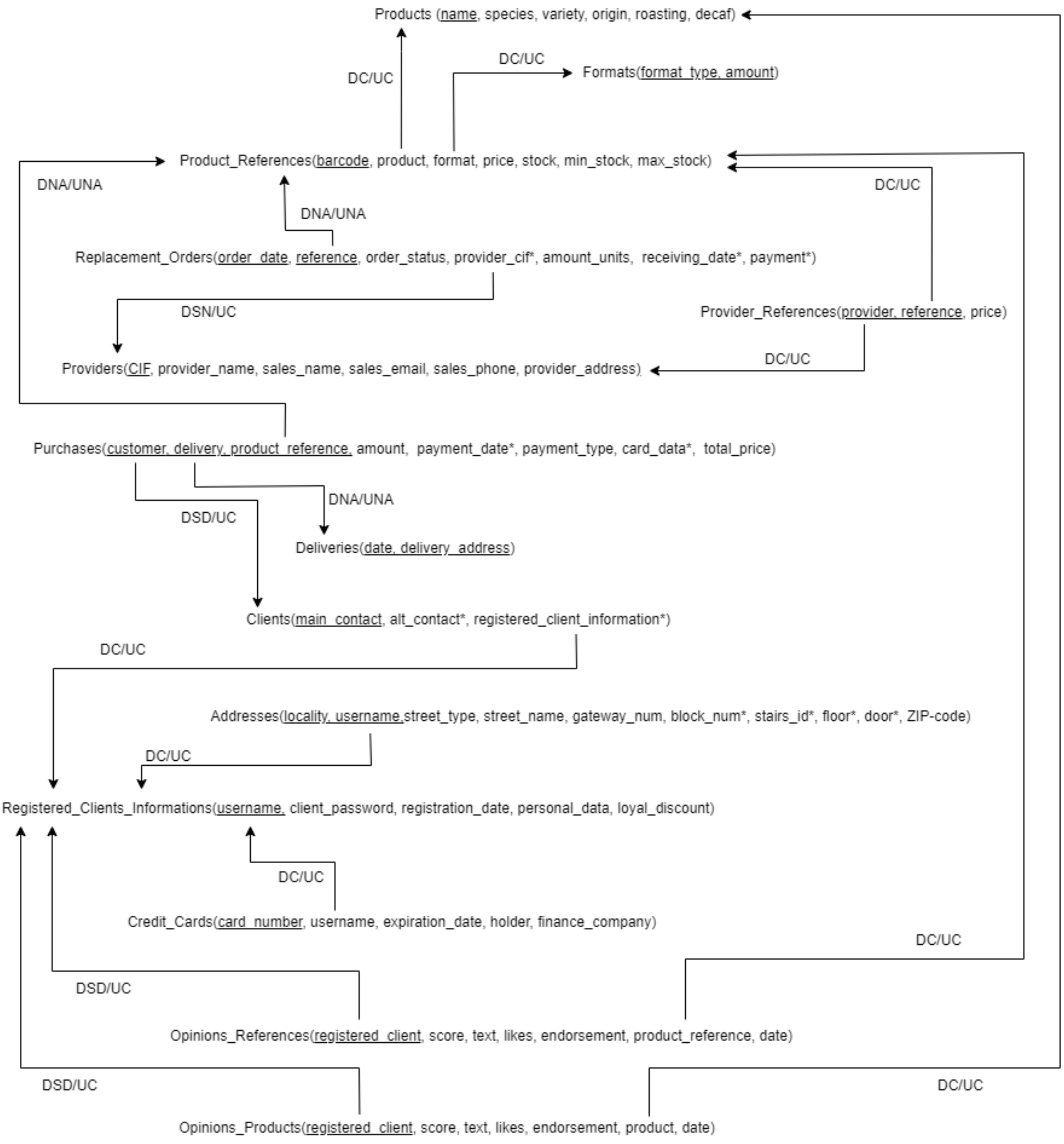
In this document, we will develop a proper relational design along with the necessary justifications to address the stated problem. This practice aims to create a database that can effectively store all relevant information related to the problem described in the statement.

