**13. Змінні середовища Web – сервера. Загальна характеристика.**

При запуску CGI-скрипта веб-сервер передає йому цілий ряд параметрів у змінні середовища. Частина змінних середовища генерується веб-сервером, а інша частина формується з полів HTTP-запиту.

В "Специфікації CGI/1.1" описано стандартний набір змінних середовища CGI, що формуються веб-сервером, а також спосіб передачі CGI-скрипту полів HTTP-запиту в змінних середовища. Крім цього, ряд веб-серверів (включаючи Apache) окрім стандартних формують свої змінні, в котрих передають скрипту додаткові ("нестандартні") параметри (E-Mail адміністратора сервера и т.п.).

**Змінні середовища CGI на конкретному хостингу.**

Щоб побачити всі доступні CGI-скрипту змінні середовища на конкретному сервері, можна запустити на ньому CGI-скрипт, що виведе імена і значення всіх своїх змінних середовища:

**#!/usr/bin/perl**

**print "Content-Type: text/html\n\n";**

**print "<HTML>\n<HEAD></HEAD><BODY>**

**<TABLE width=100% border=1 bordercolor=#00007F cellspacing=0>**

**<TR><TD align=center>Змінна</TD><TD align=center>Значення</TD></TR>\n";**

**foreach $itm(keys %ENV)**

**{print "<TR><TD>$itm</TD><TD>$ENV{$itm}</TD></TR>\n";}**

**print "</TABLE></BODY></HTML>\n";**

Залежно від умов виклику (HTTP-метод, прямий або через SSI...) набір змінних може бути різним.

**Змінні середовища CGI, що формуються веб-сервером.**

**QUERY\_STRING** - рядок параметрів виклику (всі символи, записані в URL після знаку '?').

**REQUEST\_METHOD** - метод HTTP, за допомогою котрого викликаний скрипт. Частіше за все цей метод GET або POST, хоча можуть бути й інші (PUT, DELETE и т.п.).

**GATEWAY\_INTERFACE** - версія інтерфейсу CGI у вигляді CGI/x.y. Наприклад, CGI/1.1

**REMOTE\_ADDR** - містить IP-адресу комп’ютера, з котрого було здійснено звернення до веб-сервера (адреса клієнта або останнього проксі-сервера).

**REMOTE\_PORT** - TCP-порт віддаленого комп’ютера, з котрого йде запит.

**REMOTE\_HOST** - доменне ім’я віддаленого комп’ютера, з котрого йде запит (визначається веб-сервером через DNS по значенню REMOTE\_ADDR, якщо це дозволено його конфігурацією).

**SERVER\_NAME** - доменне ім’я сервера.

**SERVER\_PORT** - номер TCP-порта веб-сервера.

**SERVER\_ADDR** - IP-адреса сервера.

**SERVER\_PROTOCOL** - версія HTTP-протоколу, що використовується для даного HTTP-запиту. Наприклад, HTTP/1.1.

**SERVER\_SOFTWARE** - програмне забезпечення сервера.

**SCRIPT\_NAME** - HTTP-шлях до скрипту.

**SCRIPT\_FILENAME** - фізичний повний шлях до скрипту в файловій системі сервера.

**PATH\_INFO** - HTTP-шлях до скрипту.

**PATH\_TRANSLATED** - повний фізичний шлях до скрипту.

Якщо HTTP-метод, що використовується для методу, передбачає передачу вмісту в тілі запиту, то передаються наступні змінні:

**CONTENT\_TYPE** - тип вмісту (MIME).

**CONTENT\_LENGTH** - довжина вмісту

Якщо стався аутентифікований запит (із вказуванням імені користувача і пароля), то передаються змінні:

**AUTH\_TYPE** - тип аутентифікації (аутентифікаційна схема, що використовується). Частіше за все - 'Basic'.

**REMOTE\_USER** – ім’я користувача, що пройшов аутентифікацію.

Якщо скрипт розрахований на обслуговування декількох користувачів, за параметром **REMOTE\_USER** він може їх розрізняти. Варто мати на увазі, що коли CGI-скрипт уже запущений і йому передано параметр REMOTE\_USER, то користувач ВЖЕ успішно пройшов аутентифікацію на рівні веб-сервера.

**14. Система управління базами даних (СУБД). Характеристика сучасних СУБД. Характеристика MySQL.**

   [База даних](http://lib.mdpu.org.ua/e-book/vstup/glos.htm" \l "%D0%91%D0%B0%D0%B7%D0%B0%20%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) (БД) - упорядкований набір логічно взаємопов'язаних даних, що використовується спільно, та призначений для задоволення інформаційних потреб користувачів. У технічному розумінні включно й система управління БД.

[Система управління базами даних](http://lib.mdpu.org.ua/e-book/vstup/glos.htm#%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0%20%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D1%96%D0%BD%D0%BD%D1%8F%20%D0%B1%D0%B0%D0%B7%D0%BE%D1%8E%20%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) (СУБД) - це комплекс програмних і мовних засобів, необхідних для створення баз даних, підтримання їх в актуальному стані та організації пошуку в них необхідної інформації.

