Курсова робота

по курсу "ОРГАНІЗАЦІЯ БАЗ ДАНИХ"

Тема "Інформаційна система "Бібліотека університету"

Зміст

[Вступ](#_Toc280482387)

[1. Постановка задачі](#_Toc280482388)

[2. Проектування бази даних](#_Toc280482389)

[2.1 Відповідно до вимог поставленими в п.1 визначимо набір суті](#_Toc280482390)

[2.2.Визначимо типи зв’язків між сутностями](#_Toc280482391)

[2.3 Визначимо атрибути сутностей та їх типи](#_Toc280482392)

[2.4. Вигляд інформаційної моделі](#_Toc280482393)

[2.5 За даною інформаційною моделлю створюємо запити на створення таблиць](#_Toc280482394)

[3. Програмна реалізація](#_Toc280482395)

[3.1 З’єднання з базою даних](#_Toc280482396)

[3.2 Обзор основных методов](#_Toc280482397)

[4. Інструкція користувача](#_Toc280482398)

[5. Контрольний приклад](#_Toc280482399)

[Висновки](#_Toc280482400)

[Список літератури](#_Toc280482401)

7

# Вступ

Дана курсова робота має перед собою мету створення бази даних для бібліотеки університету. Кінцевий результат має бути створеною базою даних і програмною оболонкою, що полегшує використання цієї бази даних. Для реалізації поставленого завдання мною була вибрана СУБД POSTGRESQL, а для створення оболонки - мова програмування PHP.

На сьогоднішній день випущена версія POSTGRESQL v8 (19 січня 2005 року), яка є значною подією в світі баз даних, оскільки кількість нових можливостей доданих в цій версії, дозволяє говорити про виникнення інтересу крупного бізнесу як у використанні, так і його просуванні. Так, найбільша компанія в світі, Fujitsu підтримала роботи над версією 8, випустила комерційний модуль Extended Storage Management <http://www.fastware.com/postgresql.html>. Ліберальна BSD-лицензия дозволяє комерційним компаніям випускати свої версії POSTGRESQL під своїм ім'ям і здійснювати комерційну підтримку. Наприклад, компанія Pervasive оголосила про випуск Pervasive Postgres <http://www.pervasivepostgres.com/>.

POSTGRESQL підтримується на всіх сучасних системах (34 платформи) Unix, включаючи найбільш поширені, такі як Linux, FREEBSD, NETBSD, OPENBSD, SUNOS, Solaris, DUX, а також під Mac OS X. Зачинаючи з версії 8.X POSTGRESQL працює в "native" режимі під MS Windows NT, Win2000, WINXP, Win2003. Відомо, що є успішні спроби працювати з POSTGRESQL під Novell Netware 6 і OS2.

Основні можливості і функціональність POSTGRESQL.

Надійність POSTGRESQL є перевіреним і доведеним фактом і забезпечується наступними можливостями:

Продуктивність POSTGRESQL грунтується на використанні індексів, інтелектуальному планувальнику запитів, тонкої системи блокувань, системі управління буферами пам'яті і кешування, чудовій масштабованості при конкурентній роботі.

Розширюваність POSTGRESQL означає, що користувач може набудовувати систему шляхом визначення нових функцій, агрегатів, типів, мов, індексів і операторів.

Підтримка SQL, окрім основних можливостей, властивих будь-який SQL базі даних, POSTGRESQL підтримує:

Багатий набір типів даних POSTGRESQL включає:

Символьні типи (character (n)) як визначено в стандарті SQL і тип text з практично необмеженою довжиною.

Numeric тип підтримує довільну точність, дуже затребувану в наукових і фінансових застосуваннях.

Масиви згідно стандарту SQL: 2003

Великі об'єкти (Large Objects) дозволяють зберігати в базі даних бінарні дані розміром до 2Gb

Геометричні типи (point, line, circle,polygon, box.) дозволяють працювати з просторовими даними на плоскості.

