

Министерство образования Республики Беларусь
Учреждения образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ
И РАДИОЭЛЕКТРОНИКИ

КАФЕДРА ИНФОРМАТИКИ

Отчет по лабораторной работе №2
По теме «Проектирование и создание БД»

Выполнил:
студент гр. 053501
Шебеко Ю.А.

Проверил:
Гриценко Н. Ю.

Минск 2023

СОДЕРЖАНИЕ

1	Цель работы.....	3
2	Создание логической и физической модели базы данных	4
2.1	DFD диаграмма	4
2.2	ERD диаграмма	5
2.3	IDEF1.X диаграмма.....	6
3	Создание базы данных.....	8
	Приложение А DDL скрипты.....	9

1 ЦЕЛЬ РАБОТЫ

Целью лабораторной работы №2 является разработать структуру базы данных, создать логическую и физическую модель, построив DFD, ERD и IDEF1.X диаграммы для визуализации и представления хранимых данных. Сравнить использованные для построения структуры диаграммы.

Создать базу данных и сгенерировать DDL скрипты. Аргументировать выбор SQL базы данных.

2 СОЗДАНИЕ ЛОГИЧЕСКОЙ И ФИЗИЧЕСКОЙ МОДЕЛИ БАЗЫ ДАННЫХ

2.1 DFD диаграмма

Диаграмма потоков данных (data flow diagram, DFD) — один из основных инструментов структурного анализа и проектирования информационных систем, существовавших до широкого распространения UML.

Исторически сложилось так, что для описания диаграмм DFD используются две нотации — Йордана (Yourdon) и Гейна-Карсон (Gane-Sarson), отличающиеся синтаксисом.

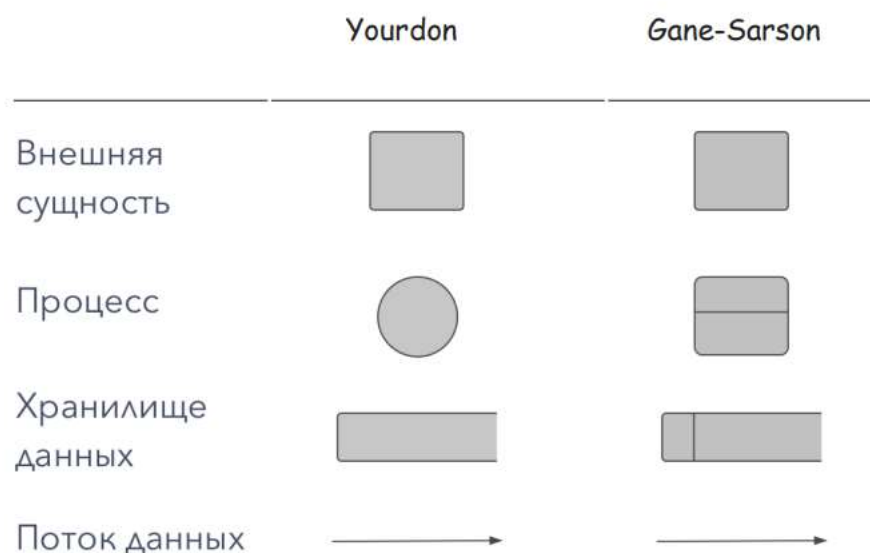


Рисунок 2.1.1 – Различие нотаций

Нотация DFD — удобное средство для формирования контекстной диаграммы, то есть диаграммы, показывающей разрабатываемую АИС в коммуникации с внешней средой. Это — диаграмма верхнего уровня в иерархии диаграмм DFD. Её назначение — ограничить рамки системы, определить, где заканчивается разрабатываемая система и начинается среда.

