Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

Постреляционные базы данных

Домашнее задание № 1

Выполнил:
студент группы ИУ5 – 23М Крутов Т.Ю.

Преподаватель: Виноградова М.В.

2020г.

Целью работы является освоение технологии построения моделей базы данных: ER, объектной, объектно-реляционной и полуструктурированных данных. Сравнение возможностей этих моделей с возможностями реляционной модели.

Предметная область работы - система оцифровки текстов и изображений

Сущности системы: Определим в системе пять основных сущностей – пользователь, аккаунт, заказ, текст и изображение.

Атрибуты:

Пользователь (ID пользователя, ФИО пользователя, Телефон, Пол, Дата рождения, Паспорт)

Аккаунт (ID аккаунта, Логин, Пароль, Дата регистрации)

Аккаунт клиента (ID аккаунта, Логин, Пароль, Дата регистрации, Скидка)

Аккаунт сотрудника (ID аккаунта, Логин, Пароль, Дата регистрации, Права доступа)

Заказ (ID заказа, Дата, ФИО, Телефон, Стоимость)

Текст (ID текста, Название, Содержание)

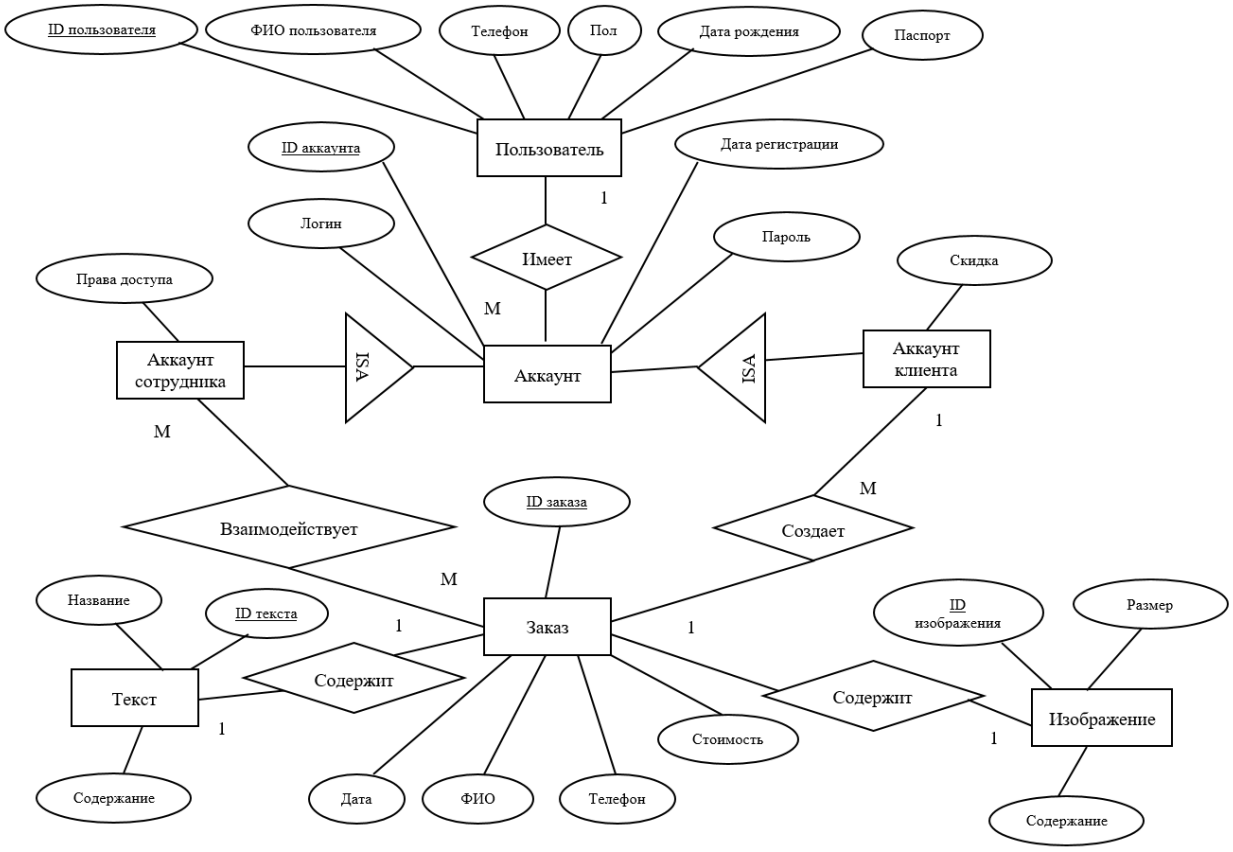
Изображение (ID изображения, Размер, Содержание)

