УДК 004.65

ОЛЬГА ЛЯЛИК, ВІКТОР МАНДЗЮК

NOSQL СИСТЕМ ЗБЕРЕЖЕННЯ ДАНИХ: ПОРІВНЯЛЬНИЙ АНАЛІЗ І ПЕРСПЕКТИВИ ЇХ ВИКОРИСТАННЯ В НАВЧАЛЬНИХ ПОРТАЛАХ

Обґрунтовано необхідність використання альтернативних засобів збереження даних як доповнення до традиційних реляційних баз даних. Здійснено порівняльний аналіз поширених нереляційних систем зберігання даних. Описано особливості розглянутих програмних засобів, схарактеризовано кожного з них та напрямки можливого застосування.

Ключові слова: *електронне навчання, база даних, NoSQL, продуктивність, моделі даних.*

ОЛЬГА ЛЯЛЫК, ВИКТОР МАНДЗЮК

NOSQL СИСТЕМ ХРАНЕНИЯ ДАННЫХ: СРАВНИТЕЛЬНЫЙ АНАЛИЗ И ПЕРСПЕКТИВЫ ИХ ИСПОЛЬЗОВАНИЯ В УЧЕБНЫХ ПОРТАЛАХ

Обоснована необходимость использования альтернативных средств хранения данных в дополнение к традиционным реляционнымм базам данных. Осуществлен сравнительный анализ рассмотренных не реляционных систем хранения данных. Описаны особенности рассмотренных программных средств. Дана характеристика каждому из них.

Ключевые слова: *електронное обучение, база данных, NoSQL, модели данных.*

OLGA LIALYK, VIKTOR MANDZYUK

NOSQL STORAGE SYSTEMS: COMPARATIVE ANALYSIS AND THE PROSPECTS FOR THEIR USAGE IN EDUCATIONAL PORTALS

The necessity of usage of alternative means storing data in addition to traditional sql-databases are grounded. The comparative analysis NoSQL data storage systems and features of the considered software is done. The characteristics of softwares and their possible applications are described.

Key words: *e-leaning, storage systems, NoSQL, performance, data model.*

Поняття «електронне навчання» поєднує у собі багато форм і методів використання сучасних інформаційних технологій. У країнах зарубіжжя домінують різні напрями розвитку дистанційної освіти. Для прикладу, в США значна увага приділяється використанню сучасних програмно-технічних і комунікаційних технологій. У Європі ж акцент ставиться на навчально-методичне забезпечення, яке дає можливість об'єднати якнайбільшу кількість слухачів і мінімізувати затрати на освіту [4].

Останнім часом у глобальній мережі Інтернет спостерігається значне зростання обсягу інформації, пов'язаної, зокрема, із проблемами науки й освіти. При цьому стандартні сервісні можливості мережі вже не дозволяють ефективно використати цю інформацію в практичних цілях. Це особливо помітно, коли виникає необхідність пошуку інформації. У зв'язку з цим з'явилися нові конструктивні пропозиції щодо організації роботи з Інтернет-ресурсами. Вся інформація групується в певні значеннєві розділи й далі її обробка й, відповідно, сервіс, що надається користувачам, організуються вже в межах виділеного інформаційного простору. Головне, що при цьому різко скорочується обсяг інформації, який безпосередньо перебуває в роботі й значно зростає ефективність пошукових механізмів. Реалізується такий підхід за допомогою

порталів, через які практично й здійснюється вихід на мережеві інформаційні ресурси певного профілю. Ідея порталу сьогодні визнається й підтримується більшістю фахівців [6], але до його детального значеннєвого визначення та форм конкретної реалізації відношення неоднозначне. Враховуючи це, було проведено аналіз літератури таких авторів, як Б. И. Покровський, В. Н. Васильєв, С. К. Стафеєва, Ю. Н. Сєміна та ін. Профільні освітні портали розглядаються як ефективний засіб удосконалення вітчизняної організаційно-навчальної структури. Вони покликані забезпечити на сучасному технологічному рівні ефективну інформаційну підтримку освітньої діяльності.

