Міністерство освіти і науки України
Національний університет «Запорізька політехніка»

МЕТОДИЧНІ ВКАЗІВКИ
до лабораторних робіт
з дисципліни «Бази даних»
«Робота з СКБД MySQL»
для студентів спеціальності
122 «Комп'ютерні науки»
усіх форм навчання

2021

Методичні вказівки до лабораторних робіт з дисципліни «Бази даних». «Робота з СКБД MySQL» для студентів спеціальності 122 «Комп'ютерні науки» усіх форм навчання /Уклад.: С.К. Корнієнко – Запоріжжя: НУ «Запорізька політехніка», 2021. – 34 с.

Укладач: С.К. Корнієнко, доцент, к.т.н.

Рецензент: О.О. Степаненко, доцент, к.т.н.

Відповідальний
за випуск: С.О. Субботін, професор, д.т.н.

Затверджено
на засіданні кафедри
«Програмні засоби»

Протокол № 1 від 18.08.2020 р.

ЗМІСТ

1 Лабораторна робота № 1 «Знайомство з пакетом «Денвер» 4
2 Лабораторна робота № 2 «Створення бази даних» 10
3 Лабораторна робота № 3 «Проектування запитів»..... 14
4 Лабораторна робота № 4 «Робота з поданнями»..... 20
5 Лабораторна робота № 5 «Вбудовані процедури та курсори»..... 24
Література 30
Додаток А Інсталяція пакету Денвер..... 31

1 ЛАБОРАТОРНА РОБОТА № 1 «ЗНАЙОМСТВО З ПАКЕТОМ «ДЕНВЕР»

1.1 Мета роботи

Метою роботи є знайомство з пакетом ДЕНВЕР і реєстрація нової бази даних.

1.2 Завдання до лабораторної роботи

1.2.1 Ознайомитися зі змістом пункту 1.3 даних методичних указівок.

1.2.2 Перейти до директорії **`Z:\denwer`** та увімкнути пакет за допомогою файлу **`Run`**. Наприкінці роботи необхідно обов'язково вимкнути пакет «Денвер» за допомогою відповідного файлу. Якщо завдання будуть виконуватись самостійно на комп'ютері, де ***НЕ встановлено пакет*** **Денвер**, то спочатку необхідно його встановити (додаток А)

Застосовуючи браузер **Mozilla Firefox** перейдіть до ресурсу http://localhost:8088 На моніторі з'явиться наступне (рис. 1.1):

URL	Описание
https://subdomain.localhost/ssl.php	Проверка SSL
http://subdomain.localhost/	Проверка "не-Интернет" доменов второго уровня, а также SSI
http://test1.ru/	Проверка "Интернет"-доменов второго уровня: test1.ru (вначале отключите прокси-сервер!)
http://subdomain.test1.ru/	Проверка "Интернет"-доменов третьего уровня
http://localhost/Tests/phpnotice/index.php	Проверка перехвата PHP Notice в Денвере
http://localhost/Tests/PHP5/index.php5	PHP5 information

Рисунок 1.1 – Головна сторінка з посиланнями на компоненти пакету «Денвер»

Для коректних переходів необхідно обов'язково додавати новий порт 8088 до усіх локальних ресурсів одразу після адреси хоста:

http://localhost:8088/Tests/PHP5/index.php5

1.2.3 Створити новий хост. Для цього:

– відкрити каталог, у який була проведена інсталяція пакету;

– у каталозі пакету із назвою **`home`** створити власну директорію з такою назвою, яку буде мати домен сайту. Якщо майбутній сайт повинен мати адресу **http://test.ua**, то створюємо директорію **`test.ua`**;

– у створену директорію необхідно додати каталоги: **`www`** (для сторінок сайту) та **`cgi`** (для cgi-скриптів);

– виконати перезавантаження пакету «Денвер» за допомогою утиліт, які знаходяться у каталозі **`denwer`**;

– у браузері набрати адресу щойно створеного сайту.

1.2.4 Виконати підготовку до створення нової бази даних. Для цього необхідно:

– перейти до інтерфейсу phpMyAdmin, який дозволяє керувати базою даних;

– створити нового користувача.

Для переходу до інтерфейсу phpMyAdmin треба:

– запустити «Денвер»;

– у адресному рядку браузера набрати http://localhost:8088

– натиснути кнопку оновлення сторінки;

– перейти до пункту http://localhost:8088/Tools/phpMyAdmin як показано на рис. 1.2.

URL	Описание
https://subdomain.localhost/ssl.php	Проверка SSL
http://subdomain.localhost/	Проверка "не-Интернет" доменов второго уровня, а также SSI
http://test1.ru/	Проверка "Интернет"-доменов второго уровня: test1.ru (вначале отключите прокси-сервер!)
http://subdomain.test1.ru/	Проверка "Интернет"-доменов третьего уровня
http://localhost/Tests/phpnotice/index.php	Проверка перехвата PHP Notice в Денвере
http://localhost/Tests/PHP5/index.php5	PHP5 information
http://localhost/Tools/phpMyAdmin	Проверка MySQL и phpMyAdmin
http://custom-host:8648	Проверка хоста с другим IP-адресом и портом (127.0.0.2:8648) *В Windows XP SP2 имеется ошибка, из-за которой данный хост может не работать. Официальную "заплатку" от Microsoft качайте здесь.*
http://localhost/Tests/sendmail/index.php	Проверка отладочной заглушки для sendmail

Рисунок 1.2– Початкова сторінка пакету «Денвер»

Загальна сторінка phpMyAdmin наведена на рис. 1.3.

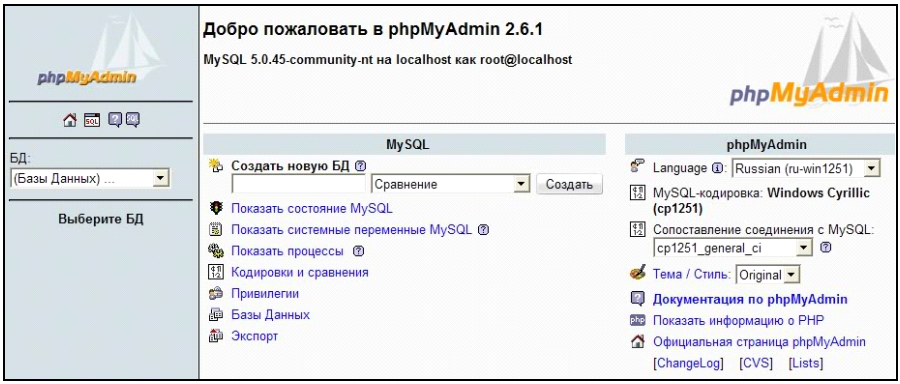

Рисунок 1.3– Інтерфейс phpMyAdmin

Для створення нового користувача необхідно:

