Лабораторная работа №3

Реализация системы обмена данными с БД.

Цель работы: реализовать систему обмена данными с БД.

Ход работы:

Цель этапа концептуального проектирования – создание концептуальной модели данных исходя из представлений пользователей о предметной области.

Это начальная стадия проектирования, на которой принимаются решения определяющие последующий облик, и проводится исследование и согласование параметров созданных технических решений с возможной их организацией. Для ее достижения выполняется ряд последовательных процедур. Для идентификации сущностей определяются объекты, которые существуют независимо от других. Такие объекты являются сущностями. Каждой сущности присваивается осмысленное имя, понятное пользователям.

Построение модели «сущность - связь» заключается в следующем:

− выделение сущностей (сильные, слабые, идентификационно-зависимые);

− определение атрибутов (идентификационные, информационные, простые, сложные, однозначные, многозначные, производные);

− определение степени связи (полная, частичная, рекурсивная, альтернативная);

− определение показателя кардинальности (максимальное или минимальное кардинальное число);

− устранение ловушек концептуального моделирования.

Определяются только те связи между сущностями, которые необходимы для удовлетворения требований к проекту базы данных. Устанавливается тип каждой из них. Выявляется класс принадлежности сущностей.

Результат концептуального проектирования представляется в виде графической схемы, на которой изображены сущности и связи между ними определенной мощности, как показано на рисунке 1.

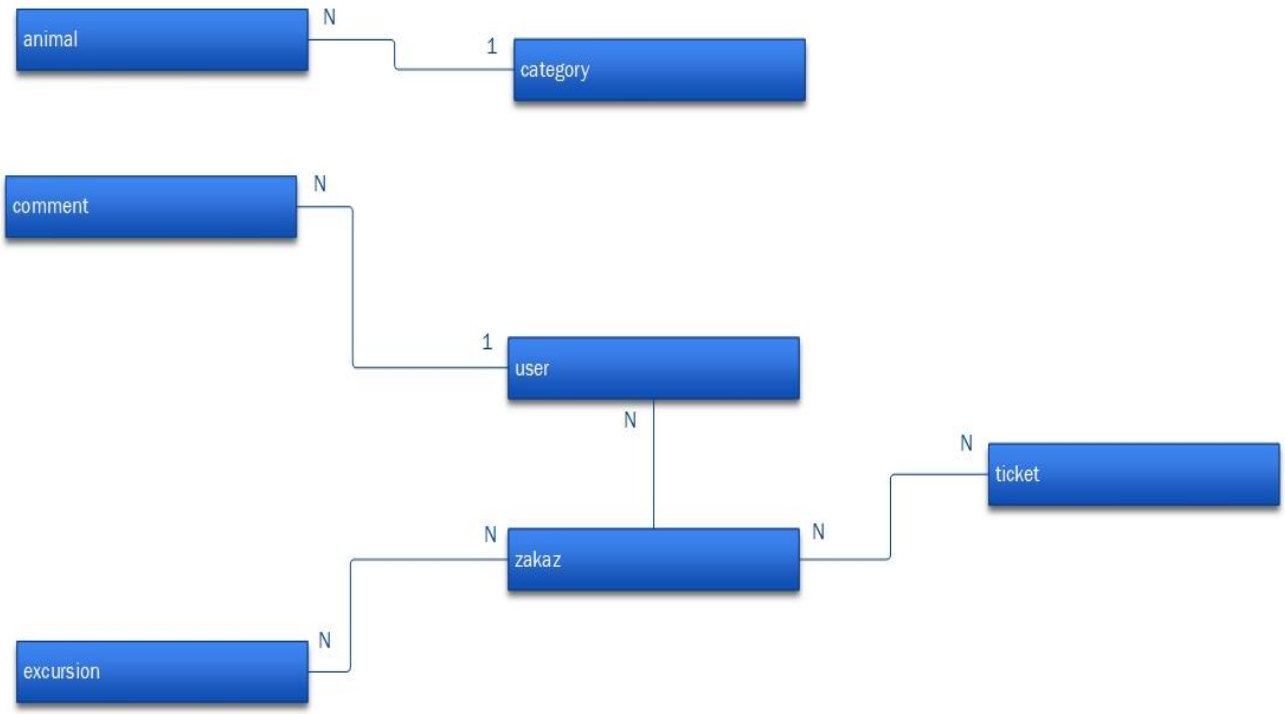

Рисунок 1 - Концептуальная модель базы данных

Сущность Animal связана с сущностью category как N: 1, так как у одного Animal моет быть только одна category, а у одной category может быть много Animal.

