Приложение Д. Скрипт базы данных

АННОТАЦИЯ

В данном программном документа приведен скрипт базы данных для мобильного приложения социальной сети «DelWiz».

В данном программном документе, в разделе «Скрипт базы данных» указано наименование программы для которой разрабатывалась база данных, область применения скрипта, а также сам скрипт базы данных.

СОДЕРЖАНИЕ

1. СКРИПТ БАЗЫ ДАННЫХ

1.1. Наименование скрипта

Наименование – «social_network.sql»

1.2. Область применения объекта

Скрипт предназначен для мобильного приложения для социальной сети «AIsoc».

1.3. Скрипт

```
create database SocialNetwork
go

use SocialNetwork
go

CREATE TABLE Tokens (
    token_id INT IDENTITY(1,1) PRIMARY KEY,
    token VARCHAR(200) NOT NULL,
    token_datetime DATETIME2 NOT NULL DEFAULT(SYSDATETIME())
);

create table [dbo].Role(
   ID_Role int not null identity(1,1),
   Name_Role varchar(50) not null,
   constraint [PK_Role] primary key clustered
   ([ID_Role] ASC) on [PRIMARY],
   constraint [UQ_Name_Role] unique ([Name_Role])
)

insert into [dbo].Role (Name_Role)
values('Администратор БД'),
('Подписчик'),
('Блогер'),
('Модератор'),
('Специалист службы поддержки')
go

select * from Role

create table [dbo].Message_User(
   ID_Message int not null identity(1,1),
   Text_Message varchar(max) not null,
   Sending_time datetime not null default getdate(),
   [User_ID] [int]  not null,
   [Sender_ID] [int]  not null,
   constraint [FK_User_Message_User] foreign key ([User_ID])
   references [dbo].[Users] ([ID_User]),
   constraint [FK_Sender_Message_User] foreign key ([Sender_ID])
   references [dbo].[Users] ([ID_User]),
   constraint [PK_Message] primary key clustered
   ([ID_Message] ASC) on [PRIMARY],
)

select * from Message_User

create table [dbo].Picture(
   ID_Picture int not null identity(1,1),
   PhotoData varbinary(max) not null, -- Бинарные данные изображения
    UploadDate datetime not null, -- Дата загрузки изображения
   constraint [PK_Picture] primary key clustered
   ([ID_Picture] ASC) on [PRIMARY]
)



select * from Picture

create table [dbo].Progress(
   ID_Progress int not null identity(1,1),
   Name_Progress  NVARCHAR(50)  NOT  NULL  CONSTRAINT  CHK_Name_Progress  CHECK
(Name_Progress COLLATE Latin1_General_BIN LIKE '%[a-zA-Z]%'),
    Description_Progress      NVARCHAR(100)      NOT      NULL      CONSTRAINT
CHK_Description_Progress CHECK (Description_Progress COLLATE Latin1_General_BIN
LIKE '%[a-zA-Z]%'),
   ID_Picture int not null,
   constraint [PK_Progress] primary key clustered
   ([ID_Progress] ASC) on [PRIMARY],
   constraint [FK_Picture_Progress] foreign key (ID_Picture)
   references [dbo].[Picture] ([ID_Picture])
)


CREATE TABLE [dbo].Users (
    ID_User int not null identity(1,1),
   First_name NVARCHAR(50) NOT NULL CONSTRAINT CHK_First_name CHECK (First_name
COLLATE Cyrillic_General_CI_AS LIKE N'%[А-Яа-я]%'),
    Last_name NVARCHAR(50) NOT NULL CONSTRAINT CHK_Last_name CHECK (Last_name
COLLATE Cyrillic_General_CI_AS LIKE N'%[А-Яа-я]%'),
    Middle_name NVARCHAR(50) NULL CONSTRAINT CHK_Middle_name CHECK (Middle_name
COLLATE Cyrillic_General_CI_AS LIKE N'%[А-Яа-я]%'),
    Login_User VARCHAR(50) NOT NULL CONSTRAINT CHK_Login_User CHECK (Login_User
LIKE '[a-zA-Z]{8,}'),
