A5: Modelo Relacional, validação e refinamento do esquema

Neste artefacto é apresentado o modelo relacional da plataforma Bookhub, que foi obtido a partir do modelo conceptual de dados previamente elaborado. Este artefacto tem, portanto, como principais objetivos a definição do esquema relacional e dos seus atributos, domínios, chaves primárias (*Primary Keys*) , chaves estrangeiras (*Foreign Keys*) e outras regras de integridade.

1. Esquema Relacional

Para o esquema relacional foram consideradas todas as restrições presentes no esquema conceptual, excetuando algumas restrições, em particular referentes a datas, que só poderão ser exercidas recorrendo a *triggers*.

Abreviaturas:

- UK -> *UNIQUE KEY*
- NN -> *NOT NULL*
- DF -> *DEFAULT*
- CK -> *CHECK*

Referência da relação	**Notação Compacta da Relação**
R01	Administrator(**id**, username UK NN, password NN)
R02	Country(**id**, countryName UK NN)
R03	Member(**id**, address, age NN CK age>=18, email UK NN, name NN, password NN, paypalEmail, phone NN, postalCode NN, username UK NN, dateCreated NN DF Today, status DF 'normal' CK status IN MemberStatus, dateBanned, dateSuspended, dateTerminated, **idCountry** → Country NN, **idImage** → Image)
R04	RequestedTermination(**id**, dateRequested NN DF Today, **idMember** -> Member NN)
R05	Auction(**id**, author NN, description NN, duration NN CK duration>5, ISBN NN, title NN, dateCreated NN DF Today, status DF 'waitingApproval' CK status IN AuctionStatus, dateApproved, dateRemoved, **idPublisher** → Publisher, **idLanguage**-> Language NN, **idSeller** → Member NN)
R06	Category(**id**, categoryName UK NN)
R07	Publisher(**id**, publisherName UK NN)
R08	Language(**id**, language UK NN)
R09	CategoryAuction(**idCategory** -> Category, **idAuction** -> Auction)

R10	WishList(**idBuyer**-> Member, **idAuction** -> Auction)
R11	Bid(**idBuyer**-> Member, **idAuction** -> Auction, bidDate NN DF Today, bidValue CK bidValue>0.0)
R12	AuctionModification(**id**, dateRequested NN DF Today,newDescription, is_approved, dateApproved, **idApprovedAuction**-> Auction NN)
R13	Notification(**id**, dateSent NN DF Today, information, is_seen DF FALSE, dateSeen, **idMember** -> Member NN)
R14	NotificationAuction(**idAuction**->Auction, **idNotification**->Notification)
R15	Image(**id**, source UK NN, **idAuction** → Auction, **idAuctionModification**->AuctionModification)
R16	Message(**id**, dateSent NN DF Today, text NN, **idSender** → Member NN, **idReceiver** → Member NN)
R17	Comment(**id**, datePosted NN DF Today, liked, text NN, is_removed DF FALSE, **idParent** → Comment, **idSender** → Member NN, **idReceiver** → Member NN)

2. Domínios

Especificação de domínios adicionais:

Nome	**Especificação**
Today	DATE DEFAULT CURRENT_DATE
MemberStatus	ENUM ('moderator', 'suspended', 'banned', 'normal','terminated')
AuctionStatus	ENUM ('approved', 'removed', 'waitingApproval')

3. Dependências Funcionais e validação do esquema

Para validar o esquema relacional obtido através do modelo conceptual, todas as dependências funcionais estão identificadas e o nível de normalização de cada relação foi identificado:

Tabela R01 (Administrator)	
Chaves: {id}, {username}	
Dependências Funcionais	
FD0101	{id} → {username, password}
FD0102	{username} → {id, password}
Forma Normal	BCNF

Tabela R02(Country)

Chaves: {id}, {countryName}	
Dependências Funcionais	
FD0201	{ id } → {countryName}
FD0202	{ countryName } → {id}
Forma Normal	BCNF

Tabela R03 (Member)	
Chaves: {id}, {email}, {username}	
Dependências Funcionais	
FD0301	{id} → {address, age, email, name, password, paypalEmail, phone, postalCode, username, dateCreated, status, dateBanned, dateSuspended, dateTerminated, idCountry, idImage}
FD0302	{email} → {id, address, age, name, password, paypalEmail, phone, postalCode, username, dateCreated, status, dateBanned, dateSuspended, dateTerminated, idCountry, idImage}
FD0303	{username} → {id, address, age, email, name, password, paypalEmail, phone, postalCode, dateCreated, status, dateBanned, dateSuspended, dateTerminated, idCountry, idImage}
Forma Normal	BCNF

Tabela R04 (RequestedTermination)	
Chaves: {id}	
Dependências Funcionais	
FD0401	{ id } → {dateRequested,idMember}
Forma Normal	BCNF