– зайти на загальну сторінку http://localhost:8088;

– перейти прокруткою до розділу «Утилиты» (рис. 1.4);

– перейти до підрозділу «Заведение новых БД и пользователей MySQL» (рис. 1.5);

– додати назву бази даних, логін та пароль;

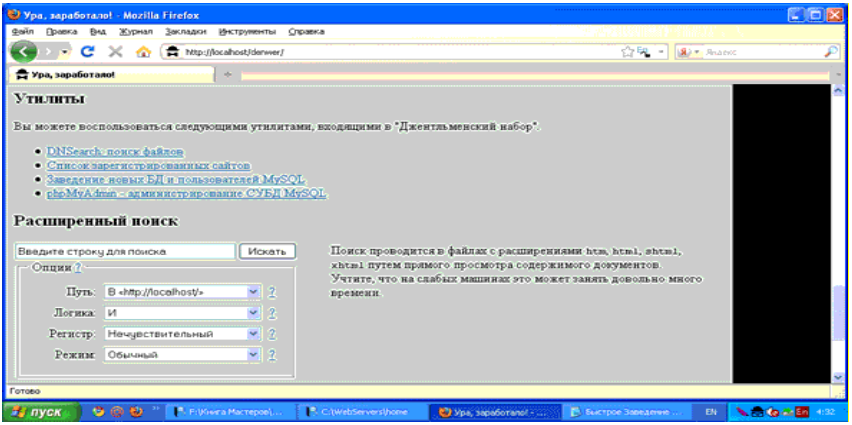

Рисунок 1.4 – Розділ «Утилиты»

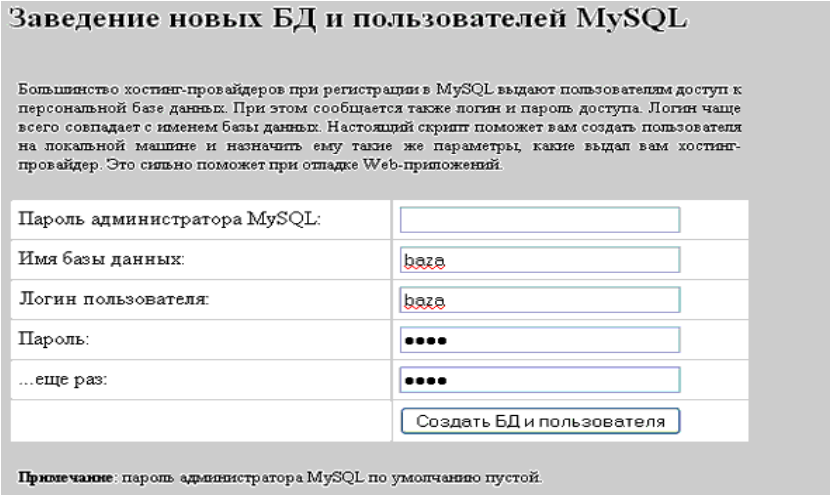

Рисунок 1.5 – Додавання нових БД та користувачів MySQL

1.3 Основні теоретичні відомості

ДЕНВЕР – набір дистрибутивів і програмна оболонка, зібрані для створення та відлагодження сайтів на локальній (домашній) Windows-машині без необхідності підключення до мережі Інтернет. Назвою пакету є скорочення від "***джентльменський набір web-розробника***" («Д.н.w.р», читається «Денвер»).

Денвер – ідеальний набір для початкуючого web-програміста. Відразу після установки існує доступ до повністю працюючого серверу Apache, на якому може працювати необмежена кількість сайтів. Щоб створити новий сайт, досить створити нову теку. Щоб побачити новий код у дії, досить зберегти файл і відновити сторінку у браузері. Також є можливість розміщення сайту на Flash-накопичувачі.

Склад базового пакету:

1. Веб-сервер ***Apache*** з підтримкою SSL і mod_rewrite.
2. ***PHP5***: виконавчі файли, модуль для веб-сервера Apache, дистрибутивний і адаптований конфігураційний файл, бібліотека GD, модулі підтримки MySQL і SQLite.
3. ***Mysql5*** з підтримкою Innodb, транзакцій і російських кодувань (windows-1251).
4. ***HPmyadmin*** – візуальний інтерфейс, написаний на PHP, для роботи з MySQL. PHPmyadmi дозволяє користуватися можливостями браузера. Багато з базових SQLфункций роботи з даними в PHPmyadmi зведені до інтуїтивно зрозумілих інтерфейсів і дій, що нагадують перехід за посиланнями в Internet.
6. Відлагоджувальний емулятор sendmail (/usr/sbin/sendmail), який не відправляє листа, а записує їх у директорію /tmp/!sendmail.
7. Система автоматичного пошуку віртуальних хостів і відновлення системного файлу hosts, а також конфігурації Apache.

Apache HTTP-сервер – відкритий веб-сервер Інтернет для UNIX-подібних, Microsoft Windows, Novell NetWare та інших операційних систем. На сьогодні є найпопулярнішим веб-сервером мережі Інтернет. Він є самостійним, некомерційним, вільно розповсюджуваним продуктом. Продукт підтримує безліч можливостей, багато з яких реалізовані як скомпільовані модулі, що

розширюють основні функціональні можливості. Вони різняться від серверної підтримки мов програмування до схем аутентифікації.

Існують інтерфейси для підтримки мов програмування Perl, Python, Tcl і PHP. Популярні методи стискування на Apache включають зовнішній модуль mod_gzip, створений для зменшення розміру веб-сторінок, переданих по HTTP. Функції віртуального хостингу дозволяють одній інсталяції Apache обслуговувати різні веб-сайти. Наприклад, одна машина з однією інсталяцією Apache може одночасно містити www.example.com, www.test.com, test47.test-server.test.com та інші.

MySQL – безкоштовна система керування реляційними базами даних (СКБД). Дана СКБД з відкритим кодом була створена як альтернатива комерційним системам. Зараз MySQL – одна з найпоширеніших СКБД. Це досить швидка, надійна та, головне, проста у використанні СКБД, яка цілком підходить для не занадто глобальних проектів.

PHP (*Hypertext Preprocessor* – гіпертекстовий препроцесор) – одна з найпоширеніших мов, які використовуються у сфері веб-розробок (разом із Java, .NET, Perl, Python, Ruby). PHP підтримується переважною більшістю хостинг-провайдерів. Вона використовується, у першу чергу, для створення динамічних веб-сторінок з боку веб-сервера, оскільки має чудову підтримку з боку різноманітних мов програмування. PHP — проект відкритого програмного забезпечення.