ГИС (GIS) типи в POSTGRESQL є доказом розширюваності POSTGRESQL і дозволяють ефективно працювати з тривимірними даними. Подробиці можна знайти на сайті проекту PostGis <http://postgis.refractions.net/>.

Мережевих типів (Network types) підтримують типи даних inet для IPV4, IPV6, а також cidr (Classless Internet Domain Routing) блоки і macaddr

Композитних типів (composite types) об'єднують один або декілька елементарних типів і дозволяють користувачам маніпулювати з складними об'єктами.

Тимчасові типи (timestamp, interval, date, time) реалізовані з дуже великою точністю

POSTGRESQL має дуже багатий набір вбудованих функцій і операторів для роботи з даними, повний список яких можна глянути в документації <http://www.postgresql.org/docs/current/static/functions.html>.

Підтримка 25 різних наборів символів (charsets), включаючи ASCII, LATIN, WIN, KOI8 і UNICODE.

Підтримка NLS (Native Language Support)

Інтерфейси в POSTGRESQL реалізовані для доступу до бази даних з лави мов (C,C++,C#,python,perl,ruby,php,Lisp та інші) і методів доступу до даних (JDBC, ODBC).

Процедурні мови дозволяють користувачам розробляти свої функції на стороні сервера, тим самим переносити логіку додатка на сторону бази даних, використовуючи мови програмування, відмінні від вбудованих SQL і C

Простота використання завжди була важливим чинником для розробників.

phpPgAdmin <http://phppgadmin.sourceforge.net/> (ліцензія GPL) представляє можливість за допомогою веб-сервера браузера адмініструвати POSTGRESQL кластер.

pgAdmin III <http://www.pgadmin.org/> (GNU Artistic license) надає зручний інтерфейс для роботи з базами даних POSTGRESQL і працює під Linux, FREEBSD і Windows 2000/XP.

PgEdit <http://www.pgedit.com/> - програмна середа для розробки додатків і SQL-редактор, доступна для Windows і Mac.

Безпека даних також є найважливішим аспектом будь-якої СУБД. У POSTGRESQL вона забезпечується 4-мя рівнями безпеки:

POSTGRESQL не можна запустити під привілейованим користувачем - системний контекст

SSL,SSH шифрування трафіку між клієнтом і сервером - мережевий контекст

Складна система аутентифікації на рівні хоста або IP адреси/підмережі.

Деталізована система прав доступу до всіх об'єктів бази даних, яка спільно з схемою, що забезпечує ізоляцію назв об'єктів для кожного користувача, POSTGRESQL надає багату і гнучку інфраструктуру.

# 1. Постановка задачі

За завданням необхідно скласти базу для університетської бібліотеки, яка повинна забезпечувати збереження всіх необхідних даних. Створення інформаційної системи дозволить автоматизувати процес автоматичної роботи бібліотеки, збереження інформації про студентів, викладачів, книги, час видачі екземплярів. Це істотно заощадить час бібліотечних працівників.

Вся інформація буде систематизована і представлена в зручному і звичному для користувача вигляді.

# 2. Проектування бази даних

# 2.1 Відповідно до вимог поставленими в п.1 визначимо набір суті

|  |  |
| --- | --- |
| Об’єкт | Опис об’єкту |
| Книги | Інформація о всіх книгах на складі |
| Студенти | Студенти в бібліотеці |
| Вчителя | Вчителя університету |
| Мови | Інформація про мови, на якій написана книга |
| Видачі | Інформація о выдачі и возвраті книг |
| Статистика | Інформація о кількості книг |

# 2.2.Визначимо типи зв’язків між сутностями

|  |  |  |
| --- | --- | --- |
| Сутність 1 | Ступінь зв’язку | Сутність 2 |
| Книги | N: 1 | Мови |
| Студенти | N: 1 | Видачі |
| Вчителя | N: 1 | Видачі |
| Статистика | 1: 1 | Книги |
| Книги | N: 1 | Видачі |