Tabela R05 (Auction)	
Chaves: {id}	
Dependências Funcionais	
FD0501	{id} → {author, description, duration, ISBN, title, dateCreated, status, dateApproved, dateRemoved, idLanguage, idPublisher, idSeller}
Forma Normal	BCNF

Tabela R06 (Category)

Chaves: {id}, {categoryName}	
Dependências Funcionais	
FD0601	{id} → {categoryName}
FD0601	{categoryName} → {id}
Forma Normal	BCNF

Tabela R07 (Publisher)	
Chaves: {id}, {publisherName}	
Dependências Funcionais	
FD0701	{id} → {publisherName}
FD0701	{publisherName} → {id}
Forma Normal	BCNF

Tabela R08 (Language)	
Chaves: {id}, {language}	
Dependências Funcionais	
FD0801	{id} → {language}
FD0801	{language} → {id}
Forma Normal	BCNF

Tabela R09 (CategoryAuction)	
Chaves: {idCategory , idAuction}	
Dependências Funcionais	
(none)	
Forma Normal	BCNF

Tabela R10 (WishList)	
Chaves: {idBuyer , idAuction}	
Dependências Funcionais	
(none)	
Forma Normal	BCNF

Tabela R11 (Bid)	
Chaves: {idBuyer , idAuction}	
Dependências Funcionais	
FD1101	{idBuyer , idAuction} → {bidDate, bidValue}
Forma Normal	BCNF

Tabela R12 (AuctionModification)	
Chaves: {id}	
Dependências Funcionais	
D1201	{id} → {dateRequested, newDescription, is_approved, dateApproved, idApprovedAuction}
Forma Normal	BCNF

Tabela R13 (Notification)	
Chaves: {id}	
Dependências Funcionais	
FD1301	{id} → {dateSent, information, is_seen, dateSeen, idMember}
Forma Normal	BCNF

Tabela R14 (NotificationAuction)	
Chaves: {idNotification, idAuction}	
Dependências Funcionais	
(none)	
Forma Normal	BCNF

Tabela R15 (Image)	
Chaves: {id}, {source}	
Dependências Funcionais	
FD1501	{id} → {source, idAuction, idAuctionModification}
FD1502	{source} → {id, idAuction, idAuctionModification}
Forma Normal	BCNF

Tabela R16 (Message)	
Chaves: {id}	
Dependências Funcionais	
FD1601	{id} → {dateSent, text, idSender, idReceiver}
Forma Normal	BCNF

Tabela R17 (Comment)	
Chaves: { id }	
Dependências Funcionais	
FD1701	{id} → {datePosted, liked, text, is_removed, idParent, idSender, idReceiver}
Forma Normal	BCNF

Como todos os esquemas de relações estão na forma normal *Boyce–Codd Normal Form* (BCNF), o modelo relacional encontra-se inteiramente normalizado, não tendo sido necessário realizar nenhumas normalizações extra.