Связи:

- Для сущностей «Пользователь» и «Аккаунт» определим бинарную связь один-ко-многим «Имеет», т.о. «Пользователь» может иметь несколько аккаунтов, но каждый «Аккаунт» имеет лишь одного пользователя.
- Введём связь типа ISA для сущностей «Аккаунт» - «Аккаунт клиента»
- Введём связь типа ISA для сущностей «Аккаунт» - «Аккаунт сотрудника»
- Связь многие-ко-многим «Взаимодействует». Один заказ могут курировать несколько сотрудников и один сотрудник может работать с несколькими заказами.

- Связь один-ко многим «Создает» между сущностями «Аккаунт клиента» и «Заказ».
- Связь один-к-одному «Содержит» между сущностями «Заказ» и «Текст»
- Связь один-к-одному «Содержит» между сущностями «Заказ» и «Текст»

ER модель проектируемой системы



Объектная модель базы данных на языке ODL (Object Definition Language)

```
Class User(extent Users key(userID))
{
    attribute integer userID;
    attribute Struct FIO
    {
        string firstname,
        string lastname,
        string middlenme
    } fio;
```

```
    attribute string phone;
    attribute string sex;
    attribute date birthDate;
    attribute string pass;
    relationship Set <Account> acnts
        inverse Account::usr;
    integer UsrCount() raises (noUsers);
    void UsrInPhone( in string phone, out Set<Users>)
        raises(noUsers, badPhone);
};

Class Account (extent Accounts key(accountID))
{
    attribute integer accountID;
    attribute string login;
    attribute string password;
    attribute date registrDate;
    relationship User usr
        inverse User::acnts;
    integer AccCount() raises (noAccounts)
    void AccInLogin(in string login, out Set <Accounts>)
        raises(noAccounts, badLogin)

}
Class Cln_account extends Account(extent Cln_accounts)
{
    attribute string accessRules;
    relationship Set <Order> ords1;
        inverse Order::cacnts;
}

Class Wrk_account()
{
    attribute integer sale;
    relationship Set <Order>ords2;
        inverse Order::wacnts;
}

Class Oder (extent Orders key(orderID))
{
    attribute integer orderID;
    attribute date orderDate;
    Struct FIO
```

```
    {
        string firstname,
        string lastname,
        string middlenme
    } fio;
    attribute Set <string> phones;
    attribute cost;
    relationship Cln_account cacnts
        inverse Account::ords1;
    relationship Set <Wrk_account> wacnts
        inverse Account::ords2;
    relationship Set<Text_>txts
        inverse Text_::ords;
    relationship Set<Image> imgs
        inverse Image::ords;

}

Class Text_(extent Texts key (textID))
{
    attribute integer textID;
    attribute string name;
    attribute string body;
    relationship Set<Order> ords
        inverse Order::txts;
    integer TxtCount() raises (noTexts);
    void TextInName(in string name, out Set <Texts>)
        raises(noTexts, badName)
}
Class Image (extent Images key(imageID))
{
    attribute integer imageID;
    attribute integer size;
    attribute string body;
    relationship Set <Order>ords
        inverse Order::imgs;
    integer ImgCount() raises(noImages)
    void ImageInSize(in integer size, out Set <Images>)
        raises(noImages, badSize)
}
}
```

Реляционная модель.