Контрольні питання

- Що таке пакет «Денвер»?
- Які компоненти містить паке пакет «Денвер» та призначення кожного з компонент?
- Як створити новий хост або сторінку користувача у пакеті «Денвер»?
- Призначення компоненту phpMyAdmin?
- Як зареєструвати нову БД?
- Як створити нового користувача у системі?

2 Лабораторна робота № 2 «Створення бази даних»

2.1 Мета роботи

Метою роботи є надбання навичок створення баз даних за допомогою СКБД MySQL пакету ДЕНВЕР.

2.2 Завдання до лабораторної роботи

2.2.1 Ознайомитися зі змістом пункту 2.3 даних методичних указівок.

2.2.2 Згідно з узгодженою з викладачем предметною областю створити відповідну базу даних.

2.2.3 Наповнити створені таблиці даними.

2.3 Основні теоретичні відомості

2.3.1 Створення нової бази даних:

– перейти на головну сторінку phpMyAdmin (рис. 2.1);

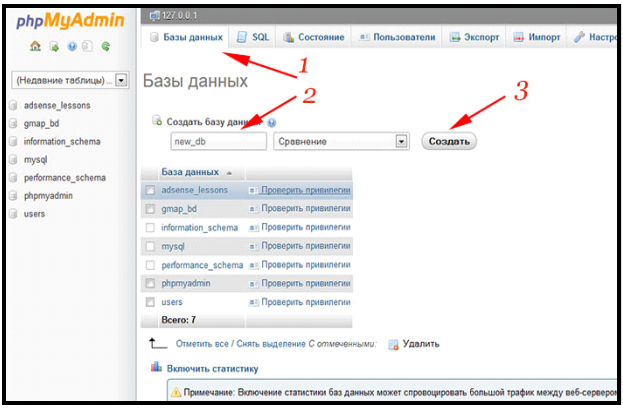

Рисунок 2.1 – Керування базами даних

– перейти до розділу «Базы данных»;

– у полі «Создать базу данных» додати назву нової бази та підтвердити кнопкою «Создать».

Після натискання кнопки «Создать» база з новим ім’ям

додається до списку баз даних на панелі ліворуч і до таблиці на центральному полі. Нова база даних обирається натисканням по її назві. Відкриється розділ створення та редагування таблиць (рис. 2.2)

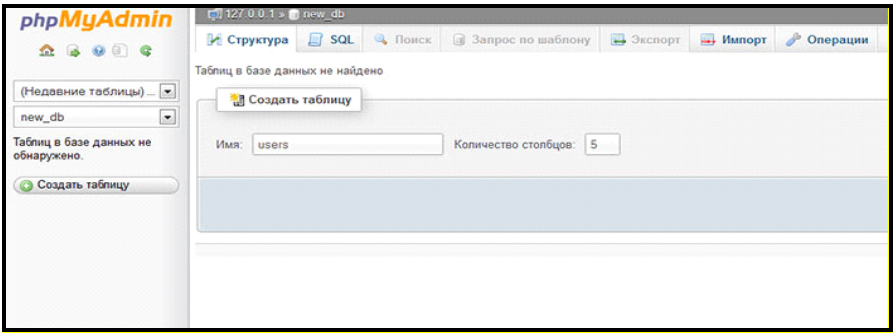

Рисунок 2.2 – Створення таблиці

2.3.2 Створення та редагування таблиць:

– назву таблиці необхідно додати до поля «Имя» та ввести кількість стовпців таблиці;

– у розділі редагування полів таблиці (рис. 2.3) додаються для кожного поля: назва, тип даних, розмір (якщо необхідно) та інші атрибути, такі як первинний ключ або авто-інкремент;

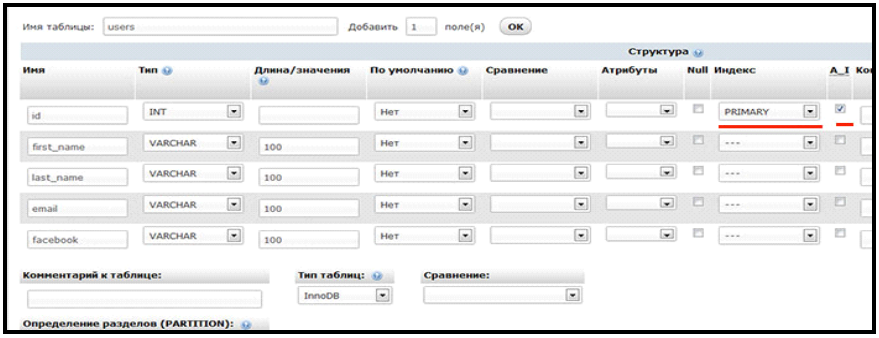

Рисунок 2.3 – Редагування полів таблиці

– таблиця з новим ім’ям додається до списку таблиць на панелі

ліворуч. Якщо її відмітити, то на центральному полі з'являється структура даної таблиці, яку можна відредагувати (рис. 2.4);

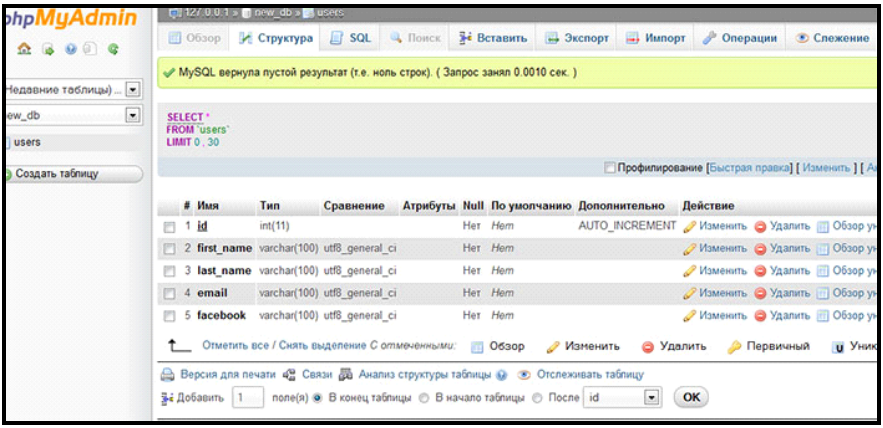

Рисунок 2.4 – Редагування структури таблиці

– для додавання нового елементу до таблиці необхідно перейти на вкладку «Вставить» (рис. 2.5). Заповніть усі поля (окрім поля id, воно буде заповнюватися автоматично);

