A5: Relational Schema, validation and schema refinement

This artifact contains the Relational Schema obtained by mapping from the Conceptual Data Model. The Relational Schema includes the relation schema, attributes, domains, primary keys, foreign keys and other integrity rules: UNIQUE, DEFAULT, NOT NULL, CHECK.

1. Relational Schema

Relation schemas are specified in the compact notation:

Relation reference	**Relation Compact Notation**
R01	users(**id**, name NN, username UK NN, email UK NN, NIF UK, password NN)
R02	addresses(**id**, name NN, street NN, postal_code NN, city_id → cities NN, user_id → users NN)
R03	faqs(**id**, question UK NN, answer NN)
R04	purchases(**id**, date NN DF Today, total NN CK total > 0, user_id → users NN, address_id → addresses NN, status NN CK status IN Purchase_Status DF 'Processing')
R05	delivery_types(**id**, name UK NN, cost NN CK cost >=0)
R06	product_carts(**product_id** → products, **user_id** → users, quantity NN CK quantity > 0)
R07	products(**id**, name NN UK, price NN CK price > 0, quantity_available NN CK quantity_available >= 0, score NN CK score >= 0 AND score <= 5, category_id → categories NN)
R08	product_purchases(**product_id** → products, **purchase_id** → purchases, quantity NN CK quantity > 0, price NN CK price > 0)
R09	photos(**id**, path NN, product_id → products NN)

R10	categories(**id**, name UK NN, is_navbar_category NN DF FALSE)
R11	properties(**id**, name UK NN)
R12	category_properties(**id**, category_id → categories NN, property_id → properties NN, is_required_property NN DF FALSE)
R13	reviews(**user_id** → users, **product_id** → products, score NN CK score >= 0 AND score <= 5, title NN, content NN)
R14	wishlists(**user_id** → users, **product_id** → products)
R15	admins(**user_id** → users)
R16	archived_products(**product_id** → products)
R17	values_lists(**id**, (category_property_id → category_properties NN, product_id → products NN) UK)
R18	values(**id**, name, values_list_id → values_lists NN)
R19	countries(**id**, name NN UK)
R20	cities(**id**, name NN UK,country_id → countries NN)

Note: UK means UNIQUE KEY, NN means NOT NULL, DF means DEFAULT and CK means CHECK.

2. Domains

The specification of additional domains can also be made in a compact form, using the notation:

Domain Name	**Domain Specification**
Purchase_Status	ENUM (‘Processing’, ‘Shipped’, ‘Delivered’)
Today	DATE DEFAULT CURRENT_DATE

3. Functional Dependencies and schema validation

To validate the Relational Schema obtained from the Conceptual Model, all functional dependencies are identified and the normalization of all relation schemas is accomplished.

Table R01 (users)	
Keys: { id }, { username }, { email }, { NIF }	
Functional Dependencies	
FD0101	{ id } :- { name, username, email, NIF, password }
FD0102	{ username } :- { id, name, email, NIF, password }
FD0103	{ email } :- { id, username, name, NIF, password }
FD0104	{NIF} :- {id, username, email, password}
Normal Form	BCNF

Table R02 (addresses)	
Keys: { id }	
Functional Dependencies	
FD0201	{ id } :- { name, street, postal_code, city_id, user_id }
Normal Form	BCNF

Table R03 (faqs)	
Keys: { id }, { question }	
Functional Dependencies	
FD0301	{ id } :- { question, answer }
FD0302	{ question } :- { id, answer }
Normal Form	BCNF

Table R04 (purchases)	
Keys: { id }	
Functional Dependencies	
FD0401	{ id } :- { date, total, user_id, address_id, status }
Normal Form	BCNF

Table R05 (delivery_types)	
Keys: { id }, { name }	
Functional Dependencies	
FD0501	{ id } :- { name, cost }
FD0502	{ name } :- { id, cost }
Normal Form	BCNF

Table R06 (product_carts)	
Keys: { product_id, user_id }	
Functional Dependencies	
FD0601	{ product_id, user_id } :- { quantity }
Normal Form	BCNF

Table R07 (products)	
Keys: { id }	
Functional Dependencies	
FD0701	{ id } :- { name, price, quantity_available, score, category_id }
Normal Form	BCNF