Метою статті є розгляд технології NoSQL збереження даних, яку використовують у сучасних інтернет-порталах і сайтах, для яких є характерним високе навантаження на сховище даних. На сучасному етапі розвитку електронного навчання, інтернет-портали, які його забезпечують, приймають на себе щоразу більше навантаження, що пропорційне як зростанню кількості користувачів, так і обсягам даних, які доводиться зберігати на такому ресурсі.

Проблемами розвитку сучасних СКБД займалися багато вітчизняних та зарубіжних дослідників, зокрема, В. Е. Фрейман, Н. В. Сазонова, Н. Р. Балик, Ю. С. Рамський, Г. Ю. Цибко, А. П. Маллан, К. Дейт, Дж. Хаббард, С. М. Діго, Р. Крамм, Дж. Ульман та інші, видано чимало методичних матеріалів, присвячених цим питанням.

Безпрецедентні обсяги даних змушують розробників придивлятися до альтернатив реляційних баз даних, які використовуються вже більше тридцяти років. Сукупність таких технологій відома як «NoSQL бази даних». Цей термін виник в IT-середовищі і на час написання статті не має чіткого сформульованого визначення. Вони переважно орієнтовані на збереження значних обсягів даних.

Виділяють такі загальні види систем керування базами даних:

Реляційні СУБД. Реляційна модель дає можливість сконцентрувати дані про певний об’єкт у вигляді комірок одного рядка таблиці та встановити зв’язки між рядками різних таблиць. Модель має серйозний недолік — заздалегідь жорстко фіксується і типізується набір властивостей об’єкта. У результаті цього сфера застосування такої моделі відносно вузька. Але на практиці реляційні СУБД використовуються повсюдно ціною значних зусиль і впровадження різних «надбудов» (до яких усі настільки звикли, що часто не усвідомлюють всієї складності універсального застосування цієї моделі).

Triplestore. Сховище фактів у вигляді триплетів (об’єкт, предикат, суб’єкт) — дуже гнучкий спосіб керованого зберігання даних. Факти утворюють семантичну мережу. За допомогою правил неважко автоматично вивести нові факти з наявних. Якщо в реляційних СУБД рядок таблиці містить у собі дані, то в семантичній мережі вузол не містить нічого крім ідентифікатора. Проте можна знайти факти, що посилаються на цей вузол, які і є його описом.

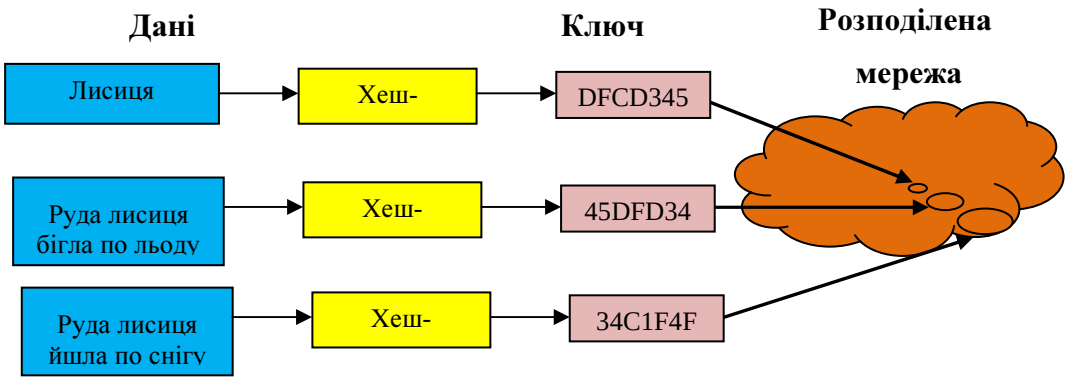
Document-oriented. Документно-орієнтовані бази даних — компромісна модель, що поєднує гнучкість триплетів з ємністю реляційних кортежів. Документ містить довільне число властивостей. Через концепцію документу можна виразити як стовпці таблиці, так і зв’язки вузла (щоправда, немає можливості спеціально позначати деякі властивості як логічно виведені). Пошук вузла за властивістю і вибірка всіх властивостей вузла можлива одним запитом, а не двома, як при роботі з триплетами. Відповідно, за необхідності отримати розширену інформацію про ланцюжок вузлів, документно-орієнтована модель виявляється вдвічі ефективнішою, ніж триплетна (якщо зв’язки в ланцюжку не транзитивні). Документно-орієнтована модель добре поєднується з ООП (об’єктно-орієнтовним програмуванням).