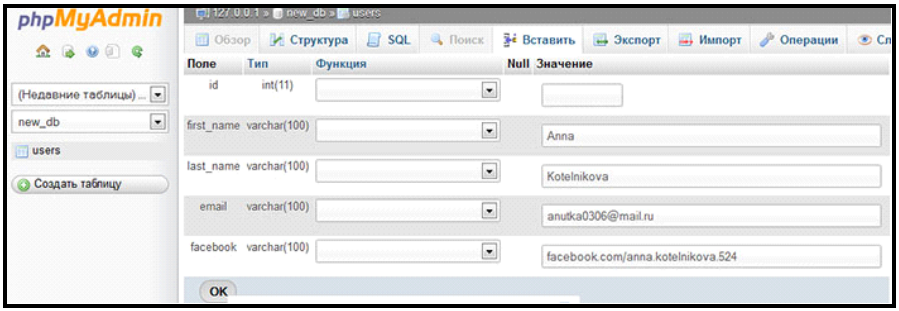

Рисунок 2.5 – Додавання елемента у таблицю

– на вкладці «Обзор» можна переглянути новий елемент, доданий до таблиці (рис. 2.6);

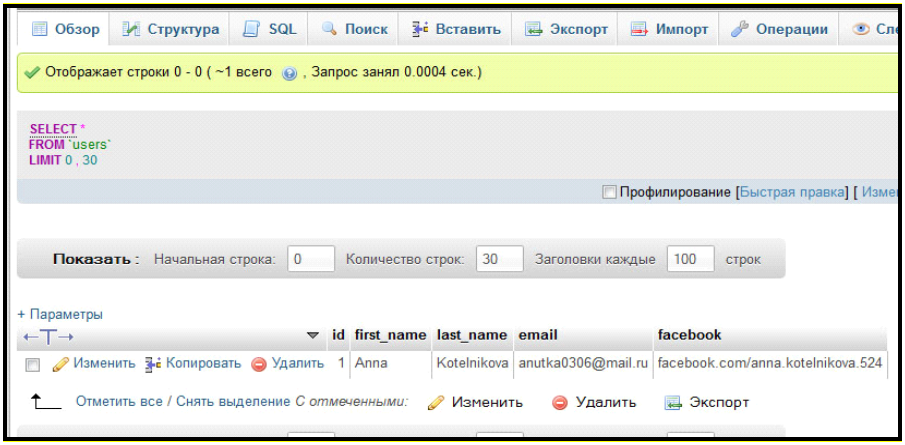

Рисунок 2.6 – Огляд таблиці

Контрольні питання

- Як відредагувати існуючу базу даних?
- Як створити нову таблицю у БД (різними способами)?
- Як відредагувати існуючу таблицю, структуру таблиці?
- Які альтернативни методи створення баз даних існують?

3 Лабораторна робота № 3 «Проектування запитів»

3.1 Мета роботи

Метою роботи є придбання навичок розробки запитів в середовищі MySQL.

3.2 Завдання до лабораторної роботи

3.2.1 Ознайомитися зі змістом пункту 3.3 методичних указівок.

3.2.2 Отримати від викладача завдання, згідно з яким розробити відповідні запити до створеної в попередній роботі БД, використовуючи можливості СКБД MySQL.

3.2.3 Оформити звіт, який повинен містити концептуальну модель предметної області, схему БД, тексти запитів та результати їхнього виконання.

3.3 Основні теоретичні відомості

3.3.2 Розробка запитів до бази даних

Загальний формат команди SELECT такий:

SELECT [***ALL***|***DISTINCT***] список полів, що вибираються |*
FROM список таблиць
[***WHERE*** умова вибірки]
[***GROUP BY*** умова угруповання
[***HAVIHG*** умова вибірки групи]]
[***ORDER BY*** умова впорядкування];

DISTINCT – аргумент, який усуває дублювання значень із результату виконання речення ***SELECT.*** Альтернативою ***DISTINCT*** є ключове слово ***ALL***, яке використовується по умовчанню .

<u>Вибір рядків: речення WHERE</u>

Речення ***WHERE*** задає предикат, умову, що може бути вірною або помилковою для кожного рядка таблиці. У результаті вибираються тільки ті рядки з таблиці, для яких предикат має значення "істина". У табл. 3.2 наведені оператори та ключові слова для завдання умов вибору.

Таблиця 3.2 – Оператори завдання умов

Оператор, ключове слово	Приклад використання
оператори порівняння ***(=, <, >, >=, <=, <>, ! =);***	where ціна > 1000;
комбінації умовних і логічних операцій - (***AND,OR,NOT***)	where ціна < 5000 and ціна >2000
діапазони (***BETWEEN і NOT BETWEEN***)	where ціна between 450 and 500
списки (***IN, NOT IN)***	where товар in ("монітор", "принтер")
невідомі значення (***IS NULL*** і ***IS NOT NULL)***	where телефон is null
відповідності символів ***(LIKE*** і ***NOT LIKE)***	where телефон not like "415%"; where назва like "*БД*"

<u>Агрегатні функції</u>

Агрегатні функції (табл. 3.3) використовуються для одержання узагальнюючих значень. Вони дають єдине значення для цілої групи рядків таблиці

Таблиці 3.3 - Агрегатні функції

Функція	Результат
COUNT	Визначає кількість рядків або значень поля, обраних за допомогою запиту, що ***не є NULL-значеннями***
SUM	Обчислює арифметичну суму всіх обраних значень даного поля
AVG	Обчислює середнє значення для всіх обраних значень даного поля
MAX	Обчислює найбільше з всіх обраних значень даного поля
MIN	Обчислює найменше з всіх обраних значень даного поля

<u>Групування даних</u>

Речення ***GROUP BY*** розділяє таблицю на набори, а агрегатна функція обчислює для кожного з них підсумкове значення. Ці значення називаються ***агрегатним вектором.***

```
SELECT список вибору
        FROM список таблиць
        [WHERE умови]
        [GROUP BY  список_угруповання];
```

Наприклад:

```
select художник, count(картина)
    from Музеї
      group by  художник;
```

До списку вибору включаються стовпчик, за яким проводиться угруповання, та агрегатна функція.

<u>Речення HAVING</u>

Речення ***WHERE*** накладає обмеження на рядки, а ***HAVING -*** на групи. Як правило, речення ***HAVING*** використовується разом із реченням ***GROUP BY.***

Якщо в списку вибору є агрегатні функції, речення ***WHERE*** виконується перед ними, тоді як речення ***HAVING*** застосовується до всього запиту в цілому, після розбивки на групи та обчислення значень функцій.