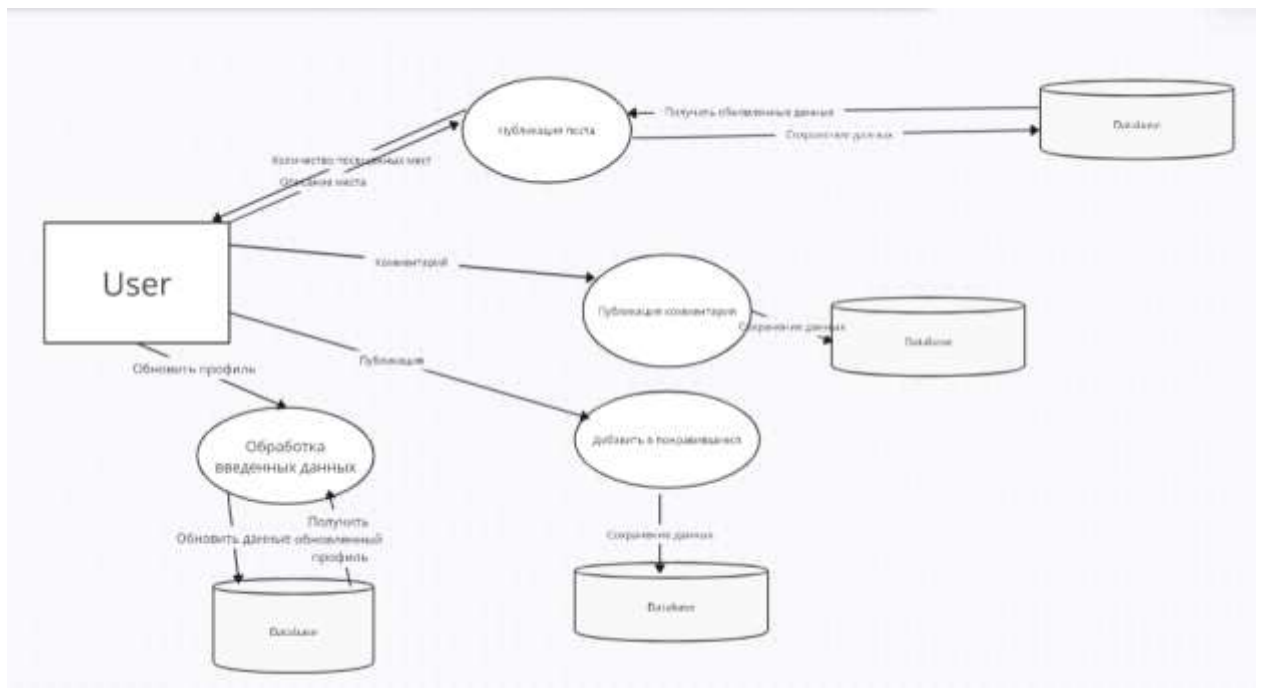


Рисунок 2.1.2 – Диаграмма потоков данных

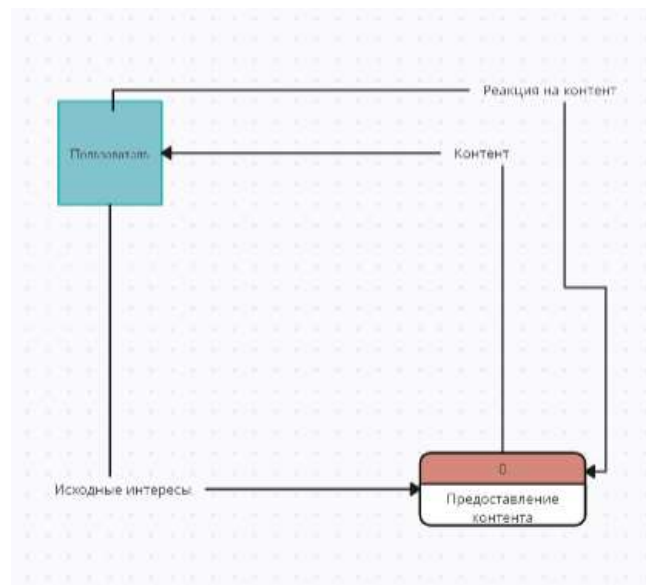


Рисунок 2.1.3 – Контекстная диаграмма

2.2 ERD диаграмма

ER-модель (*Entity-Relationship model*, модель «сущность — связь») — модель данных, позволяющая описывать концептуальные схемы предметной области.

ER-модель используется при высокоуровневом (концептуальном) проектировании баз данных. С её помощью можно

выделить ключевые сущности и обозначить связи, которые могут устанавливаться между этими сущностями.