Сущность comment связана с сущностью как user 1:N, так как один comment принадлежит одному user, а один user может оставить много comment.

Сущность user связана с сущностью Zakaz как 1:N, так как один user может совершить один Zakaz, а у Zakaz может быть у многих user.

Сущность Zakaz связана с сущностью excursion как N: N, так как одна excursion может быть во многих Zakaz, а в одном Zakaz может быть много excursion.

Сущность Zakaz связана с сущностью ticket как N: N, так как один ticket может быть во многих Zakaz, а в одном Zakaz может быть много ticket.

Созданные таблицы:

Таблица «animal», в которой хранится информация о животных:

```
CREATE TABLE `animal` (
  `id` int(11) NOT NULL,
  `name` varchar(200) COLLATE utf8mb4_unicode_ci NOT NULL,
  `image` varchar(200) COLLATE utf8mb4_unicode_ci NOT NULL,
  `description` text COLLATE utf8mb4_unicode_ci NOT NULL,
  `id_category` int(11) NOT NULL
) ;
```

Таблица «category», в которой храниться категории животных (Млекопитающие, рабы, птицы, земноводные, рептилии и беспозвоночные):

```
CREATE TABLE `category` (
  `id` int(11) NOT NULL,
  `namr` varchar(200) COLLATE utf8mb4_unicode_ci NOT NULL
) ;
```

Таблица «comment», в которой храниться все комментарии пользователей:

```
CREATE TABLE `comment` (
  `id` int(11) NOT NULL,
  `id_user` int(11) NOT NULL,
  `date` date NOT NULL DEFAULT current_timestamp(),
  `text` text NOT NULL
) ;
```

Таблица «excursion», в которой представлены все экскурсии в зоопарке:

```
CREATE TABLE `excursion` (
  `id` int(11) NOT NULL,
  `name` varchar(200) NOT NULL,
  `total_person` int(11) NOT NULL,
  `price` int(11) NOT NULL
) ;
```

Таблица «ticket», в которой храниться информация о видах билетов и их ценам:

```
CREATE TABLE `ticket` (
  `id` int(11) NOT NULL,
  `name` varchar(200) NOT NULL,
  `price` int(11) NOT NULL
) ;
```

Таблица «users» с зарегистрированными пользователями:

```
CREATE TABLE `users` (
```

```
  `id` int(11) NOT NULL,
  `fio` varchar(200) NOT NULL,
  `login` varchar(200) NOT NULL,
  `password` varchar(200) NOT NULL
) ;
```

Таблица «waliery» с информацией о вольерах, в которых обитают животные:

```
CREATE TABLE `waliery` (
  `id` int(11) NOT NULL,
  `id_waliery` int(11) NOT NULL,
  `name` varchar(200) COLLATE utf8mb4_unicode_ci NOT NULL,
  `img` varchar(200) COLLATE utf8mb4_unicode_ci NOT NULL,
  `description` text COLLATE utf8mb4_unicode_ci NOT NULL
) ;
```

Таблица с заказами:

```
CREATE TABLE `zakaz` (
  `id` int(11) NOT NULL,
  `id_user` int(11) NOT NULL,
  `date` date NOT NULL
) ;
```

Таблица с заказанными экскурсиями:

```
CREATE TABLE `zakaz_excursion` (
  `id` int(11) NOT NULL,
  `id_vid` int(11) NOT NULL,
  `id_zakaz` int(11) NOT NULL,
  `count` int(11) NOT NULL
) ;
```

Таблица с заказанными билетами:

```
CREATE TABLE `zakaz_ticket` (
  `id` int(11) NOT NULL,
  `count` int(11) NOT NULL,
  `id_ticket` int(11) NOT NULL,
  `id_zakaz` int(11) NOT NULL
) ;
```

Запросы:

1. Примеры функциональных пользовательских запросов:

1.1 Регистрация пользователя:

```
SELECT COUNT(*) FROM users WHERE login = '".$_POST['login']."'
```

```
INSERT INTO users(fio,login,password) VALUES ('".$_POST['fio']."','".$_POST['login']."','".$_POST['password']."')
```

Где Users-Таблица зарегистрированных пользователей сайта;
Login- Логин пользователя;
$_POST['login']- id пользователя;
users(fio,login,password)- вставляются данные пользователя в таблицу;
$_POST['fio'], $_POST['login'], $_POST['password']-данные пользователя, которые вносятся в таблицу для регистрации.