В умові речення ***WHERE*** не можуть знаходитися агрегатні функції. Крім того, елементи речення ***HAVING*** повинні включатися до списку вибору. На речення ***WHERE*** це обмеження не поширюється. У реченні ***HAVING*** може міститися будь-яка кількість умов.

```
select тип, count(*)
    from Товари
    group by тип
      having count(*) > 1;
```

```
select автор
    from Книги
      group by автор
        having автор like "П%";
```

<u>З'єднання таблиць</u>

Кожне з'єднання задається для двох таблиць, хоча в одному операторі ***SELECT*** може виконуватися декілька з'єднань. При цьому стовпчики, за якими виконується з'єднання, повинні бути ***сумісними для об'єднання.***

```
SELECT список_вибору
    FROM таблиця_1, таблиця_2 [ , таблиця_3]…
        WHERE[ таблиця_1.]стовпчик оператор_з'єднання
                [ таблиця_2.] стовпчик
```

Операція INNER JOIN

Вона використовується у реченні FROM і з'єднує записи двох таблиць, якщо поля, за якими проводиться з'єднання, містять однакові значення. Ця операція також називається ***внутрішнім з'єднанням.***

В наступному прикладі операція використовується з таблицями ***Відділи*** та ***Співробітники*** для вибірки всіх співробітників кожного відділу.

```
SELECT B.Ном,C.Прізв
        FROM Відділи B INNER JOIN Співробітники C
        ON B.Ном=C.Ном_від;
```

Операція LEFT JOIN, RIGHT JOIN

Операція LEFT JOIN використовується для створення ***лівого зовнішнього з'єднання,*** до якого включаються всі записи з першої (лівої) таблиці, навіть якщо немає співпадаючих значень для записів із другої (правої) таблиці.

Операція JOIN використовується для створення ***правого зовнішнього з'єднання,*** до якого включаються всі записи з другої (правої) таблиці, навіть якщо немає співпадаючих значень для записів із першої (лівої) таблиці.

Наприклад, операцію LEFT JOIN можна використовувати з таблицями ***Відділи*** (ліва) та ***Співробітники*** (права) для вибірки всіх відділів, включаючи ті, в яких немає співробітників.

```
SELECT B.Ном,C.Прізв
        FROM Відділи B LEFT JOIN Співробітники C
        ON B.Ном=C.Ном_від;
```

Якщо потрібно вибрати всіх співробітників, включаючи тих, які не приписані до жодного відділу, то використовується операція RIGHT JOIN.

SELECT B.Ном,C.Прізв
FROM Відділи B RIGHT JOIN Співробітники C
ON B.Ном=C.Ном_від;

<u>Запити на об'єднання</u>

Операція ***Union*** може об’єднувати результати декількох запитів, таблиць або інструкцій SQL. Наприклад,

TABLE [Замовлення]
Union
TABLE [Замовлення_архів];

Для об'єднання двох і більш запитів необхідно, щоб їхні стовпчики, що входять у набір вихідних даних, були ***сумісні по об'єднанню.***

Контрольні питання

– Які типи даних використовуються у SQL?
– Як можна задати значення по замовчанню у числові поля?
– Як задається ключове поле?
– Які властивості поля підтримує SQL?
– Як задаються обмеження на стовпчики таблиці?
– Що таке ***первинні*** та ***зовнішні ключі***?
– Основний формат команди ***Select?***
– Які оператори використовуються для завдання умов вибірки даних?
– Що таке агрегатні функції?
– Для чого використовується групування даних у запитах?
– Що таке з’єднання таблиць, які існують різновиди цієї операції?
– Для чого використовуються запити на об’єднання?
– Послідовність виконання логічних і умовних операторів у предикаті запиту?
– Які оператори маніпулювання даними використовуються у MySQL?

4 Лабораторна робота № 4 «Робота з поданнями»

4.1 Мета роботи

Метою роботи є придбання навичок створення та використання подань в середовищі MYSQL.

4.2 Завдання до лабораторної роботи

4.2.1 Ознайомитися зі змістом пункту 4.3 даних методичних указівок.

4.2.2 Згідно з отриманим завданням створити відповідні об’єкти БД. Навести приклади їхнього використання.

4.3 Основні теоретичні відомості

Бувають ситуації, коли користувачеві необхідно дозволити доступ не до всієї інформації, що зберігається в таблиці, а лише до її частини: наприклад, необхідно, щоб продавці могли переглядати таблицю Salespeople, але не могли бачити комісійні своїх колег. Або ситуація, коли щодня необхідно одержувати інформацію на основі різних таблиць за допомогою складних запитів. Для того, щоб не повторювати щораз складний запит, можна створити на його основі подання (VIEW).

Подання – це динамічний результат однієї або декількох реляційних операцій над базовими відношеннями з метою створення деякого іншого відношення.

Подання є ***віртуальним відношенням*** (***virtul relation***), якого реально в базі даних не існує, але яке створюється на вимогу окремого користувача в момент надходження цієї вимоги.

З погляду користувача подання є відношенням, що постійно існує та з яким можна працювати, як із базовим відношенням.

Однак подання не зберігається в базі даних так, як базові відношення (хоча його визначення зберігається в системному каталозі). Вміст подання визначається як результат виконання запиту до одного або декількох базових відношень.

Подання – це вікно, через яке користувач може побачити інформацію з БД. При зверненні до подання виконується запит до

базової таблиці або таблиць, на основі яких створено подання.

Наприклад, необхідно дозволити покупцям бачити в таблиці ***Товари*** тільки дані про назву товара та ціну відпускну. У цьому випадку можна створити подання ***Прайс-лист*** на основі таблиці ***Товари***, а потім дозволити всім покупцям читати інформацію з цього подання.

create view Прайс-лист
as
select назва, ціна_відпускна as ціна
from Товари;

Зверніть увагу на те, що в пропозиції CREATE VIEW після імені подання не зазначені імена стовпців, з яких воно буде складатися. Тому беруться імена стовпців базової таблиці або їхні псевдоніми, на основі яких будується подання. Якщо при створенні подання використовуються агрегатні функції, то такі стовпці повинні бути названі в пропозиції CREATE VIEW.

Подання, засновані на запиті, що використовує ***речення GROUP BY,*** називаються ***груповими***. При цьому допускається використання речення ***HAVING***.

При створенні подання можна використовувати з'єднання таблиць. Наприклад, в БД є три таблиці:

Покупці(Код,Прізвище,Тел),
Продавці (Код, Прізвище),
Замовлення (Ном, Покупець, Дата, Вартість, Продавець).

Необхідно визначити за минулий місяць всіх покупців з їхніми замовленнями та продавців, які їх обслуговували:

CREATE VIEW Покупці_Замовлення
AS
SELECT П.Прізвище as Покупець, Пр.Прізвище as Продавець, З.Дата, З.Вартість
FROM Замовлення З, Продавці Пр, Покупці П
WHERE З.Покупець=П.Код and З.Продавець=Пр.Код
and month(date())-month (З.Дата)=1;

Це подання побудовано на основі запиту, що з'єднує три таблиці. Звернутися до нього можна також через запит. Наприклад, нам необхідно одержати всі замовлення покупця Іванова в минулому місяці:

SELECT *
FROM Покупці_Замовлення
WHERE Покупець='Іваненко';

Будь-які операції над поданням автоматично транслюються в операції над відношеннями, на підставі яких це подання було створено. Подання мають ***динамічну*** природу, тобто зміни в базових відносинах, які можуть вплинути на вміст подання, негайно відбиваються на вмісті цього подання.