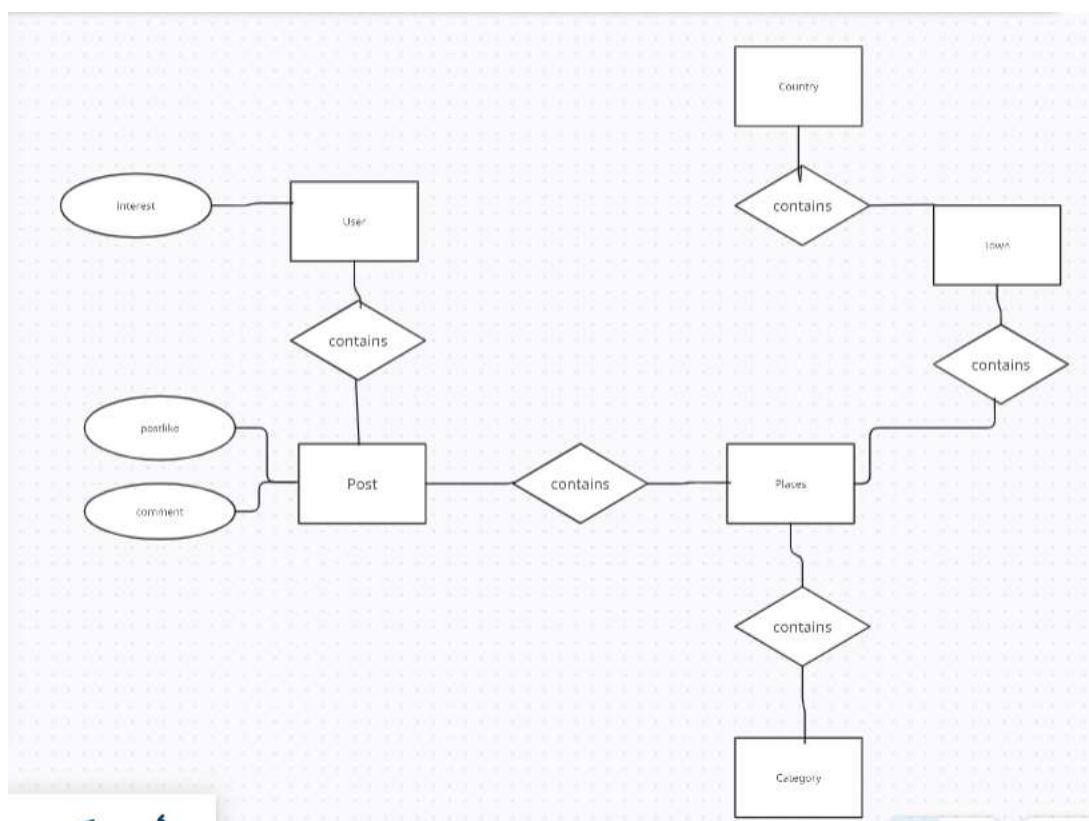


Рисунок 2.2.1 – ER диаграмма

2.3 IDEF1.X диаграмма

IDEF1X (IDEF1 Extended) — Data Modeling — методология моделирования баз данных на основе модели «сущность-связь». Применяется для построения информационной модели, которая представляет структуру информации, необходимой для поддержки функций производственной системы или среды. Метод IDEF1, разработанный Т. Рэйми (T. Ramey) на основе подходов П. Чена и позволяет построить модель данных, эквивалентную реляционной модели в третьей нормальной форме. В настоящее время на основе совершенствования методологии IDEF1 создана её новая версия — методология IDEF1X. Она разработана с учётом таких требований, как простота изучения и возможность автоматизации.