     Централізований характер управління даними в базі даних передбачає необхідність існування деякої особи (групи осіб), на яку покладаються функції адміністрування даними, що зберігаються в базі.

     Головним завданням [БД](http://lib.mdpu.org.ua/e-book/vstup/glos.htm#%D0%91%D0%B0%D0%B7%D0%B0%20%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) є гарантоване збереження значних обсягів інформації та надання доступу до неї користувачеві або ж прикладній програмі. Таким чином БД складається з двох частин: збереженої інформації та системи управління нею. З метою забезпечення ефективності доступу записи даних організовують як множину фактів (елемент даних).

     Існує величезна кількість різновидів баз даних, що відрізняються за критеріями (наприклад, в Енциклопедії технологій баз даних [21] визначаються понад 50 видів БД).

     Відзначимо тільки основні класифікації.

     Класифікація БД *за моделлю даних*:

* ієрархічні,
* мережеві,
* реляційні,
* об'єктні,
* об'єктно-орієнтовані,
* об'єктно-реляційні.

Класифікація БД *за технологією фізичного зберігання*:

* БД у вторинній пам'яті (традиційні);
* БД в оперативній пам'яті (in-memory databases);
* БД у третинній пам'яті (tertiary databases).

Класифікація БД *за вмістом*:

* географічні.
* історичні.
* наукові.
* мультимедійні.

Класифікація БД *за ступенем розподіленості*:

* централізовані (зосереджені);
* розподілені.

     Окреме місце в теорії та практиці займають просторові (англ. spatial), тимчасові, або темпоральні (temporal) і просторово-часові (spatial-temporal) БД.

*Ієрархічні* бази даних можуть бути представлені як дерево, що складається з об'єктів різних рівнів. Верхній рівень займає один об'єкт, другий - об'єкти другого рівня і т.д.

     Між об'єктами існують зв'язки, кожен об'єкт може включати в себе декілька об'єктів більш низького рівня. Такі об'єкти перебувають у відношенні предка (об'єкт більш близький до кореня) до нащадка (об'єкт більш низького рівня), при цьому можлива ситуація, коли об'єкт-предок не має нащадків або має їх декілька, тоді як у об'єкта-нащадка обов'язково тільки один предок. Об'єкти, що мають загального предка, називаються близнюками.

*Мережеві* бази даних подібні до ієрархічних, за винятком того, що в них є покажчики в обох напрямках, які з'єднують споріднену інформацію.

     До основних понять мережевої моделі бази даних відносяться: рівень, елемент (вузол), зв'язок.

     Вузол - це сукупність атрибутів даних, що описують деякий об'єкт. На схемі ієрархічного дерева вузли представляються вершинами графа. У мережній структурі кожен елемент може бути пов'язаний з будь-яким іншим елементом.

     Незважаючи на те, що ця модель вирішує деякі проблеми, пов'язані з ієрархічною моделлю, виконання простих запитів залишається досить складним процесом.

     Також, оскільки логіка процедури вибірки даних залежить від фізичної організації цих даних, то ця модель не є повністю незалежною від програми. Іншими словами, якщо необхідно змінити структуру даних, то потрібно змінити і додаток.

*Реляційна* модель орієнтована на організацію даних у вигляді двовимірних таблиць. Кожна реляційна таблиця являє собою двовимірний масив і має наступні властивості:

* кожен елемент таблиці - один елемент даних;
* всі осередки в стовпчику таблиці однорідні, тобто всі елементи в стовпчику мають однаковий тип (числовий, символьний тощо);
* кожен стовпчик має унікальне ім'я;
* однакові рядки в таблиці відсутні;
* порядок проходження рядків і стовпчиків може бути довільним.

*Об'єктна* СУБД ідеально підходить для інтерпретації складних даних, на відміну від реляційних СУБД, де додавання нового типу даних досягається ціною втрати продуктивності або за рахунок різкого збільшення термінів і вартості розробки додатків. Об'єктна база, на відміну від реляційної, не вимагає модифікації ядра при додаванні нового типу даних. Новий клас і його екземпляри просто надходять у зовнішні структури бази даних. Система управління ними залишається без змін.

*Об'єктно-орієнтована* база даних (ООБД) - база даних, в якій дані оформлені у вигляді моделей об'єктів, що включають прикладні програми, які управляються зовнішніми подіями. Результатом поєднання можливостей (особливостей) баз даних і можливостей об'єктно-орієнтованих мов програмування є об'єктно-орієнтовані системи управління базами даних (ООСУБД). ООСУБД дозволяють працювати з об'єктами баз даних також, як з об'єктами у програмуванні в об'єктно-орієнтованих мовах програмування. ООСУБД розширює мови програмування, прозоро вводячи довготривалі дані, управління паралелізмом, відновлення даних, асоційовані запити й інші можливості.