To accomplish this task, we will first design a relational model for our database and implement it in SQL (creation.sql). Furthermore, we will execute a large-scale upload of data from the old database (upload.sql). During this process, some errors may arise. Our objective is to identify these errors and, if possible, fix them.

The document's structure will consist of three distinct sections: the relational design, the SQL implementation design, and lastly, a table containing the data extracted from the outdated database.

2 Relational Design

- Relational schema:


Products (name, species, variety, origin, roasting, decaf)
DC/UC
DC/UC
Formats(format_type, amount)
Product_References(barcode, product, format, price, stock, min_stock, max_stock)
DNA/UNA
DC/UC
DNA/UNA
Replacement_Orders(order_date, reference, order_status, provider_cif*, amount_units, receiving_date*, payment*)
DSN/UC
Provider_References(provider, reference, price)
DC/UC
Providers(CIF, provider_name, sales_name, sales_email, sales_phone, provider_address)
Purchases(customer, delivery, product_reference, amount, payment_date*, payment_type, card_data*, total_price)
DNA/UNA
DSD/UC
Deliveries(date, delivery_address)
Clients(main_contact, alt_contact*, registered_client_information*)
DC/UC
Addresses(locality, username,street_type, street_name, gateway_num, block_num*, stairs_id*, floor*, door*, ZIP-code)
DC/UC
Registered_Clients_Informations(username, client_password, registration_date, personal_data, loyal_discount)
DC/UC
Credit_Cards(card_number, username, expiration_date, holder, finance_company)
DC/UC
DSD/UC
Opinions_References(registered_client, score, text, likes, endorsement, product_reference, date)
DSD/UC
DC/UC
Opinions_Products(registered_client, score, text, likes, endorsement, product, date)

- Implicit semantics:

Presp_id	Stage	Mechanism	Description
I_1	Design	Primary key	Products are identified by their name.
I_2	Design	Primary key	Providers are identified by their CIF.

I_4	Design	Primary key	Credit Cards are identified by their card number.
I_5	Design	Primary key	Product references are identified by their barcode.
I_6	Design	Primary key	Purchases are identified by the customer, associated delivery and product reference.
I_7	Design	Primary key	Deliveries are identified by the order date and delivery address.
I_8	Design	Primary key	Replacement orders are identified by the product reference and the order date.
I_9	Design	Primary key	Registered addresses are identified by their associated client and the locality.
I_{10}	Design	Semantic	Locality is defined as a string of the format "City, Country".
I_{11}	Design	Primary key	The provider's offers are identified by the provider and the reference of the product they offer.
I_{12}	Design	Definition	A general table of clients will be created, with the possibility of associating a username that identifies a registered client.
I_{13}	Design	Definition	For client opinions, two different tables are created, one for product references and one for general products.
I_{14}	Design	Primary key	Opinions are identified by username, reference/product and date of the post.

Table 1: Implicit semantics incorporated into the relational graph

- Non-observed explicit semantics:

Presp_id	**Description**
S_1	When the available amount is less than a certain minimum threshold, a new replacement order will be automatically registered.
S_2	Already placed orders can't be deleted, and shouldn't be updated.
S_3	Fulfilled orders can't be deleted or updated.
S_4	Draft orders can be deleted or updated.
S_5	If there is not enough stock of a given reference, the quantity will be set to the maximum available and a message will be generated to inform the user of such event.

S_6	If a registered user wants to be deregistered, all their purchases will be converted to anonymous buyer's purchases and then all data will be automatically deleted.
S_7	The voucher will only be valid for the following 30 days.
S_8	If a user unsubscribes, their comments remain, but anonymized.
S_9	Next time the automatic process is run, it will set the order for filling the stocks up to the maximum threshold (by default, 10 units higher than the minimum stock).
S_{10}	The providers have a unique name, CIF, salesperson's full-name and email, phone number and commercial address.