База даних, орієнтована на документи, зберігає не «відношення», а «документи» (тісно переплетені набори даних, які в цілому не пов’язані з іншими елементами даних в системі). Наприклад, записи в системі блогів абсолютно не пов’язані один з одним, і навіть коли в одному з них є посилання на інший, найчастіше вони з’єднані гіперпосиланням, який розбирається браузером, а не є якимось внутрішнім зв’язком. Коментарі до запису в блозі видимі виключно в рамках цього запису і невідомо чи прийде комусь в голову вибирати всі коментарі незалежно від того, до якого запису вони відносяться.

Більш того, бази даних, орієнтовані на документи, зазвичай ефективніші у високопродуктивних середовищах або середовищах з високим ступенем паралелізму; MongoDB, зокрема, націлена на високу продуктивність, а її близька родичка, CouchDB, — більшою мірою розрахо-

вана на середовища з високим ступенем паралелізму. В обох системах відмовилися від будь-якої підтримки транзакцій за участю декількох об'єктів, виходячи з того, що, хоча вони підтримують паралельну модифікацію одного об'єкта в базі даних, будь-яка спроба змінити більше одного об'єкта одночасно залишає невелике тимчасове вікно, коли ці модифікації можуть бути видимі «у процесі їх застосування». Документи оновлюються атомарно, але поняття транзакції, що охоплює поновлення кількох документів відразу, немає. Це не означає, що в MongoDB немає ніяких засобів забезпечення стійкості до збоїв, — просто MongoDB «не переживе», наприклад, випробування збоєм електропостачання, на відміну від екземпляра SQL Server. Системи, що вимагають повної семантики ACID (atomicity, consistency, isolation and durability), краще будувати на основі традиційних систем управління реляційними базами даних, тому швидше за все в найближчій перспективі ніхто не буде зберігати у MongoDB критично важливі дані, крім, можливо, вже реплікованих або тих, що кешуються на веб-сервері.

На низькому рівні такі бази даних будуються на базі хеш-таблиць та їх різновидів — розподіленої хеш-таблиці (DHT). Це просто звичайна, хоч і величезна таблиця, яка може автоматично розподілятися на будь-яку кількість комп'ютерів і підтримує пошук і отримання даних з баз знань, де вони конкретно перебувають. Хоча зазвичай для швидкої роботи дані зберігаються в оперативній пам'яті, але деякі сервери забезпечують зберігання на диску і резервне копіювання так, що після вимкнення такого сервера всі дані зберігаються.



Основною проблемою реляційних баз даних є те, що вони не можуть впоратися із навантаженнями актуальними в наш час. Чітко виділяються три проблемних галузі [5]:

- горизонтальне масштабування за великих обсягів даних, наприклад, як у випадку соціальних мереж (потрібно проаналізувати 50 терабайт текстових даних для пошуку за вхідними повідомленнями), або всесвітньо відомий аукціон eBay (інформація, якою оперує даний ресурс займає більше 2-ох петабайт [5]);
- продуктивність кожного окремого сервера (відомо, що практично будь-який достатньо високо завантажений веб-проект, в основі якого лежить реляційна база даних, розміщує її на кількох серверах для того, щоб зменшити навантаження);
- негнучкий дизайн щодо логічної структури даних (якщо на початковій стадії розробки проекту, при проектуванні структури реляційної бази даних не було враховано деяких моментів чи напряму розвитку проекту, то згодом часто доводиться «ламати» всю структуру бази даних).

Термін NoSQL був запропонований Еріком Евансом [1], коли Джоан Оскарсон [2] з Last.fm хотів організувати захід для обговорення розподілених баз даних з відкритим вихідним кодом.