4. SQL Code

Ficheiro *schema.sql* (requer acesso ao repositório GitHub do projeto)

```
-----------------------------------------
--Tables
-----------------------------------------
--1

CREATE TABLE administrator (
    id SERIAL NOT NULL PRIMARY KEY,
    username text NOT NULL UNIQUE,
    password text NOT NULL
);

--2
CREATE TABLE country (
    id SERIAL NOT NULL PRIMARY KEY,
    countryName text NOT NULL UNIQUE
);

--3
CREATE TABLE member (
    id SERIAL NOT NULL PRIMARY KEY,
    address text,
    age SMALLINT NOT NULL,
    email text NOT NULL UNIQUE,
```

```
    name text NOT NULL,
    password text NOT NULL,
    paypalEmail text,
    phone text NOT NULL,
    postalCode text NOT NULL,
    username text NOT NULL UNIQUE,
    dateCreated TIMESTAMP WITH TIME zone DEFAULT now() NOT NULL,
    member_status text NOT NULL DEFAULT 'normal'::text,
    dateBanned TIMESTAMP WITH TIME zone NOT NULL,
    dateSuspended TIMESTAMP WITH TIME zone NOT NULL,
    dateTerminated  TIMESTAMP WITH TIME zone NOT NULL,
    idCountry  INTEGER NOT NULL,
    idImage  INTEGER,
    CONSTRAINT status_ck CHECK ((member_status = ANY (ARRAY['moderator'::text,
'suspended'::text, 'banned'::text, 'normal'::text, 'terminated'::text]))),
    CONSTRAINT age_ck CHECK (age>=18)
);


--4
CREATE TABLE requested_termination (
    id SERIAL NOT NULL PRIMARY KEY,
    dateRequested  TIMESTAMP WITH TIME zone DEFAULT now() NOT NULL,
    idMember INTEGER NOT NULL
);

--5

CREATE TABLE auction (
    id SERIAL NOT NULL PRIMARY KEY,
    author  text NOT NULL,
    description  text NOT NULL,
    duration interval NOT NULL,
    ISBN  text NOT NULL,
    title  text NOT NULL,
    dateCreated TIMESTAMP WITH TIME zone DEFAULT now() NOT NULL,
    auction_status text NOT NULL DEFAULT 'waitingApproval'::text,
    dateApproved TIMESTAMP WITH TIME zone NOT NULL,
    dateRemoved TIMESTAMP WITH TIME zone NOT NULL,
    idPublisher INTEGER,
    idLanguage INTEGER NOT NULL,
    idSeller INTEGER NOT NULL,
    CONSTRAINT auction_status_ck CHECK ((auction_status = ANY
(ARRAY['approved'::text, 'removed'::text, 'banned'::text,
'waitingApproval'::text]))),
    CONSTRAINT duration_ck CHECK (duration > '00:05:00'::interval)
);

--6
CREATE TABLE category (
    id SERIAL NOT NULL PRIMARY KEY,
    categoryName text NOT NULL UNIQUE
```

```
);

--7
CREATE TABLE publisher (
    id SERIAL NOT NULL PRIMARY KEY,
    publisherName  text NOT NULL UNIQUE
);

--8

CREATE TABLE language (
    id SERIAL NOT NULL PRIMARY KEY,
    languageName text NOT NULL UNIQUE
);

--9

CREATE TABLE category_auction (
    idCategory INTEGER NOT NULL,
    idAuction INTEGER NOT NULL,
    CONSTRAINT categoryAuction_pk PRIMARY KEY (idCategory, idAuction)
);

--10
CREATE TABLE whishlist (
    idBuyer INTEGER NOT NULL,
    idAuction INTEGER NOT NULL,
    CONSTRAINT wishList_pk PRIMARY KEY (idBuyer, idAuction)
);

--11
CREATE TABLE bid (
    idBuyer INTEGER NOT NULL,
    idAuction INTEGER NOT NULL,
    bidDate TIMESTAMP WITH TIME zone DEFAULT now() NOT NULL,
    bidValue REAL,
    CONSTRAINT bidValue_ck CHECK (bidValue > 0.0),
    CONSTRAINT bid_pk PRIMARY KEY (idBuyer, idAuction)
);

--12
CREATE TABLE auction_modification (
    id SERIAL NOT NULL PRIMARY KEY,
    dateRequested TIMESTAMP WITH TIME zone DEFAULT now() NOT NULL,
    newDescription text,
    is_approved boolean,
    dateApproved TIMESTAMP WITH TIME zone,
    idApprovedAuction INTEGER NOT NULL
);


--13
```

```
CREATE TABLE notification (
    id SERIAL NOT NULL PRIMARY KEY,
    dateSent TIMESTAMP WITH TIME zone DEFAULT now() NOT NULL,
    information text,
    is_seen boolean DEFAULT FALSE,
    dateSeen TIMESTAMP WITH TIME zone,
    idMember INTEGER NOT NULL
);


--14

CREATE TABLE notification_auction (
    idAuction INTEGER NOT NULL,
    idNotification INTEGER NOT NULL,
    CONSTRAINT notificationAuction_pk PRIMARY KEY (idAuction, idNotification)
);


--15
CREATE TABLE image (
    id SERIAL NOT NULL PRIMARY KEY,
    source text NOT NULL UNIQUE,
    idAuction  INTEGER,
    idAuctionModification  INTEGER
);

--16
CREATE TABLE message (
    id SERIAL NOT NULL PRIMARY KEY,
    dateSent TIMESTAMP WITH TIME zone DEFAULT now() NOT NULL,
    message_text text NOT NULL,
    idSender INTEGER NOT NULL,
    idReceiver INTEGER NOT NULL