Якщо користувачі вносять у подання деякі припустимі зміни, ці зміни негайно заносяться до відповідних базових таблиць.

На підставі таблиці ***Продавці*** (Код, Прізвище, Місто, Рейтінг) створимо подання:

CREATE VIEW Топ_Продавці
AS
SELECT *
FROM Продавці
WHERE Рейтінг=300;

А тепер виконаємо команду оновлення

INSERT INTO Топ_Продавці VALUES (201, 'Гриценко', 'Запоріжжя', 250);

Ця команда виконається й у базову таблицю буде вставлений запис, але до подання запис не буде доданий, тому що значення Рейтінг не рівно 300.

Для того, щоб відслідковувати такі ситуації, можна скористатися реченням ***WITH CHECK OPTION***, яке вставляється в команду створення подання.

CREATE VIEW Топ_Продавці
AS
SELECT *
FROM Продавці
WHERE Рейтінг=300;
WITH CHECK OPTION;

Тепер така команда ***INSERT*** буде відхилена.

Для створення подання треба у вкладці «SQL» інтерфейсу PHPmyAdmin ввести текст відповідного запиту на створення (рис. 4.1):

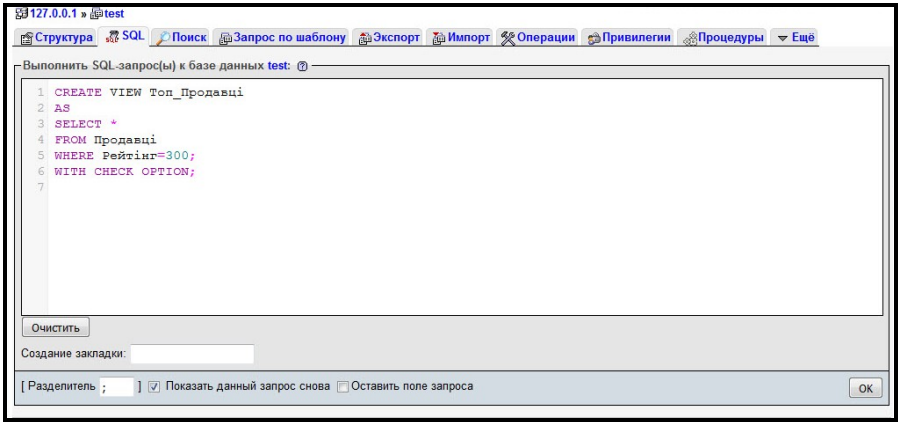

Рисунок 4.1 – Вікно створення подання

Контрольні питання

- Яка різниця між базовим і віртуальним відношеннями?
- Що таке подання, його призначення?
- Як створити подання?
- Як використовується операція з’єднання при створенні подання?
- Для чого використовується речення WITH CHECK OPTION?

5 Лабораторна робота № 5 «Вбудовані процедури та курсори»

5.1 Мета роботи

Метою роботи є придбання навичок створення та використання вбудованих процедур і курсорів в середовищі MYSQL.

5.2 Завдання до лабораторної роботи

5.2.1 Ознайомитися зі змістом пункту 5.3 даних методичних указівок.

5.2.2 Згідно з отриманим завданням створити відповідні об’єкти БД. Навести приклади їхнього використання.

5.3 Основні теоретичні відомості

5.3.1 Вбудовані процедури

Процедура – це програмна функція, яка створюється та зберігається у базі даних MySQL. Вона може складатися з SQL-операторів та низки спеціальних керуючих структур. Збережена процедура може бути корисна, коли одну і ту ж функцію необхідно викликати з різних додатків або з різних платформ або як засіб інкапсуляції функціональних можливостей. Збережені процедури в базах даних можна вважати аналогом об’єктно-орієнтованого підходу у програмуванні. Вони дозволяють керувати способом доступу до даних.

<u>Створення збереженої процедури</u>

```
CREATE PROCEDURE `p2` ()
LANGUAGE SQL
DETERMINISTIC
SQL SECURITY DEFINER
COMMENT 'A procedure'
BEGIN
   SELECT 'Hello World !';
END;/
```

Перша частина коду створює збережену процедуру. Наступна - містить необов'язкові параметри. Потім йде назва та, нарешті, тіло самої процедури.

Назви збережених процедур чутливі до регістру. Не можна створювати кілька процедур з однаковою назвою. Усередині збереженої процедури не може бути виразів, які змінюють саму базу даних.

Характеристики збереженої процедури:

– ***Language***: з метою забезпечення переносимости, за замовчуванням зазначена мова SQL;

– ***Deterministic***: якщо процедура увесь час повертає той самий результат, і приймає ті ж самі вхідні параметри. Це для реплікації й процесу реєстрації. Значення за замовчуванням – ***NOT DETERMINISTIC***;

– ***SQL Security:*** під час виклику йде перевірка прав користувача. ***INVOKER*** – це користувач, що викликає збережену процедуру. ***DEFINER*** – це “творець” процедури. Значення за замовчуванням – ***DEFINER***;

– ***Comment***: з метою документування, за замовчуванням – "".

Виклик збереженої процедури

Для виклику збереженої процедури необхідно ввести ключове слово CALL, потім назву процедури, а в дужках указати параметри (змінні або значення). Дужки обов'язкові.

CALL stored_procedure_name (param1, param2, ...)

CALL procedure1(10, 'string parameter', @parameter_var);

Видалення збереженої процедури

DROP PROCEDURE IF EXISTS p2;

Вираз ***IF EXISTS*** відловлює помилку в разі відсутності такої процедури .

<u>Параметри</u>

Параметри можна передавати в збережену процедуру в такий спосіб:

– ***CREATE PROCEDURE*** proc1 (): порожній список параметрів;

– ***CREATE PROCEDURE*** proc1 (***IN*** varname DATA-TYPE): один вхідний параметр. Слово IN необов'язково, тому що за замовчуванням параметри визначаються як вхідні (IN);

– ***CREATE PROCEDURE*** proc1 (***OUT*** varname DATA-TYPE): один параметр, що повертається.

– ***CREATE PROCEDURE*** proc1 (***INOUT*** varname DATA-TYPE): один параметр, одночасно вхідний, та такий, що повертається.

Аналогічно можна задавати кілька параметрів різних типів.