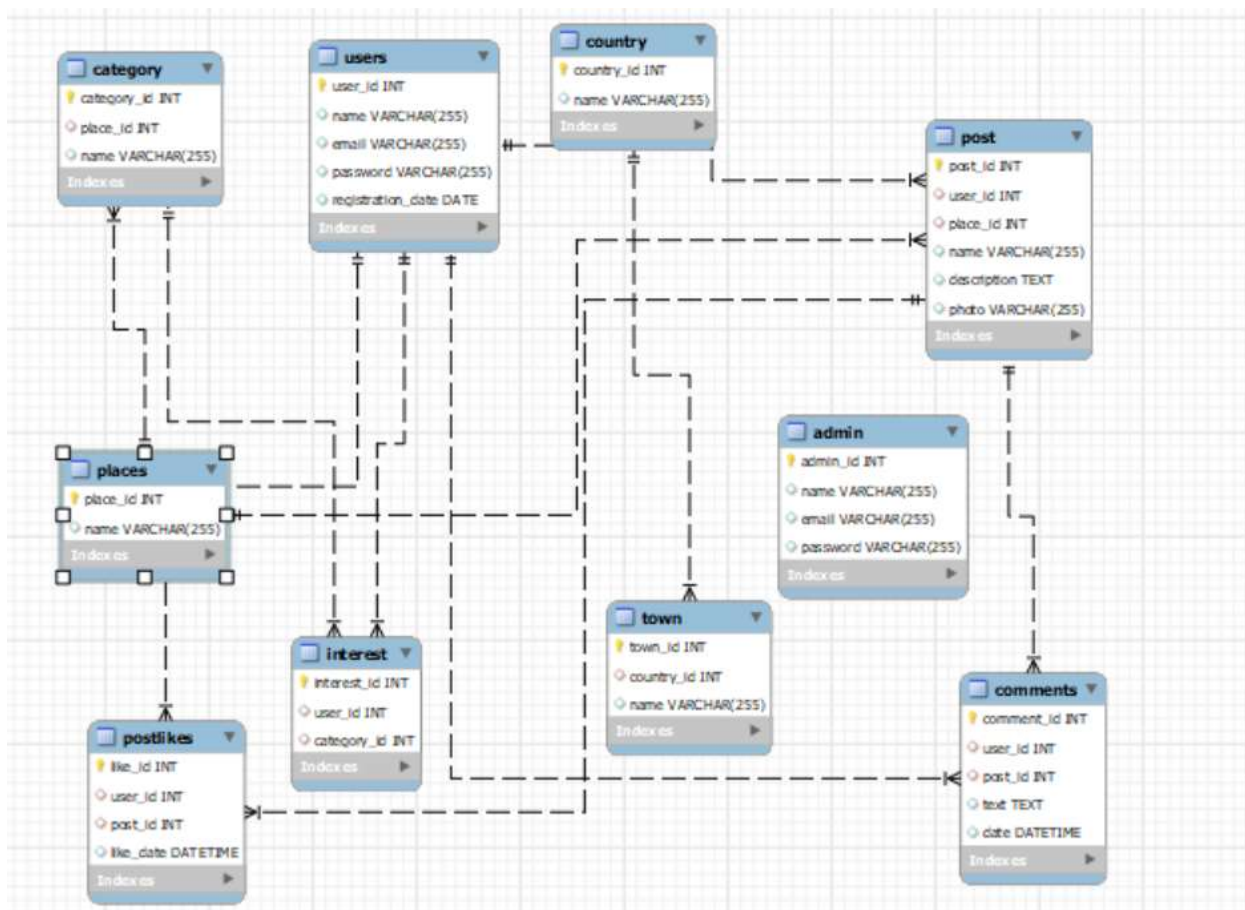


Рисунок 2.3.1 - Диаграмма IDEF1X

3 СОЗДАНИЕ БАЗЫ ДАННЫХ

Для создания базы данных использовала систему управления базами данных – MySQL. Генерацию DDL скриптов смотреть в приложении А.

SQL и NoSQL — две популярные модели баз данных, которые используют для решения различных задач.

SQL (Structured Query Language) — это язык структурированных запросов, используемый для управления и манипулирования реляционными базами данных. SQL-базы данных применяются там, где необходимо хранить и управлять данными структурированной природы, например, информацией о продуктах, покупателях и оформленных заказах в магазине.

NoSQL (Not Only SQL) — это широкий термин, который относится к нереляционным моделям баз данных, которые используют различные структуры для хранения данных: документы, ключ-значение, столбцовые и графовые БД. NoSQL-базы данных применяются, когда необходимо хранить данные неструктурированной природы, например, большие объёмы текстовых данных, изображения и видео.

Была выбрана SQL база данных для эффективного хранения и извлечения информации, благодаря структурированному формату хранения и организации данных. Также в моем приложении будут выполняться сложные операции извлечения данных (поиск мест по категориям, которые интересуют пользователя), а SQL позволяет легко выполнять такие операции.

ПРИЛОЖЕНИЕ А DDL СКРИПТЫ

```
create database travel;
```

```
Use travel;
```

```
CREATE TABLE Users (  
    user_id INT primary key,  
    name varchar(255),  
    email varchar(255),  
    password varchar(255),  
    registration_date date  
);
```

```
CREATE TABLE Admin (  
    admin_id INT primary key,  
    name varchar(255),  
    email varchar(255),  
    password varchar(255)  
);
```

```
use travel;
```

```
CREATE TABLE PostLikes (  
    like_id INT primary key,  
    user_id int,  
    post_id int,  
    like_date datetime,  
    foreign key (user_id) references Users(user_id),  
    foreign key (post_id) references Post(post_id)  
);
```

```
CREATE TABLE Country (  
    country_id INT primary key,  
    name varchar(255)  
);
```

```
CREATE TABLE Town (  
    town_id INT primary key,  
    country_id INT,  
    name varchar(255),  
    foreign key (country_id) references Country(country_id)  
);
```

```
CREATE TABLE Places (  

```

```

        place_id INT primary key,
town_id int,
name varchar(255),
foreign key (town_id) references Town(town_id)
);

```

```

CREATE TABLE Post (
    post_id INT primary key,
    user_id INT,
    place_id INT,
    name varchar(255),
    description TEXT,
    photo varchar(255),
    foreign key (user_id) references Users(user_id),
    foreign key (place_id) references Places(place_id)
);

```

```

CREATE TABLE Comments (
    comment_id INT primary key,
    user_id INT,
    post_id int,
    text text,
    date datetime,
    foreign key (user_id) references Users(user_id),
    foreign key (post_id) references Post(post_id)
);

```

```

use travel;

```

```

CREATE TABLE Category (
    category_id INT primary key,
    place_id INT,
    name varchar(255),
    foreign key (place_id) references Places(place_id)
);

```

```

CREATE TABLE Interest (
    interest_id INT primary key,
    user_id INT,
    category_id INT,
    foreign key (user_id) references Users(user_id),
    foreign key (category_id) references Category(category_id)
);

```