Table R08 (product_purchases)	
Keys: { product_id, purchase_id }	
Functional Dependencies	
FD0801	{ product_id, purchase_id } :- { quantity, price }
Normal Form	BCNF

Table R09 (photos)	
Keys: { id }	
Functional Dependencies	
FD0901	{ id } :- { path, product_id }
Normal Form	BCNF

Table R10 (categories)	
Keys: { id }, { name }	
Functional Dependencies	
FD1001	{ id } :- { name, is_navbar_category }
FD1002	{ name } :- { id, is_navbar_category }
Normal Form	BCNF

Table R11 (properties)	
Keys: { id }, { name }	
Functional Dependencies	
FD1101	{ id } :- { name }
FD1102	{ name } :- { id }
Normal Form	BCNF

Table R12 (category_properties)	
Keys: { id }	
Functional Dependencies	
FD1201	{ id } :- { category_id, property_id, is_required_property }
Normal Form	BCNF

Table R13 (reviews)	
Keys: { user_id, product_id }	
Functional Dependencies	
FD1301	{ user_id,product_id} :- { score, title, content }
Normal Form	BCNF

Table R14 (wishlists)	
Keys: { user_id , product_id }	
Normal Form	BCNF

Table R15 (admins)	
Keys: { id }	
Normal Form	BCNF

Table R16 (archived_products)	
Keys: { id }	
Normal Form	BCNF

Table R17 (values_lists)	
Keys: { id },{category_id, product_id }	
Functional Dependencies	
FD1701	{ id } :- { category_id, product_id }
FD1702	{category_id, product_id } :- { id }
Normal Form	BCNF

Table R18 (values)	
Keys: { id }	
Functional Dependencies	
FD1801	{ id } :- { name, values_list_id }
Normal Form	BCNF

Table R19 (countries)	
Keys: { id }, { name }	
Functional Dependencies	
FD1901	{ id } :- { name }
FD1902	{ name } :- { id }
Normal Form	BCNF

Table R20 (cities)	
Keys: { id },{ name }	
Functional Dependencies	
FD2001	{ id } :- { name, country_id }
FD2002	{ name } :- { id, country_id }
Normal Form	BCNF

As all relations schemas are in the Boyce–Codd Normal Form (BCNF), the relational schema is also in the BCNF and therefore there is no need to be refined using normalisation.