    Password_User  VARCHAR(100)  NOT  NULL  CONSTRAINT  CHK_Password_User  CHECK
(Password_User LIKE '[a-zA-Z]{8,}'),
    Salt varchar(256) null,
    Photo_ID int null,
    Number_of_messages int null,
    Time_in_the_app time null,
    Role_ID int not null,
    constraint PK_User primary key clustered (ID_User ASC) on [PRIMARY],
    constraint  FK_Role_Users  foreign  key  (Role_ID)  references  [dbo].[Role]
(ID_Role) ON DELETE CASCADE
);

insert into Users(First_name, Last_name, Middle_name, Login_User, Password_User,
Number_of_messages, Role_ID)
values ('Фамилия','Имя','Отчество','Логин','Пароль',1,1)

select * from Users

create table [dbo].Correspondence(
   [ID_Correspondence] [int] not null identity(1,1),
   [Message_ID] [int]  not null,
   [User_ID] [int]  not null,
   [Sender_ID] [int]  not null,
   constraint [PK_Correspondence] primary key clustered
   ([ID_Correspondence] ASC) on [PRIMARY],
   constraint [FK_User_Correspondence] foreign key ([User_ID])
   references [dbo].[Users] ([ID_User]),
   constraint [FK_Message_Correspondence] foreign key ([Message_ID])
   references [dbo].[Message_User] ([ID_Message]),
   constraint [FK_Sender_Correspondence] foreign key ([Sender_ID])
   references [dbo].[Users] ([ID_User])
)
go


select * from Correspondence

   create table [dbo].User_achievements(
         [ID_User_achievements] [int] not null identity(1,1),
         [Progress_ID] [int]  not null,
         [User_ID] [int]  not null
         constraint [PK_User_achievements] primary key clustered
         ([ID_User_achievements] ASC) on [PRIMARY],
         constraint [FK_User_User_achievements] foreign key ([User_ID])
         references [dbo].[Users] ([ID_User]),
         constraint [FK_Progress_User_achievements] foreign key ([Progress_ID])
         references [dbo].[Progress] ([ID_Progress])
   )
   go

CREATE TABLE [dbo].UserPuctire (
    Photo_ID int not null identity(1,1),
    UserID int not null, -- Связь с таблицей Users
    PhotoData varbinary(max) not null, -- Бинарные данные изображения
    UploadDate datetime not null, -- Дата загрузки изображения
    -- Другие метаданные, если необходимо
    constraint PK_UserPhoto primary key clustered (Photo_ID ASC) on [PRIMARY],
    constraint  FK_UserPhoto_User  foreign  key  (UserID)  references  [dbo].[Users]
(ID_User) ON DELETE CASCADE
);

select * from UserPuctire

CREATE TABLE [dbo].UserPhotos (
    ID_Photo int not null identity(1,1),
    UserID int not null,
    PhotoData varbinary(max) not null,
    UploadDate datetime not null,
    constraint PK_UserPhotos primary key clustered (ID_Photo ASC) on [PRIMARY],
    constraint FK_UserPhotos_Users foreign key (UserID) references [dbo].[Users]
(ID_User) ON DELETE CASCADE
);

select * from UserPhotos

create table [dbo].News(
   ID_News int not null identity(1,1),
   Description_News varchar(100) not null,
   Likes int null,
   DisLike int null,
   ID_User int not null,
   [Picture_ID] [int]  not null,
   Sending_time datetime not null default getdate(),
   constraint [FK_Picture_News] foreign key ([Picture_ID])
   references [dbo].[Picture] ([ID_Picture]),
   constraint [PK_News] primary key clustered
   ([ID_News] ASC) on [PRIMARY],
   constraint [FK_User_News] foreign key (ID_User)