1.2 Вход пользователя:

```
SELECT COUNT(*) FROM users WHERE login = '".$_POST['login']."' AND password = '".$_POST['password']."'
```

Где Users-таблица зарегистрированных пользователей сайта;
Login - логин пользователя;
Password – пароль пользователя;
$_POST['login'] - логин;
$_POST['password'] - пароль;

1.3 Добавление заказа:

```
INSERT INTO `zakaz`(`id_user`,`date`) VALUES(".$_SESSION['user'][0].",\"".$_POST['date-zakaz']."\")
```

Zakaz- таблица с заказами;
id_user- id пользователя;
date- дата заказа;

1.4 Заказ услуг зоопарка:

```
SELECT id FROM zakaz WHERE date = '".$_POST['date-zakaz']."' AND id_user=".$_SESSION['user'][0]." ORDER BY id DESC LIMIT 1 ";
```

id – id заказа;
Zakaz- таблица с заказами;
date- дата заказа;
id_user – id пользователя;

1.5 Добавление заказанных экскурсий:

```
INSERT INTO `zakaz_excursion`(`id_vid`,`id_zakaz`,`count`) VALUES(".$ex['id'].",".$id_zakaz.",".$ex['count'].")
```

zakaz_excursion- таблица с заказанными экскурсиями;
id_vid- id вида экскурсий;

id_zakaz- id заказа;
count –количество посетителей экскурсиями;

1.6 Добавление заказанных взрослых билетов в зоопарк:

```
INSERT INTO zakaz_ticket(count,id_ticket,id_zakaz)
VALUES(".$tickets['countOld'].",1,".$id_zakaz.")
```

zakaz_ticket- таблица заказанных билетов;
count- количество купленных взрослых билетов;
id_ticket- id билета;
id_zakaz – id заказа;

1.7 Добавление детских билетов в зоопарк:

```
INSERT INTO zakaz_ticket(count,id_ticket,id_zakaz)
VALUES(".$tickets['countChild'].",2,".$id_zakaz.")
```

zakaz_ticket- таблица заказанных билетов;
count- количество купленных детских билетов;
id_ticket- id билета;
id_zakaz – id заказа;

1.8 Добавление комментария:

```
INSERT INTO `comment`(`id_user`,`text`)
VALUES(".$_SESSION['user'][0].",\"".$_POST['text']."\")\
```

Comment-таблица комментариев;
id_user – Id пользователя;
text – текст комментария;

2. Примеры информационных-агрегаитвных запросов :

2.1 Комментарии:

```
SELECT COUNT(*) FROM comment
```

Comment- таблица комментариев;

2.2 Получение информации о вольере:

```
SELECT * FROM waliery WHERE id_waliery = ".$_GET['id'],
```

Где waliery- таблица Вольеров;
id_waliery – столбец ID по которому выбирается вольер;
$_GET['id'] - id выбранного вольера;

2.3 Категории к которым относятся животные:

```
SELECT * FROM animal WHERE id_category = 1
SELECT * FROM animal WHERE id_category = 2
SELECT * FROM animal WHERE id_category = 3
SELECT * FROM animal WHERE id_category = 4
SELECT * FROM animal WHERE id_category = 5
SELECT * FROM animal WHERE id_category = 6
```

Animal- таблица с животными;
id_category – id категории, к которой относится животное;

2.4 Получение описания конкретного животного:

```
SELECT * FROM animal WHERE id = ".$_GET['id_animal']
```

Id- id животного;
Animal- таблица животных зоопарка;

2.5 Получение комментария:

```
SELECT comment.id,users.fio,comment.text,comment.date
                                FROM comment
                                INNER JOIN users
                                ON comment.id_user = users.id
```

Comment-таблица комментариев;
comment.id- id комментария ;
users.fio- пользователь;
comment.text- текст комментария;

comment.date- дата комментария;

Вывод: В ходе выполнения лабораторной работы реализовала систему обмена данными с БД.