);

--17
CREATE TABLE comment (
    id SERIAL NOT NULL PRIMARY KEY,
    datePosted TIMESTAMP WITH TIME zone DEFAULT now() NOT NULL,
    liked boolean,
    comment_text text NOT NULL,
    is_removed boolean DEFAULT FALSE,
    idParent INTEGER,
    idSender INTEGER NOT NULL,
    idReceiver INTEGER NOT NULL
);


-----------------------------------------------------
 --FOREIGN KEYS
-----------------------------------------------------
```

```
--3

ALTER TABLE ONLY member
    ADD CONSTRAINT member_id_country_fk FOREIGN KEY (idCountry) REFERENCES
country(id) ON UPDATE CASCADE;

ALTER TABLE ONLY member
    ADD CONSTRAINT member_id_image_fk FOREIGN KEY (idImage) REFERENCES
image(id) ON UPDATE CASCADE;

--4

ALTER TABLE ONLY requested_termination
    ADD CONSTRAINT requestedTermination_id_member_fk FOREIGN KEY (idMember)
REFERENCES member(id) ON UPDATE CASCADE;

--5
ALTER TABLE ONLY auction
    ADD CONSTRAINT auction_id_publisher_fk FOREIGN KEY (idPublisher)
REFERENCES publisher(id) ON UPDATE CASCADE;

ALTER TABLE ONLY auction
    ADD CONSTRAINT auction_id_language_fk FOREIGN KEY (idLanguage) REFERENCES
language(id) ON UPDATE CASCADE;

ALTER TABLE ONLY auction
    ADD CONSTRAINT auction_id_seller_fk FOREIGN KEY (idSeller) REFERENCES
member(id) ON UPDATE CASCADE;


--9
ALTER TABLE ONLY category_auction
    ADD CONSTRAINT categoryAuction_category_fk FOREIGN KEY (idCategory)
REFERENCES "category"(id) ON UPDATE CASCADE;

ALTER TABLE ONLY category_auction
    ADD CONSTRAINT categoryAuction_auction_fk FOREIGN KEY (idAuction)
REFERENCES auction(id) ON UPDATE CASCADE;

--10

ALTER TABLE ONLY whishlist
    ADD CONSTRAINT wishList_buyer_fk FOREIGN KEY (idBuyer) REFERENCES
member(id) ON UPDATE CASCADE;

ALTER TABLE ONLY whishlist
    ADD CONSTRAINT wishList_auction_fk FOREIGN KEY (idAuction) REFERENCES
auction(id) ON UPDATE CASCADE;

--11
ALTER TABLE ONLY bid
```

```
    ADD CONSTRAINT bid_buyer_fk FOREIGN KEY (idBuyer) REFERENCES member(id) ON
UPDATE CASCADE;

ALTER TABLE ONLY bid
    ADD CONSTRAINT bid_auction_fk FOREIGN KEY (idAuction) REFERENCES
auction(id) ON UPDATE CASCADE;

--12

ALTER TABLE ONLY auction_modification
    ADD CONSTRAINT auctionModification_id_member_fk FOREIGN KEY
(idApprovedAuction) REFERENCES auction(id) ON UPDATE CASCADE;

--13
ALTER TABLE ONLY notification
    ADD CONSTRAINT notification_id_member_fk FOREIGN KEY (idMember) REFERENCES
member(id) ON UPDATE CASCADE;

--14

ALTER TABLE ONLY notification_auction
    ADD CONSTRAINT notificationAuction_notification_fk FOREIGN KEY
(idNotification) REFERENCES notification(id) ON UPDATE CASCADE;

ALTER TABLE ONLY notification_auction
    ADD CONSTRAINT notificationAuction_auction_fk FOREIGN KEY (idAuction)
REFERENCES auction(id) ON UPDATE CASCADE;

--15
ALTER TABLE ONLY image
    ADD CONSTRAINT image_auctionModification_fk FOREIGN KEY
(idAuctionModification) REFERENCES auction_modification(id) ON UPDATE CASCADE;

ALTER TABLE ONLY image
    ADD CONSTRAINT image_auction_fk FOREIGN KEY (idAuction) REFERENCES
auction(id) ON UPDATE CASCADE;

--16

ALTER TABLE ONLY message
    ADD CONSTRAINT message_sender_fk FOREIGN KEY (idSender) REFERENCES
member(id) ON UPDATE CASCADE;

ALTER TABLE ONLY message
    ADD CONSTRAINT message_receiver_fk FOREIGN KEY (idReceiver) REFERENCES
member(id) ON UPDATE CASCADE;

--17

ALTER TABLE ONLY comment
    ADD CONSTRAINT comment_parent_fk FOREIGN KEY (idParent) REFERENCES
comment(id) ON UPDATE CASCADE;
```

```
ALTER TABLE ONLY comment
    ADD CONSTRAINT comment_sender_fk FOREIGN KEY (idSender) REFERENCES
member(id) ON UPDATE CASCADE;

ALTER TABLE ONLY comment
    ADD CONSTRAINT comment_receiver_fk FOREIGN KEY (idReceiver) REFERENCES
member(id) ON UPDATE CASCADE;
```

Histórico de revisões

Mudanças feitas à primeira submissão:

1. Correção de um *bug* nas dependências funcionais e validação do esquema, onde atribuímos como possíveis chaves um conjunto de chaves embora não ser este o intuito.
2. Alterações ao schema.sql para este estar de acordo com as convenções de código *sql*.

GROUP1726, 27/3/2018

> Daniel Vieira Azevedo, up201000307@fe.up.pt

> Nelson André Garrido da Costa, up201403128@fe.up.pt

> Rúben José da Silva Torres, up201405612@fe.up.pt

> Tiago Lascasas dos Santos, up201503616@fe.up.pt