<u>Змінні</u>

Синтаксис оголошення змінної виглядає так:

DECLARE ім'я_змінної тип_даних DEFAULT значення за замовчуванням

Наприклад,

DECLARE a, b INT DEFAULT 5;

DECLARE str VARCHAR(50);

DECLARE today TIMESTAMP DEFAULT CURRENT_DATE;

DECLARE v1, v2, v3 TINYINT;

<u>Робота зі змінними</u>

Після оголошення змінної їй можна задавати значення за допомогою команд SET або SELECT. Наприклад,

SET str = 'I am a string ';

SELECT CONCAT(str,paramstr), today FROM table2

<u>Структури керування потоками</u>

Mysql підтримує конструкції IF, CASE, ITERATE, LEAVE LOOP, WHILE і REPEAT для керування потоками в межах збереженої процедури (докладніше в [7]).

5.3.2 Курсори

Курсор – це посилання на контекстну область пам'яті. У деяких реалізаціях мови програмування SQL (Oracle, Microsoft SQL) при виконанні запиту отримується результуючий набір та зв’язаний із ним покажчик поточного запису. Курсор – це віртуальна таблиця, яка являє собою альтернативне сховище даних. Але курсор дозволяє звертатися до власних даних, як до даних звичайного масиву. Використовуються курсори у збережених процедурах.

Курсор не може використовуватися у MySQL сам по собі. Він є важливим компонентом збережених процедур.

За допомогою курсору є можливість перебрати набір даних та обробити кожен запис у відповідності з певними завданнями.

Оскільки курсор реалізується у збереженій процедурі, то він має усі переваги та недоліки властиві збереженим процедурам (контроль доступу, прекомпіляція, труднощі налагодження та інше).

Офіційну документацію щодо курсорів можна знайти у [8].

Курсори повинні бути оголошені до їх використання. Змінні з умовами оголошуються поперед курсорів. Обробники оголошуються строго після оголошення курсорів.

Mysql підтримує курсори в збережених процедурах. Далі наведено короткий синтаксис створення та використання курсору.

1 ***DECLARE cursor-name CURSOR FOR SELECT ...;***
/*Оголошення курсору та його заповнення */

2 ***DECLARE CONTINUE HANDLER FOR NOT FOUND*** /*Що робити, коли більше немає записів*/

3 ***OPEN cursor-name***; /*Відкрити курсор*/

4 ***FETCH cursor-name INTO variable [, variable***]; /*Встановлення покажчика на наступний рядок (якщо рядок існує). При цьому, значення полів поточного запису присвоюються змінним, які вказані оператором ***FETCH***. Якщо більше рядків не доступно, то відбувається зміна значення змінної ***SQLSTATE*** на **02000** (***SQLSTATE=00000*** свідчить про успішне виконання оператора). Для виловлення цієї події необхідно встановити обробник:

HANDLER FOR SQLSTATE '02000'

*/

5 ***CLOSE cursor-name***; /*Закрити курсор*/

Далі наведемо деякі прості операції з використанням курсору:

```
01 CREATE PROCEDURE `proc_CURSOR` (OUT param1 INT)
02 BEGIN
03    DECLARE a, b, c INT;
04    DECLARE cur1 CURSOR FOR SELECT col1 FROM table1;
05    DECLARE CONTINUE HANDLER FOR NOT FOUND SET b=1;
06    OPEN cur1;
07    SET b = 0;
08    SET c = 0;
13    WHILE b = 0 DO
14       FETCH cur1 INTO a;
15       IF b = 0 THEN
16          SET c = c + a;
17    END IF;
18    END WHILE;
```

```
19    CLOSE cur1;
20    SET param1 = c;
21 END //
```

Курсори мають три властивості, які необхідно зрозуміти, щоб уникнути одержання несподіваних результатів:

– ***Нечутливість***: курсор, що відкрився одного разу, не буде відображати зміни в таблиці, що сталися пізніше. Дійсно, Mysql не гарантує те, що курсор обновиться, так що не сподівайтеся на це;

– ***Доступність тільки для читання***: курсори не можна змінювати;

– ***Без перемотування***: курсор здатний проходити тільки в одному напрямку – уперед, крім того, неможливо пропускати рядки, не вибираючи їх.

Контрольні питання:

- Призначення збережених процедур.
- Характеристики збереженої процедури.
- Виклик збереженої процедури.
- Що таке змінні, робота з ними?
- Структури керування потоками в Mysql,
- Функції курсорів.
- Як створити курсор?
- Властивості курсорів.

ЛІТЕРАТУРА

1. Корнієнко С.К. Проектування інформаційного забезпечення автоматизованих систем: Навч. Посібник / С.К Корнієнко . – Запоріжжя: ЗНТУ, 2015. – 224 с
2. Кригель П. SQL. Библия пользователя / П. Кригель. – К.: Диалектика/Вильямс, 2010. – 752 с.

Додаток А
Інсталяція пакету Денвер

А.1 Компоненти програмного модуля Денвер

Для початку необхідно безкоштовно отримати інсталяційний пакет "Денвер". Останню версію інсталянту під ОС Windows можна за адресою: http://www.denwer.ru/dis/Base

Інсталянт містить:
– Apache, SSI, mod_rewrite, mod_php;
– PHP з підтримкою GD і MySQL;
– MySQL з підтримкою транзакцій (mysqld-max);
– система керування віртуальними хостами. Щоб створити новий хост, ви повинні додати директорію в каталог /home. При цьому змінювати файли конфігурації не потрібно. Стандартно підтримуються схеми назв директорій багатьох популярних хостерів;
– система керування запуском та завершенням;
– phpMyAdmin – система управління MySQL через Web-інтерфейс;
– ядро Perl без стандартних бібліотек (є можливість інсталювати іншого разу);
– емулятор sendmail. Листи із сайту, складаються у директорію tmp (зручно при тестуванні онлайн-голосувань);
– інсталятор.

А.2 Первинний тест мережі

Необхідно виконати простий тест, чи комп’ютер насправді налаштований щодо роботи у мережі. Виконується командою:

ping 127.0.0.1

Якщо відклик від хоста 127.0.0.1 існує, то внутрішній мережевий інтерфейс має доступ із зовні, то перейдемо до А.3.

Якщо відклику від хоста 127.0.0.1 немає, то користувачам Windows 98/ME потрібно зайти у ***Панель управления–>Установка и***

удаление программ–>Установка Windows та додати ***Удаленный доступ к сети***. Користувачам Windows 95 треба зайти у ***Панель управления–>Сеть*** і додати ***Протокол TCP/IP(від Microsoft)***. Після цього необхідно у тому ж вікні обрати із списку ***Способ входа в сеть*** значення "***Обычный вход в Windows***".

А.3 Інсталяція пакету

1) Перейти до акаунта з адміністративними правами.

2) Запустити інсталяційний файл *.exe.

3) Обрати, куди встановлюється Денвер, якщо на Flash-носій, то вводимо букву диска.