Схемы отношений:

1. Объектный подход
    - Пользователь (ID пользователя, ФИО пользователя, Телефон, Пол, Дата рождения, Паспорт)
    - Аккаунт (ID аккаунта, Логин, Пароль, Дата регистрации)
    - Аккаунт клиента (ID аккаунта, Логин, Пароль, Дата регистрации, Скидка)
    - Аккаунт сотрудника (ID аккаунта, Логин, Пароль, Дата регистрации, Права доступа)
    - Заказ (ID заказа, Дата, ФИО, Телефон, Стоимость)
    - Текст (ID текста, Название, Содержание)
    - Изображение (ID изображения, Размер, Содержание)
2. Сущностный подход
    - Пользователь (ID пользователя, ФИО пользователя, Телефон, Пол, Дата рождения, Паспорт)
    - Аккаунт (ID аккаунта, Логин, Пароль, Дата регистрации)
    - Аккаунт клиента (ID аккаунта, Скидка)
    - Аккаунт сотрудника (ID аккаунта, Права доступа)
    - Заказ (ID заказа, Дата, ФИО, Телефон, Стоимость)
    - Текст (ID текста, Название, Содержание)
    - Изображение (ID изображения, Размер, Содержание)
3. Подход пустых значений
    - Пользователь (ID пользователя, ФИО пользователя, Телефон, Пол, Дата рождения, Паспорт)
    - Аккаунт (ID аккаунта, Логин, Пароль, Дата регистрации, Права доступа, скидка)
    - Заказ (ID заказа, Дата, ФИО, Телефон, Стоимость)
    - Текст (ID текста, Название, Содержание)

- Изображение (ID изображения, Размер, Содержание)

```
CREATE TABLE User
(
    userID INTEGER PRIMARY KEY UNIQUE,
    fio VARCHAR(120) NOT NULL,
    phone VARCHAR (15) NOT NULL,
    sex VARCHAR(3),
    birthday DATE DEFAULT '01.01.2000',
    passport NOT NULL

);

CREATE TABLE Account
(
    accountID INTEGER PRIMARY KEY NOT NULL UNIQUE,
    login VARCHAR(30) NOT NULL UNIQUE,
    password VARCHAR(30) CHECK (password != '') ,
    registrDate DATE NOT NULL,
    userID INTEGER,
    FOREIGN KEY(userID) REFERENCES User(userID) ON UPDATE CASCADE ON DELETE S
ET NULL
);

CREATE TABLE Cln_account
(
    clnID INTEGER NOT NULL REFERENCES Account(accountID) ON UPDATE CASCADE ON
 DELETE CASCADE,
    PRIMARY KEY(clnID),
    sale INTEGER DEFAULT 0

;)

CREATE TABLE Wrk_account
(
    wrkID INTEGER NOT NULL REFERENCES Account(accountID) ON UPDATE CASCADE ON
 DELETE CASCADE,
    accessRules INTEGER DEFAULT 0,
    PRIMARY KEY wrkID
;)

CREATE TABLE AccOrder
(
```

```
    wrkID  INTEGER NOT NULL REFERENCES Wrk_account(wrkID) ON UPDATE CASCADE O
N DELETE CASCADE,
    orderID INTEGER NOT NULL REFERENCES Order(orderID) ON UPDATE CASCADE ON D
ELETE CASCADE,
    PRIMARY KEY (wrkID, orderID)
)

CREATE TABLE Order
(
    orderID INTEGER PRIMARY KEY NOT NULL UNIQUE,
    orderDate DATE NOT NULL,
    fio VARCHAR(120) NOT NULL,
    phone VARCHAR (15) NOT NULL,
    cost FLOAT,
    accountID INTEGER,
    FOREIGN KEY(accountID) REFERENCES Account (accountID) ON UPDATE CASCADE O
N DELETE SET NULL
);

CREATE TABLE Text_
(
    textID INTEGER PRIMARY KEY UNIQUE,
    name VARCHAR(300) NOT NULL,
    body VARCHAR NOT NULL,
    orderID INTEGER,
    FOREIGN KEY(orderID) REFERENCES Order (orderID) ON UPDATE CASCADE ON DELE
TE SET NULL

);

CREATE TABLE Image
(
    imageID INTEGER PRIMARY KEY UNIQUE,
    size INTEGER NOT NULL,
    body VARCHAR NOT NULL,
    orderID INTEGER,
    FOREIGN KEY(orderID) REFERENCES Order(orderID) ON UPDATE CASCADE ON DELET
E SET NULL
);
```