     Об'єктно-орієнтовані бази даних звичайно рекомендовані для тих випадків, коли потрібна високопродуктивна обробка даних, які мають складну структуру.

     Система, яка забезпечує об'єктну інфраструктуру і набір реляційних розширювачів, називається "*об'єктно-реляційною*".

     Об'єктно-реляційні системи поєднують переваги сучасних об'єктно-орієнтованих мов програмування з такими властивостями реляційних систем як множинні представлення даних і високорівневі непроцедурні мови запитів.

*За технологією обробки даних* бази даних поділяються на централізовані й розподілені.

*Централізована* база даних зберігається у пам'яті однієї обчислювальної системи. Якщо ця обчислювальна система є компонентом мережі ЕОМ, можливий розподілений доступ до такої бази. Такий спосіб використання баз даних часто застосовують у локальних мережах ПК.

*Розподілена* база даних складається з декількох, можливо пересічних або навіть дублюючих одна одну частин, які зберігаються в різних ЕОМ обчислювальної мережі. Робота з такою базою здійснюється за допомогою системи управління розподіленою базою даних (СУРБД).

*За способом доступу* до даних бази даних поділяються на бази даних з локальним доступом і бази даних з віддаленим (мережевим) доступом.

     Системи централізованих баз даних з мережевим доступом припускають різні архітектури подібних систем:

* файл-сервер;
* клієнт-сервер.

     Файл-сервер. [Архітектура систем](http://lib.mdpu.org.ua/e-book/vstup/glos.htm" \l "%D0%90%D1%80%D1%85%D1%96%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%B0%20%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B8) БД з мережевим доступом передбачає виділення однієї з машин мережі в якості центральної (сервер). На такій машині зберігається спільно використовувана централізована БД. Усі інші машини мережі виконують функції робочих станцій, за допомогою яких підтримується доступ користувальницької системи до централізованої бази даних. Файли бази даних відповідно до призначених для користувача запитів передаються на робочі станції, де в основному і проводиться обробка. При великій інтенсивності доступу до одних і тих же даних продуктивність інформаційної системи падає. Користувачі можуть створювати також на робочих станціях локальні БД, які використовуються ними монопольно.

     Клієнт-сервер. У цій концепції мається на увазі, що крім зберігання централізованої бази даних центральна машина (сервер бази даних) повинна забезпечувати виконання основного обсягу обробки даних. Запит на дані, який видається клієнтом (робочою станцією), породжує пошук і вилучення даних на сервері. Витягнуті дані (але не файли) транспортуються по мережі від сервера до клієнта. Специфікою архітектури клієнт-сервер є використання мови запитів SOL.

**MySQL** - вільна система управління базами даних. MySQL є власністю компанії Oracle Corporation, що отримала її разом з поглиненою Sun Microsystems, яка здійснює розробку і підтримку додатку. Розповсюджується під GNU General Public License і під власною комерційною ліцензією, на вибір. Крім цього розробники створюють функціональність на замовлення ліцензійних користувачів, саме завдяки такому замовленню майже в найраніших версіях з'явився механізм реплікації.

     Цю систему управління базами даних з відкритим кодом було створено як альтернатива комерційним системам. MySQL із самого початку була дуже схожою на mSQL, проте з часом вона все розширювалася і зараз MySQL - одна з найпоширеніших систем управління базами даних. Вона використовується, у першу чергу, для створення динамічних веб-сторінок, оскільки має чудову підтримку з боку різноманітних мов програмування.

     MySQL є рішенням для малих і середніх додатків. Зазвичай MySQL використовується як сервер, до якого звертаються локальні або віддалені клієнти, проте до дистрибутиву входить бібліотека внутрішнього сервера, що дозволяє включати MySQL до автономних програм. Вихідні коди сервера компілюються на багатьох платформах. Найповніше можливості сервера виявляються в UNІХ-системах, де є підтримка багатонитевості, що підвищує продуктивність системи в цілому.

     Гнучкість СУБД MySQL забезпечується підтримкою великої кількості типів таблиць: користувачі можуть вибрати як таблиці типу MyISAM, що підтримують повнотекстовий пошук, так і таблиці InnoDB, що підтримують транзакції на рівні окремих записів. Більш того, СУБД MySQL поставляється із спеціальним типом таблиць EXAMPLE, що демонструє принципи створення нових типів таблиць. Завдяки відкритій архітектурі й GPL-ліцензуванню, в СУБД MySQL постійно з'являються нові типи таблиць. MySQL характеризується великою швидкістю, стійкістю і простотою використання.

     Для некомерційного використання MySQL є безкоштовною. Можливості сервера MySQL:

* простота у встановленні та використанні;
* підтримується необмежена кількість користувачів, що одночасно працюють із БД;
* кількість рядків у таблицях може досягати 50 млн.;
* висока швидкість виконання команд;
* наявність простої та ефективної системи безпеки.

**15. PHP. Основи синтаксису мови PHP. Прості оператори. Робота з Web-документом.**