4) Надалі, чи встановлюється Денвер у директорію **WebServers** (y/n)?

5) Інформація щодо директорії встановлення інсталяційного пакету та опис віртуального диску (рис. А.1)

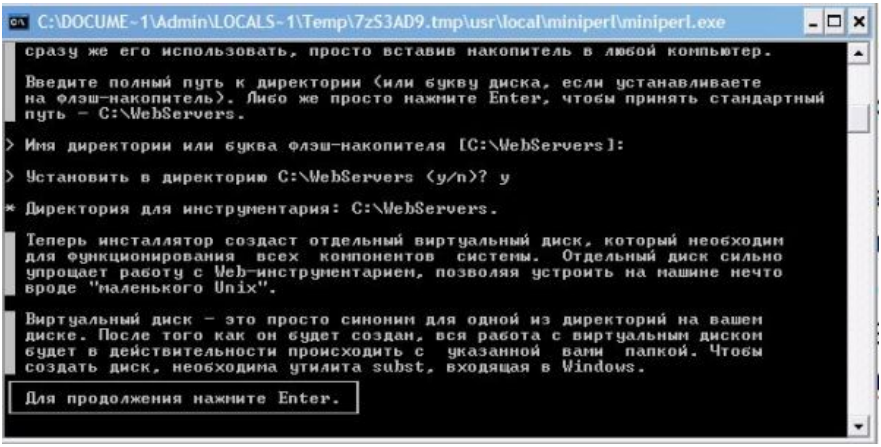

Рисунок А.1 – Віртуальний диск

6) Обирання букви віртуального диску. Якщо залишити диск з ім’ям **Z**, то «ENTER» (рис. А.2)

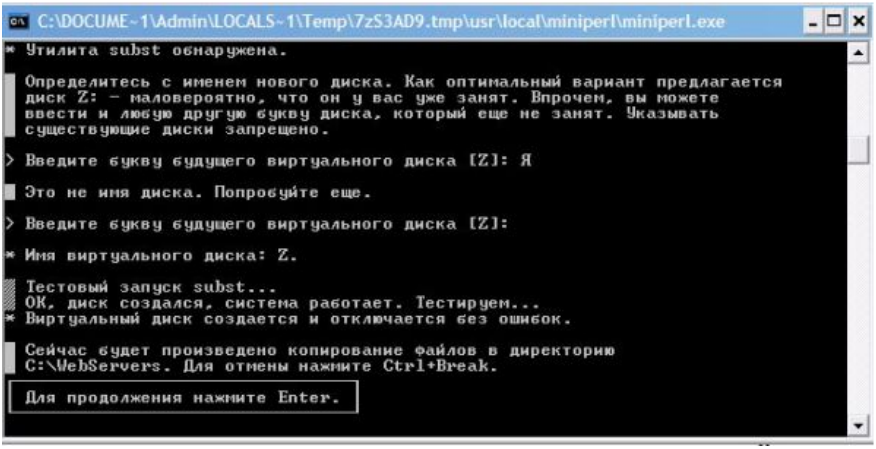

Рисунок А.2 – Вибір віртуального диску

7) Вибір режиму роботи віртуального сервера «**1**» (рис. А.3)

8) Для створення на робочому столі ярликів обираємо «**y**» (рис. А.3.5)

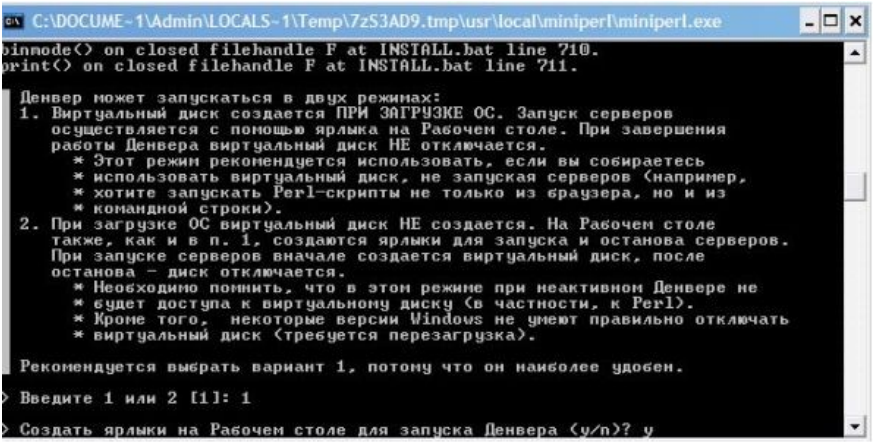

Рисунок А.3 – Вибір режиму роботи віртуального сервера

9) На робочому столі з’являються 3 ярлики: «**Start Denwer**», «**Restart Denwer**», «**Stop Denwer**».

А.4 Налаштування пакету

1) Ярлики повинні створитись не тільки під акаунтом адміністратора, а також і під акаунтом користувача. Якщо потрібно, то ярлики необхідно скопіювати у профільну директорію користувача. Але пакетом «Денвер» можливо керувати не створюючи ярликів, тоді необхідно запускати програми керування із зазначеної для «Денверу» директорії. Наприклад, пакет знаходиться у **C:\LANG\WebServers**, а файли для запуску, зупинки та інші у піддиректорії **\denwer**.

2) Якщо робота у пакеті «Денвер» планується під акаунтом користувача який не має достатніх привілеїв на зміну файлів та на доступ до директорій, то необхідно обов'язково надати ці привілеї (зміна даних) на директорію (та файли), у яку виконувалась інсталяція пакету.

Рекомендовано змінити привілеї для системного файлу хостів:

WINDOWS\system32\drivers\etc\hosts

Цьому файлу необхідно надати привілеї *зміни даних* для потрібного користувача, під яким планується робота з пакетом, а також додати як виключення для антивірусної програми, так як цей файл буде змінюватись у процесі роботи пакету «Денвер» при створенні та видаленні віртуальних хостів.

3) Для програми Skype **не використовувати** як альтернативні порти **80** та **443**.

Порти 80 (8080) використовуються пакетом для стандартного доступу за http-протоколом, а порт 443 застосовується для SSL.

4) Якщо порт Web-серверів 8080 зайнятий іншою програмою, то необхідно переназначити роботу пакета «Денвер» на інший порт, наприклад на 8088, як це вже зроблено у комп'ютерному класі. Для цього необхідно підкорегувати конфігурацію ***apache***.

Для переназначення основного порту **з 8080 на 8088**, необхідно у файлі **WebServers\usr\local\apache\conf\httpd.conf** додати строку **Listen 8089** після строки **#Listen 80**, та замінити порти **8080** на **8088** у шаблоні віртуального хоста у кінці цього ж файла.