Объектный подход

```
CREATE TABLE Account
(
```

```
    accountID INTEGER PRIMARY KEY NOT NULL UNIQUE,
    login VARCHAR(30) NOT NULL UNIQUE,
    password VARCHAR(30) CHECK (password != '') ,
    registrDate DATE NOT NULL,
    userID INTEGER,
    FOREIGN KEY(userID) REFERENCES User(userID) ON UPDATE CASCADE ON DELETE S
ET NULL
);

CREATE TABLE Cln_account
(
    clnID INTEGER NOT NULL REFERENCES Account(accountID) ON UPDATE CASCADE ON
 DELETE CASCADE,
    PRIMARY KEY(clnID),
    sale INTEGER DEFAULT 0,
    login VARCHAR(30) NOT NULL UNIQUE,
    password VARCHAR(30) CHECK (password != '') ,
    registrDate DATE NOT NULL,
    userID INTEGER,
FOREIGN KEY(userID) REFERENCES User(userID) ON UPDATE CASCADE ON DELETE SET N
ULL

;)

CREATE TABLE Wrk_account
(
    wrkID INTEGER NOT NULL REFERENCES Account(accountID) ON UPDATE CASCADE ON
 DELETE CASCADE,
    accessRules INTEGER DEFAULT 0,
    PRIMARY KEY wrkID,
    login VARCHAR(30) NOT NULL UNIQUE,
    password VARCHAR(30) CHECK (password != '') ,
    registrDate DATE NOT NULL,
    userID INTEGER,
FOREIGN KEY(userID) REFERENCES User(userID) ON UPDATE CASCADE ON DELETE SET N
ULL

;)
```

Подход пустых значений для таблицы Account :

```
CREATE TABLE Account
```

```
(
    accountID INTEGER PRIMARY KEY NOT NULL UNIQUE,
    login VARCHAR(30) NOT NULL UNIQUE,
    password VARCHAR(30) CHECK (password != '') ,
    registrDate DATE NOT NULL,
    accessRules INTEGER,
    sale INTEGER,
    userID INTEGER,
    FOREIGN KEY(userID) REFERENCES User(userID) ON UPDATE CASCADE ON DELETE S
ET NULL
);
```

Объектно-реляционная модель.

- Пользователь (ID пользователя, ФИО пользователя (Фамилия, Имя, Отчество), Телефон, Пол, Дата рождения, Паспорт)
- Аккаунт (ID аккаунта, Логин, Пароль, Дата регистрации, Пользователь (* Пользователь))
- Аккаунт клиента (ID аккаунта, Логин, Пароль, Дата регистрации, Пользователь (*Пользователь), Скидка)
- Аккаунт сотрудника (ID аккаунта, Логин, Пароль, Дата регистрации, Пользователь (*Пользователь), Права доступа)
- Заказ (ID заказа, Дата, ФИО, Телефон, Стоимость, Аккаунт(*Аккаунт))
- Текст (ID текста, Название, Содержание, Заказ(*Заказ))