# 2.3 Визначимо атрибути сутностей та їх типи

|  |  |  |
| --- | --- | --- |
| Сущність | Атрибут | Тип даних |
| Knigi | number | Ключове поле, цілий |
| nazva | Символьний |
| avtor | Символьний |
| izdatel | Символьний |
| otdel | Символьний |
| kol | Цілий |
| kost | Дробний |
| god | Целый |
| id\_lang | Цілий |
| Students | number\_s | Ключевое поле, Символьний |
| fio | Символьний |
| address | Символьний |
| telephone | Символьний |
| grupa | Символьний |
| Teachers | number\_t | Символьний |
| fio | Символьний |
| address | Символьний |
| Vzat\_knigi | Id\_data\_vidachi | Ключевое поле, Цілий |
| data\_vidachi | Символьний |
| Data\_vozvrata | Символьнийв |
| number | Цілий |
| number\_a | Символьний |
| Languages | Id\_lang | Ключевое поле, Цілий |
| lang | Символьний |
| Сущність  Stat | Атрибут | Тип даних |
| id | Ключове поле, Цілий |
| nazva | Символьний |
| kol | Цілий |
| kil | Цілий |
| ost | Цілий |

# 2.4. Вигляд інформаційної моделі

Students

number\_s

fio

address

telephone

grupa

Knigi

number

nazva

avtor

izdatel

otdel

kol

kost

god

id\_lang

Teachers

number\_t

fio

address

Languages

Id\_lang

languages

Vzat\_knigi

Id\_data\_vidachi

data\_vidachi

Data\_vozvrata

number

number\_a

Stat

id

nazva

kol

kil

ost

# 2.5 За даною інформаційною моделлю створюємо запити на створення таблиць

Створюємо таблиці (Knigi, Students, Teachers, Vzat\_knigi, Languages, Stat) и сразу делаем связи между ними:

CREATE TABLE languages

(

id\_lang int PRIMARY KEY NOT NULL,

lang char (30) NOT NULL

);

CREATE TABLE knigi

(

number int PRIMARY KEY NOT NULL,

nazva char (200) NOT NULL,

avtor char (100),

izdatel char (100),

otdel char (50) NOT NULL,

kol int,

kost float,

god int,

id\_lang int NOT NULL,

FOREIGN KEY (id\_lang) REFERENCES languages (id\_lang)

);

CREATE TABLE vzat\_knigi

(

id\_data\_vidachi int PRIMARY KEY NOT NULL,

data\_vidachi date NOT NULL,

data\_vozvrata date,

number int NOT NULL,

number\_a char (10) NOT NULL,

FOREIGN KEY (number) REFERENCES knigi (number)

);

CREATE TABLE students

(

fio char (200) NOT NULL,

number\_s char (10) PRIMARY KEY NOT NULL CHECK (number\_s like 's-%'),

address char (200) NOT NULL,

telephone char (15),

grupa varchar (10) NOT NULL

);

CREATE TABLE teachers

(

fio char (200) NOT NULL,

number\_t char (10) PRIMARY KEY NOT NULL CHECK (number\_t like 't-%'),

address char (200) NOT NULL

);

CREATE TABLE stat

(id int PRIMARY KEY NOT NULL,

nazva char (200),

kol int,

kil int,

ost int

);

# 3. Програмна реалізація

# 3.1 З’єднання з базою даних

Для з’єднання з базою використовуються методи в PHP. Це робить наступна строка:

$connect\_str="host=localhost user=postgres password=RESKATOR dbname=biblioteka";

где host=localhost - локалхост, до котрого приєднуються,

user=postgres password=RESKATOR - им’я та пароль користувача,

dbname=biblioteka - назва бази даних.