4. SQL Code

Database SQL script link here
(https://github.com/literallysofia/lbaw1761/blob/master/artefacts/a5/database.sql).

```
CREATE DOMAIN "Today" AS date NOT NULL DEFAULT ('now'::text)::date;

CREATE TABLE addresses (
    id integer PRIMARY KEY,
    name text NOT NULL,
    street text NOT NULL,
    "postal_code" text NOT NULL,
    "city_id" integer NOT NULL REFERENCES cities(id) ON DELETE CASCADE,
    "user_id" integer NOT NULL REFERENCES users(id) ON DELETE CASCADE
);

CREATE TABLE admins (
    "user_id" integer PRIMARY KEY REFERENCES users(id) ON DELETE CASCADE
);

CREATE TABLE archived_products (
    "product_id" integer PRIMARY KEY REFERENCES products(id) ON DELETE CAS
);

CREATE TABLE categories (
    id integer PRIMARY KEY,
    name text NOT NULL UNIQUE,
    "is_navbar_category" boolean DEFAULT false NOT NULL
);

CREATE TABLE category_properties (
    id integer PRIMARY KEY,
    "category_id" integer NOT NULL REFERENCES categories(id) ON DELETE CAS
    "property_id" integer NOT NULL REFERENCES properties(id) ON DELETE CAS
    "is_required_property" boolean DEFAULT false NOT NULL
);

CREATE TABLE cities (
    id integer PRIMARY KEY,
    name text NOT NULL UNIQUE,
    "country_id" integer NOT NULL REFERENCES countries(id) ON DELETE CASCA
);
```

```
CREATE TABLE countries (
    id integer PRIMARY KEY,
    name text NOT NULL UNIQUE
);

CREATE TABLE delivery_types (
    id integer PRIMARY KEY,
    name text NOT NULL UNIQUE,
    cost double precision NOT NULL UNIQUE
);

CREATE TABLE faqs (
    id integer PRIMARY KEY,
    question text NOT NULL UNIQUE,
    answer text NOT NULL
);

CREATE TABLE photos (
    id integer PRIMARY KEY,
    path text NOT NULL,
    "product_id" integer NOT NULL REFERENCES products(id) ON DELETE CASCADE
);

CREATE TABLE products (
    id integer PRIMARY KEY,
    name text NOT NULL,
    price double precision NOT NULL,
    quantity_available integer NOT NULL,
    score integer NOT NULL,
    "category_id" integer NOT NULL REFERENCES categories(id) ON DELETE CAS
    CONSTRAINT price CHECK ((price > (0)::double precision)),
    CONSTRAINT quantity_available CHECK ((quantity_available >= 0)),
    CONSTRAINT score CHECK (score >= 0 AND score <= 5)
);

CREATE TABLE product_carts (
    id integer PRIMARY KEY,
    "user_id" integer NOT NULL REFERENCES users(id) ON DELETE CASCADE,
    quantity integer NOT NULL,
    CONSTRAINT quantity CHECK ((quantity > 0))
);

CREATE TABLE product_purchases (
    "product_id" integer PRIMARY KEY REFERENCES products(id) ON DELETE CAS
    "purchase_id" integer NOT NULL REFERENCES purchases(id) ON DELETE CASC
    quantity integer NOT NULL,
    price double precision NOT NULL,
    CONSTRAINT price CHECK ((price > (0)::double precision)),
    CONSTRAINT quantity CHECK ((quantity > 0))
);

CREATE TABLE properties (
    id integer PRIMARY KEY,
    name text NOT NULL UNIQUE
);

CREATE TABLE purchases (
    id integer PRIMARY KEY,
    "date" TIMESTAMP WITH TIME zone DEFAULT now() NOT NULL,
    total double precision NOT NULL,
    "user_id" integer NOT NULL REFERENCES users(id) ON DELETE CASCADE,
    "address_id" integer NOT NULL REFERENCES addresses(id) ON DELETE CASCA
    status text DEFAULT 'Processing'::text NOT NULL,
    CONSTRAINT status CHECK ((TYPE = ANY (ARRAY['Processing'::text, 'Shipp
    CONSTRAINT total CHECK ((total > (0)::double precision))
);

CREATE TABLE reviews (
    "user_id" integer PRIMARY KEY REFERENCES users(id) ON DELETE CASCADE,
    "product_id" integer PRIMARY KEY REFERENCES products(id) ON DELETE CAS
    score integer NOT NULL,
    title text NOT NULL,
    content text NOT NULL,
    CONSTRAINT score CHECK (((score >= 0) AND (score <= 5)))
);

CREATE TABLE users (
    id integer PRIMARY KEY,
    name text NOT NULL,
    username text NOT NULL UNIQUE,
    email text NOT NULL UNIQUE,
    password text NOT NULL
);

CREATE TABLE values (
    id integer PRIMARY KEY,
    name text,
    "values_list_id" integer NOT NULL REFERENCES values_lists(id) ON DELET
);

CREATE TABLE values_lists (
    id integer PRIMARY KEY,
    "category_id" integer NOT NULL REFERENCES category_properties(id) ON D
    "product_id" integer NOT NULL REFERENCES products(id) ON DELETE CASCAD
);
```

```
CREATE TABLE wishlists (
    "user_id" integer PRIMARY KEY REFERENCES users(id) ON DELETE CASCADE,
    "product_id" integer PRIMARY KEY REFERENCES products(id) ON DELETE CAS
);
```

Revision history

Changes made to the first submission:

1. Changed the tables and collumns names to match Laravel Eloquent naming convention.
2. Rearranged some of the keys in the Funtional Dependencies tables.
3. Fixed some small errors in the sql file.

GROUP1761, 28/03/2018

- Bárbara Sofia Lopez de Carvalho Ferreira da Silva, up201505628@fe.up.pt (mailto:up201505628@fe.up.pt)
- Carlos Miguel da Silva de Freitas, up201504749@fe.up.pt (mailto:up201504749@fe.up.pt)
- Julieta Pintado Jorge Frade, up201506530@fe.up.pt (mailto:up201506530@fe.up.pt)
- Luís Noites Martins, up201503344@fe.up.pt (mailto:up201503344@fe.up.pt)