Создание типизированных таблиц

```
CREATE TYPE UserT AS
(
    userID INTEGER,
    fio FIO,
    phone VARCHAR(15),
    sex VARCHAR(3),
    birthday DATE,
    passport VARCHAR(16)
);
CREATE TYPE AccountT AS
(
    accountID INTEGER,
```

```
    login varchar(40),
    password varchar (40),
    registrDate date,
    usr REF (UserT) SCOPE USR
);
CREATE TYPE Cln_accountT UNDER AccountT AS
(
    sale INTEGER
);
CREATE TYPE Wrk_accountT UNDER AccountT AS
(
    accsessRules INTEGER
);
CREATE TYPE  OderT AS
(
    orderID,
    orderDate date,
    fio FIO,
    phone ROW(tel1 varchar(15), tel2 VARCHAR(15)),
    cost float,
    clnac REF (Cln_accountT) SCOPE CLNACC
);

CREATE TYPE ImageT AS
(
    name varchar(300),
    size integer
    ord REF (OderT) SCOPE ORD
);
```

Создание таблиц:

```
CREATE TABLE USR OF UserT
(
    REF IS IDu system generated,
    PRIMARY KEY(userID)
);

CREATE TABLE ACCNT OF AccountT
(
    REF IS IDa system generated,
    CHECK (password != ''),
    PRIMARY KEY (accountID)
);
```

```
CREATE TABLE CLNACC OF Cln_accountT UNDER ACCNT ();

CREATE TABLE WRKACC OF Wrk_accountT UNDER ACCNT ();

CREATE TABLE WO
(
    wrk REF (Wrk_accountT) SCOPE WRKACC,
    ord REF (OderT) SCOPE ORD
)

CREATE TABLE ORD OF OderT
(
    REF IS IDo system generated,
    PRIMARY KEY (orderID)
)

CREATE TABLE IMG OF ImageT
(
    REF IS IDi system generated,
    PRIMARY KEY (imageID)
)
```

Методы:

```
CREATE METHOD fullname() RETURN VARCHAR(120)
FOR FIO
BEGIN
    RETURN SELF.firstname || SELF.lastname || SELF.middlenme

CREATE TYPE FIO AS
(
    firstname VARCHAR(40),
    lastname VARCHAR (40),
    middlenme VARCHAR (40)
);
METHOD fullname() RETURN VARCHAR (120);
```

Сравнение пользовательских типов данных:

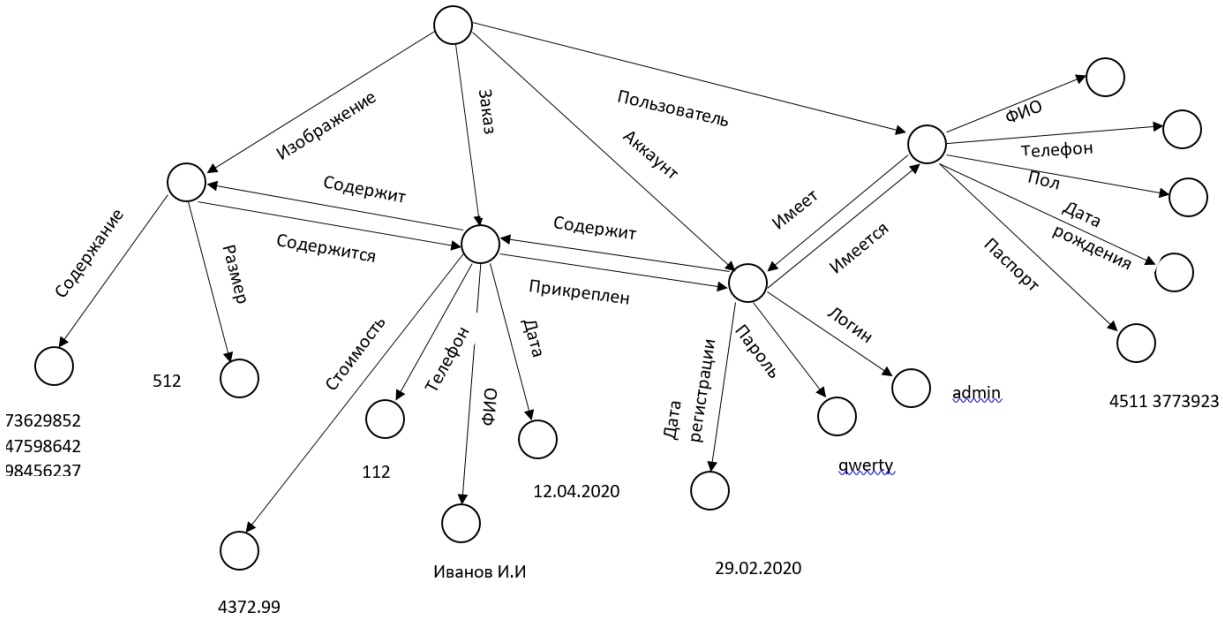
```
CREATE ORDERING FOR UserT ORDER FULL EQUALS ONLY
CREATE ORDERING FOR OderT BY STATE

CREATE ORDERING AccountT ORDER BY STATE BY RELATIVE WITH Fun
```

```
CREATE FUCNCTION Fun (IN ACC1  AccountT, IN ACC2 AccountT) RETURNS
integer
IF ACC1.login())< ACC2.login() THEN RETURN(-1)
ELSEIF ACC1.login() > ACC2.login() THEN RETURN(-1)
ELSE RETURN (0)
ENDIF;
```