Table 2: Non-observed explicit semantics

3 Relational Statics Implementation in SQL (DDL)

SQL Oracle code of the creation of tables can be found in creation.sql

- Re-incorporated semantics: "Has not been reported."
- Incorporated implicit semantics:

Presp_id	Stage	Mechanism	Description
I_{15}	Design	Check	Products attribute "decaffeinated" should be either 0 (for false) or 1 (for true).
I_{16}	Design	Check	The stock of a product reference should never be lower than 0.
I_{17}	Design	Check	Provider's CIF should be composed of 9 characters, in the format of a letter, 8 digits and a letter.
I_{18}	Design	Check	Provider's phone number should only be composed of numbers.
I_{16}	Design	Check	Any price should never be lower than 0.
I_{17}	Implem.	Check	Products' roasting type are exclusively 'natural', 'high-roast' or 'mixture'.
I_{18}	Implem.	Check	There are different marketing formats: raw grain, roasted beans, ground, freeze-dried (soluble), in capsules, or prepared.
I_{19}	Implem.	Check	Replacement orders have different states: 'draft', 'placed' or 'fulfilled'.

Table 1(cont.): implicit semantics incorporated in the definition of each table

- Excluded semantics:

Presp_id	Description	Cause	Explicit/Implicit
E_1	Whenever a purchase is made, the available units of the involved references should be updated.	PL/SQL does not observe this integrity option.	Explicit
E_2	When the available amount is less than a certain minimum threshold, a new replacement order will be automatically registered.	PL/SQL does not observe this integrity option.	Implicit
E_3	There can be only one non-fulfilled replacement order per reference.	PL/SQL does not observe this integrity option.	Explicit
E_4	There will not be more than one order placed per day and reference.	PL/SQL does not observe this integrity option.	Explicit
E_5	If a registered user wants to be deregistered, all their purchases will be converted and then all data will be automatically deleted.	PL/SQL does not observe this integrity option.	Explicit

Table 5: explicit semantics excluded in the creation of each table

4 Workload (DML)

The specific order of tables we've chosen is just one of many configurations available to us. Firstly, we first dump data into independent tables, which are tables without foreign keys. Then, we insert data into tables whose foreign keys reference tables where data have already been inserted. This approach ensures that referential integrity is maintained, as values of foreign keys can only be inserted into the child table if they exist in the parent table.

During this process, we encountered several issues:

- Since the decimal price data was with dots instead of commas, several errors have arisen when carrying out operations as Oracle works with commas. We have solved this problem by changing the character.
- Errors with the data type. When making foreign keys of type char, ORACLE automatically fills in blank spaces on the right with variable data. We have fixed this problem by declaring the field varchar2.

- When creating dependencies between tables, we realized that these table dependencies were part of the PK, which has forced us to duplicate columns in the destination tables and we have not been able to resolve this duplication that exists in both tables.
- Some of the attributes regarding the products, for example the format and the roasting types were slightly different from the statements, thus forcing us to add a case to these inserts, to fix these specific rows.
- We encountered an issue with the barcode of some products and the provider taxID as they have had price modifications in the history. For this, we chose the minimum of the prices.
- To select the addresses, we decided to create one attribute that would enclose all the fields that defined a direction, separated by commas where needed.
- We assumed the billing addresses and the delivery addresses are the same, thus removing the billing address from the schema.
- When inserting into the Purchases table, we noticed there was an error with the barcode "Q Q77433Q270983%" containing a space. This is the only one that contains this space and it creates an error in the insert. To prevent this error, we separated it from the rest, excluding it from the insert. As in the table fsdb.trolley, the product it is referring to is "Removed References" we chose to leave it out.
- We also found out that the credit cards had variable length, and are not the standard 16 numbers long, as well as being padded with white spaces.