Щоб приєднати iншi вкладенi в папку файли з базою прописується строка:

$db=pg\_connect ($connect\_str);

# 3.2 Обзор основных методов

Для нормальної роботи бази даних були використані наступні запити:

1) Виведення двох таблиць (книги і мови) в одній

$query="Select knigi.number,knigi.nazva,knigi.avtor,

knigi.izdatel,knigi.otdel,knigi.kol,knigi.kost,knigi.god,

languages.lang from knigi inner join languages on languages.id\_lang=knigi.id\_lang

where knigi.number like '%".$\_POST ['number']."%' and

knigi.nazva like '%".$\_POST ['nazva']."%' and

knigi.avtor like '%".$\_POST ['avtor']."%' and

knigi.izdatel like '%".$\_POST ['izdatel']."%' and

knigi.otdel like '%".$\_POST ['otdel']."%' and

knigi.kol like '%".$\_POST ['kol']."%' and

knigi.kost like '%".$\_POST ['kost']."%' and

knigi.god like '%".$\_POST ['god']."%' and

knigi.id\_lang in (select id\_lang from languages where lang like '%".$\_POST ['id\_lang']."%'); ";

2) Видалення інформації з таблиці

$query='DELETE from vzat\_knigi where number='.$\_POST ['number'].'; ';

$res=pg\_query ($db,$query);

$query='DELETE from knigi where number='.$\_POST ['number'].'; ';

$res=pg\_query ($db,$query);

$query="DELETE from stat where id='".$\_POST ['number']."'; ";

$res=pg\_query ($db,$query);

3) Редагування даних в таблиці

"UPDATE knigi SET

nazva='".$\_POST ['nazva']."',

avtor='".$\_POST ['avtor']."',

izdatel='".$\_POST ['izdatel']."',

otdel='".$\_POST ['otdel']."',

kol='".$\_POST ['kol']."',

kost='".$\_POST ['kost']."',

god='".$\_POST ['god']."',

id\_lang='".$\_POST ['language']."'

where number='".$\_POST ['number']."'; ";

4) Додавання даних в таблицю

"INSERT into knigi values

('".$\_POST ['number']."',

'".$\_POST ['nazva']."',

'".$\_POST ['avtor']."',

'".$\_POST ['izdatel']."',

'".$\_POST ['otdel']."',

'".$\_POST ['kol']."',

'".$\_POST ['kost']."',

'".$\_POST ['god']."',

'".$\_POST ['id\_lang']."'

)";

# 4. Інструкція користувача

Запустив програму, бачимо наступне вікно:

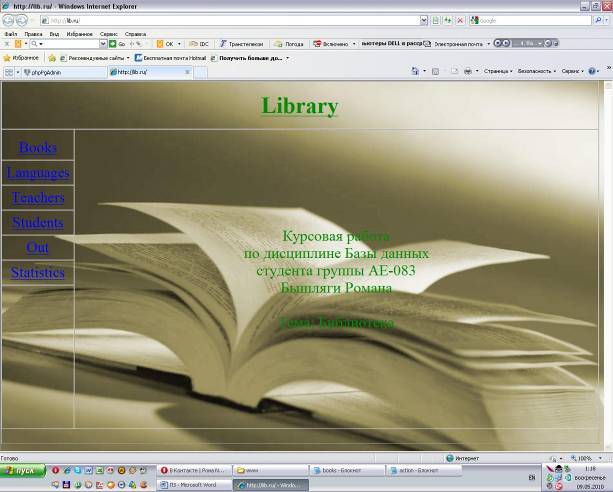


Рис.1 Стартове вікно

Для підключення до бази необхідно запустити сервер Denwer, в теці www знайти файл config.php і прописати правильне ім'я користувача і пароль для з'єднання з POSTGRESQL. У адресній стоці браузера ввести www.lib.ru. Далі, натискуючи на будь-яке заслання в меню зліва, ми отримуємо відповідні її назві запити. Наприклад, можна проглянути таблицю студенти:

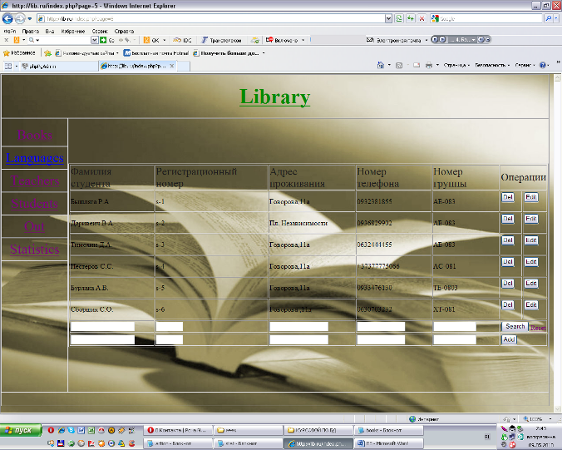


Рис.2 Таблиця Студенти

В кінці кожної таблиці додані кнопки для роботи з таблицею. Її можна редагувати, видаляти рядки або додавати нові. Також реалізована функція пошуку в таблиці:

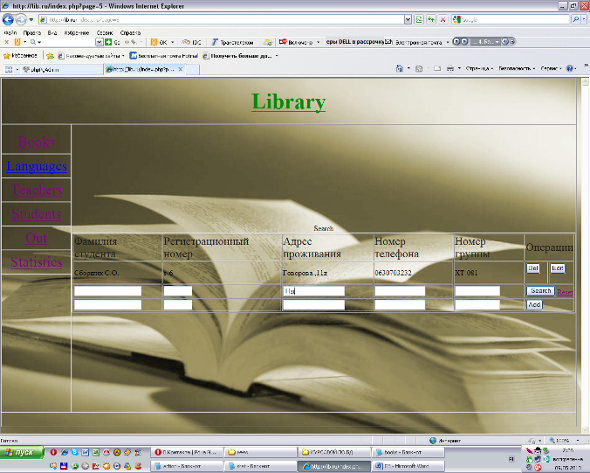


Рис.3 Пошук в таблиці студенти

# 5. Контрольний приклад

При натисненні кнопки "Edit", ми можемо редагувати дані в заданому полі. Підзапит дозволяє також змінити дані в інших зв'язаних таблицях:

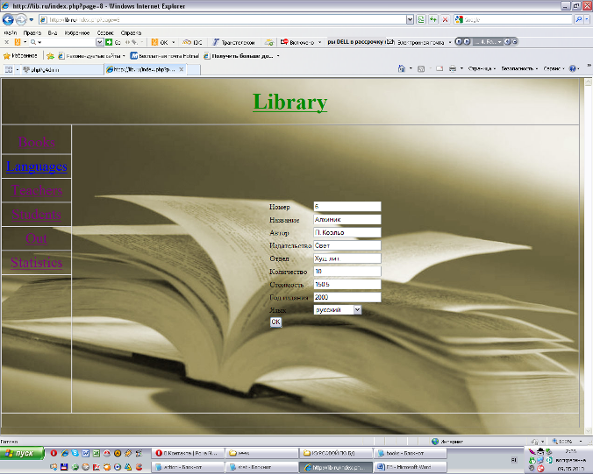


Рис.5 Контрольный пример

Після порівняння отриманих даних з таким же під запитом, виконаним в програмі pgAdminIII, робимо висновок, що програма працює правильно, оскільки дані збігаються.

# Висновки

Виконана робота дозволила набути навичок розробки баз даних від поставленого завдання до кінцевого програмного продукту, придатного до використання. В ході роботи розгледіли і освоєні навики використання мови SQL для роботи з базами даних, отриманий досвід організації баз даних в середі СУБД POSTGRESQL, зокрема використання запитів. Також освоєні прийоми звернення до баз за допомогою сервера Denwer, придбаний досвід програмування на мові PHP.

# Список літератури

1. Малахов Є.В. "Основи проектування баз даних. Конспект лекцій"