Багато розробників не схвалюють термін NoSQL, бо він звучить, як заклик не використовувати SQL. NoSQL — це не підхід проти реляційних баз даних. NoSQL — можна інтерпретувати як «не тільки SQL» (Not Only SQL), а не «No SQL» (No SQL at all).

Під терміном NoSQL криється велика кількість продуктів з абсолютно різними підходами й іноді при обговоренні розмова може йти про різні системи. Тому ми пропонуємо використо-

вувати три критерії для порівняння цих систем: масштабованість, модель даних і запитів, система зберігання даних.

Було обрано 10 NoSQL баз даних для прикладів. Це не весь список, але. на нашу думку, їх достатньо для оцінки.

Neo4J пропонує повноцінну базу даних із транзакціями, індексами (Lucene), декількома режимами роботи і простотою вивчення, завдяки дуже простій структурі. Всі коди відкриті й доступні, а програма поставляється під подвійною ліцензією AGPL для некомерційного використання і тарифною сіткою для комерційного. Щоб розпочати використання даної СКБД, достатньо знати Java і розуміти, для чого потрібна мережева структура даних (граф).

Cassandra — це NoSQL сховище даних, яке народилося на Facebook й успішно використовується в таких великих веб проектах, як Digg, Twitter та інших. Cassandra заснована на Columnfamily (перший, хто використовував даний підхід в NoSQL сховище Google зі своїми BigTable) моделі. У даній СКБД є рядки і стовпці як і у класичних БД.

HBase є продуктом з відкритим вихідним кодом (open source), нереляційною, розподіленою базою даних (на зразок BigTable від Google), що написана на Java. Він розроблений у рамках проекту Hadoop Apache Software Foundation і працює поверх HDFS (розподілена файлова система Hadoop), забезпечуючи BigTable-подібні можливості для Hadoop. Тобто, він забезпечує стійкий до збоїв спосіб зберігання значної кількості даних.

CouchDB — документно-орієнтована система управління базами даних, що не вимагає опису схеми даних. Ця програма є вільною, відкритою, вона написана на мові Erlang.

MongoDB — документно-орієнтована система управління базами даних з відкритим вихідним кодом, яка не потребує опису схеми таблиць. Написана на мові C++. СУБД оперує наборами JSON-подібних документів, що зберігаються в двійковому вигляді в форматі BSON.

Redis — надає схожі до Memcached функції для зберігання даних у форматі ключ-значення. Активне кешування даних в оперативній пам'яті дозволяє досягти продуктивності: 110000 операцій запису або 81000 операцій читання в секунду на сервері з CPU Xeon X3320 2.5 ГГц. На відміну від Memcached, Redis забезпечує постійне зберігання даних на диску і гарантує збереження БД при аварійному завершенні роботи. Вихідні тексти проекту поширюються в рамках ліцензії BSD.

Riak — це розподілене, масштабоване, стійке до збоїв сховище для зберігання інформації. Це система з відкритим вихідним кодом і підтримкою звернень за допомогою HTTP, JSON і REST.

Scalaris — масштабоване сховище даних з підтримкою транзакцій, організоване за принципом ключ-значення. Scalaris використовує структуровані перекриття з протоколом Paxos для обробки транзакцій. Scalaris написаний на мові Erlang.

Tokyo Cabinet — це сучасне гнучке сховище даних, яке підтримує кілька типів БД, включаючи як найпростіші ключ-значення, так і «табличні» та документно-орієнтовані. Останній момент ставить його в один ряд з такими продуктами як CouchDB, MongoDB та ін., при цьому воно значно простіше і легше, а тому швидше. Крім того, Tokyo Cabinet підтримує досить складні запити, а також для нього можна писати розширення на Lua.

Voldemort — розподілене сховище даних, організоване за принципом ключ-значення, написане на Java

Масштабованість. Під масштабованістю деякі дослідники мають на увазі реплікацію, інші — автоматичний розподіл даних між декількома серверами. Такі системи називають розподіленими базами даних. До них належать Cassandra, HBase, Riak, Scalaris і Voldemort. Це практично єдиний вибір, якщо використовується обсяг даних, який не може бути оброблений на одній машині, або якщо не поставлена задача керувати розподілом вручну.