   references [dbo].[Users] ([ID_User])
)

select * from News

create table [dbo].News_puctires(
   [ID_News_puctires] [int] not null identity(1,1),
   [Picture_ID] [int]  not null,
   [News_ID] [int]  not null
   constraint [PK_News_puctires] primary key clustered
   ([ID_News_puctires] ASC) on [PRIMARY],
   constraint [FK_News_News_puctires] foreign key ([News_ID])
   references [dbo].[News] ([ID_News]),
   constraint [FK_Picture_News_puctires] foreign key ([Picture_ID])
   references [dbo].[Picture] ([ID_Picture])
)
go

select * from News_puctires


create table [dbo].Friends(
   ID_Friends [int] not null identity(1,1),
   [User_ID] [int]  not null,
   [Friends_ID] [int]  not null
   constraint [PK_Friends] primary key clustered
   ([ID_Friends] ASC) on [PRIMARY],
   constraint [FK_User_User] foreign key ([User_ID])
   references [dbo].[Users] ([ID_User]),
   constraint [FK_Friends_User] foreign key ([Friends_ID])
   references [dbo].[Users] ([ID_User])
)

select * from [dbo].Friends

create table [dbo].Applications(
   ID_Applications [int] not null identity(1,1),
   [Recipient_ID] [int]  not null,
   [Sender_ID] [int]  not null
   constraint [PK_Applications] primary key clustered
   ([ID_Applications] ASC) on [PRIMARY],
   constraint [FK_Recipient_Applications] foreign key ([Recipient_ID])
   references [dbo].[Users] ([ID_User]),
   constraint [FK_Sender_Applications] foreign key ([Sender_ID])
   references [dbo].[Users] ([ID_User])
)

select * from Applications

create table [dbo].Supports(
   ID_Supports [int] not null identity(1,1),
   [Message_ID] int not null,
   [User_ID] [int]  not null,
   [Specialist_ID] [int]  not null
   constraint [PK_Supports] primary key clustered
   ([ID_Supports] ASC) on [PRIMARY],
   constraint [FK_Message_Supports] foreign key ([Message_ID])
   references [dbo].[Message_User] ([ID_Message]),
   constraint [FK_Supports_User] foreign key ([User_ID])
   references [dbo].[Users] ([ID_User]),
   constraint [FK_Supports_Specialist] foreign key ([Specialist_ID])
   references [dbo].[Users] ([ID_User])
```

```
)

CREATE VIEW UserRolesInfo AS
SELECT u.ID_User, u.First_name, u.Last_name, u.Login_User, r.Name_Role
FROM Users u
INNER JOIN Role r ON u.Role_ID = r.ID_Role;

CREATE VIEW NewsLikesInfo AS
SELECT n.ID_News, n.Description_News, n.Likes, n.DisLike, u.First_name + ' ' +
u.Last_name AS UserName
FROM News n
INNER JOIN Users u ON n.ID_User = u.ID_User;

CREATE VIEW UserFriends AS
SELECT u.First_name + ' ' + u.Last_name AS UserFullName, f2.First_name + ' ' +
f2.Last_name AS FriendFullName
FROM Users u
INNER JOIN Friends f ON u.ID_User = f.User_ID
INNER JOIN Users f2 ON f.Friends_ID = f2.ID_User;

CREATE FUNCTION dbo.CountUserMessages (@UserID INT)
RETURNS INT
AS
BEGIN
    DECLARE @MessageCount INT;
    SELECT @MessageCount = COUNT(ID_Message)
    FROM Message_User
    WHERE User_ID = @UserID;

    RETURN @MessageCount;
END;

CREATE FUNCTION dbo.GetNewsLikes (@NewsID INT)
RETURNS TABLE
AS
RETURN (
    SELECT Likes, DisLike
    FROM News
    WHERE ID_News = @NewsID
);

CREATE FUNCTION dbo.GetUserPhotos (@UserID INT)
RETURNS TABLE
AS
RETURN (
    SELECT ID_Photo, PhotoData, UploadDate
    FROM UserPhotos
    WHERE UserID = @UserID
);
```