Модель полуструктурированных данных

Граф полуструктурированных даных



XML-описание

```
<?xml version = "1.0" encoding="Windows-1251" standalone = "no"?>
<!DOCTYPE db SYSTEM "dtd.dtd">
<db>
    <user idu = "u1" toacc ="acc1 " >
        <fio>Иванов И.И. </fio>
        <phone> 112</phone>
        <sex>m</sex>
        <date>11.05.1998</date>
        <passport>4511 3829322</passport>
```

```
    </user>
    <account ida = "acc1" tou ="u1" too="o1">
        <login>admin</login>
        <password>qwerty</password>
        <date> 29.02.2020</date>
    </account>
    <order ido = "o1" toa = "acc1" toi = "img1">
        <date>12.04.2020</date>
        <fio>Иванов И.И.</fio>
        <phone>112</phone>
        <cost>4372.99</cost>
    </order>
    <image idi = "img1" too = "o1">
        <size>512</size>
        <body>93219805214751002851517098142319472193 7</body>
    </image>
</db>
```

DTD – определение

```
<!ELEMENT db (user+, account+, order*, image*) >
<!ELEMENT user(fio,phone,sex,date,passport)>
<!ELEMENT fio (#PCDATA)>
<!ELEMENT phone(#PCDATA)>
<!ELEMENT sex(#PCDATA)>
<!ELEMENT date(#PCDATA)>
<!ELEMENT passport(#PCDATA)>
<!ELEMENT account(login, password, date)>
<!ELEMENT login (#PCDATA)>
<!ELEMENT password (#PCDATA)>
<!ELEMENT date (#PCDATA)>
<!ELEMENT order(date,fio,phone,cost)>
<!ELEMENT date (#PCDATA)>
<!ELEMENT fio (#PCDATA)>
<!ELEMENT phone (#PCDATA)>
<!ELEMENT cost (#PCDATA)>
<!ELEMENT image(size, body)>
<!ELEMENT size (#PCDATA)>
<!ELEMENT body (#PCDATA)>
<!ATTLIST user idu ID toacc IDREFS >
<!ATTLIST account ida ID tou IDREF too IDREFS>
<!ATTLIST order ido ID #REQUIRED too IDREF toi IDREF>
<!ATTLIST image idm ID #REQUIRED too IDREF>
```

Список используемой литературы:

1. Постреляционные модели данных и языки запросов / Виноградов В.И., Виноградова М.В. -М. : Изд-во МГТУ им. Н. Э. Баумана, 2017. -[100] с. -Режим доступа: http://ebooks.bmstu.ru/catalog/254/book1615.html (дата обращения: 08.12.2017). -ISBN 978-5-7038-4283-6.