Є такі особливості, на які варто в першу чергу звертати увагу в розподіленій базі даних: підтримка кількох датацентрів і можливість додавання нових машин в працюючий кластер прозоро для програм.

	Прозоре додавання машини в кластер	Підтримка кількох датацентрів
Cassandra	+	+
HBase	+	
Riak	+	+
Scalaris	+	
Voldemort		Необхідно дописувати частину коду

Нерозподілені бази даних включають в себе CouchDB, MongoDB, Neo4j, Redis і Tokyo Cabinet. Ці системи можуть служити «прошарком» для зберігання даних для розподілених систем: MongoDB надає обмежену підтримку шардінга (sharding) так само. як і окремий проект Lounge для CouchDB; Tokyo Cabinet може використовуватися як система зберігання файлів для Voldemort.

Модель даних і запитів. Існує чимало різних моделей даних і API запитів в NoSQL базах даних.

	Модель даних	**API запитів**
Cassandra	Сімейства стовпців	Thrift
CouchDB	Документи	Map/Reduce
HBase	Сімейства стовпців	Thrift, REST
MongoDB	Документи	Cursor
Neo4j	Графи	Graph
Redis	Колекції	Collection
Riak	Документи	Nested Hashes
Scalaris	Ключ/значення	get/put
Tokyo Cabinet	Ключ/значення	get/put
Voldemort	Ключ/значення	get/put

Система сімейства стовпців (columnfamily) використовується в Cassandra і HBase. Ця ідея була запозичена з документів, що описують будову Google Bigtable (Cassandra трохи відійшла від ідей Bigtable і ввела supercolumns). В обох системах є рядки і стовпці, які усі звикли бачити, але кількість рядків не є великою: кожен рядок має більше або менше стовпців, залежно від потреби, також стовпці не повинні бути визначені заздалегідь.

Система ключ/значення є простою, не складною для реалізації, але не ефективною, якщо клієнти зацікавлені тільки в запиті або оновленні частини даних. Також важко реалізувати складні структури поверх розподілених систем.

Документо-орієнтовані бази даних — це наступний рівень систем ключ/значення, що дозволяють пов’язувати вкладені дані з кожним ключем. Підтримка таких запитів ефективніша, ніж просто повернення всього BLOB кожного разу.

Neo4J має справді унікальну модель даних, зберігаючи об’єкти і зв’язки як вузли та ребра графа. Для запитів, які відповідають цій моделі (наприклад, для ієрархічних даних), вони можуть бути в рази ефективніші, ніж альтернативні варіанти.

Scalaris — унікальна у використанні розподілених транзакцій між кількома ключами. Обговорення компромісів між послідовністю і наявністю вільних місць виходить за рамки цієї статті, але це інший аспект, який необхідно враховувати при оцінюванні в розподілених системах.

Система зберігання даних. Під системою зберігання даних розуміють спосіб зберігання даних всередині системи.

	Модель даних
Cassandra	MemTable/SSTable
CouchDB	Append only B-Tree
HBase	MemTable/SSTable on HDFS
MongoDB	B-Tree
Neo4j	On-disc linked listd
Redis	In memory with background snapshots
Riak	Hash
Scalaris	In-memory only
Tokyo Cabinet	Hash or B-Tree
Voldemort	Pluggable (primarily BDB MySql)

Бази даних, які зберігають дані в пам'яті, дуже швидкі (Redis може виконувати до 100,000 операцій в секунду), але не можуть працювати з даними, що перевищують розмір доступної оперативної пам'яті. Довговічність (збереження даних у разі збою на сервері або відключення живлення) теж може бути проблемою (у нових версіях буде підтримка append-only log). Кількість даних, які можуть чекати запису на диск, є достатньо значною. Інша система зі зберіганням даних в оперативній пам'яті — Scalaris, вирішує проблему довговічності за допомогою реплікації, але вона не підтримує масштабування на кілька датацентрів, так що втрата даних ймовірна й тут — у випадку відключення живлення.

Memtables і SSTables буферизують запити на запис у пам'яті (memtable), після запису в commit лог для збереження даних (докладніше можна про це почитати в wiki Cassandra — http://wiki.apache.org/cassandra/ArchitectureOverview). Після накопичення достатньої кількості записів, Memtable сортується і записується на диск, вже як SSTable. Це дає продуктивність близьку до продуктивності пам'яті, в той же час система позбавлена проблем актуальних при зберіганні тільки в пам'яті.

B-дерева (binary tree, двійкові дерева) використовуються в базах даних вже дуже давно, вони забезпечують надійну підтримку індексування, але продуктивність їх доволі низька при використанні на машинах з вінчестерами на магнітних дисках (які, як і колись, є економічно ефективніші). Це спричинено значною кількістю позиціонувань головки при записі або читанні даних.

Цікавим варіантом є використання в CouchDB B-дерев, тільки з функцією додавання (append-only B-Trees — бінарне дерево, яке не потрібно перебудовувати при додаванні елементів), що дає можливість отримати непогану продуктивність при записі даних на диск.

Популярність NoSQL різко зросла у 2009 році завдяки збільшенню кількості проектів, пов'язаних з використанням великих обсягів даних. З'являється все більше систем, які дозволяють організовувати і прозоро підтримувати величезні масиви даних, обробляти і контролювати ці дані. Хоча у статті акцент більш зроблено на технічні характеристики баз даних, а не на методиці їх вивчення чи способах використання при організації дистанційної освіти, вважаємо, результати дослідження будуть корисними при виборі технічних засобів організації дистанційного та комп'ютерно-орієнтованого навчання.

Вважаємо, що дана технологія має перспективи використання при розробці навчальних порталів, для яких характерним є наявність значної кількості даних, які розбиті на окремі незалежні блоки. Такими блоками виступають лекції, уроки, відео-фрагменти, графічні зображення тощо. Під час використання такого інтернет-ресурсу в його БД накопичується величезна кількість даних, пов'язана зі спілкування учасників навчального процесу між собою, а саме форуми, блоги, дошки оголошень, новини, історія чатів та особистих переписок. Усе це різноманіття текстових даних не є критично важливим у роботі порталу, але його потрібно десь зберігати та «вміти» оперативно здійснювати пошук.

Другим аспектом при аналізі можливостей застосування розглянутих БД у таких порталах є наявність значних об'ємів інформації у так званих базах знань. База знань — це сукупність відомостей (про реальні об'єкти, процеси, події або явища), що відносяться до певної теми або задачі, організована так, щоб забезпечити зручне представлення цієї сукупності як в цілому, так і будь-якої її частини. Це означає, що система управління базою знань (саме знань, а

не даних) повинна забезпечити уявлення і обробку моделі, зіставною за своєю складністю із моделлю, що використовується свідомістю людини. Основою для зберігання даних у такому сховищі може бути одна з розглянутих БД.

Окремо варто розглянути використання БД у соціальних мережах. Останні роки характеризуються бурхливим розвитком соціальних мереж, на які вже звернули увагу методисти, як на засіб що може сприяти навчальному процесу. Для такого роду сервісів є характерним високе навантаження на БД та висока відвідуваність ресурсу, що відповідно і спричиняє навантаження на СКБД. Дану проблему може вирішити використання NoSQL систем.

Описані результати дослідження показали, що розглянутий напрям в розвитку СКБД достатньо розвинутий і на ринку представлені різні програмні засоби з категорії «NoSQL». Розглянуті СКБД не є однотипними, хоча в більшості випадків забезпечують виконання одних і тих же завдань, однак дають різні засоби для цього. Варто також звернути увагу, що БД, які організовані за принципом key-value, як правило, дають об'єктно-орієнтований інтерфейс для програміста, а технологія ООП останнім часом вважається однією з найпрогресивніших при створенні програм, та підтримуються в усіх мовах високого рівня. На противагу цьому класична мова запитів SQL є достатньо специфічною, і багато дослідників навіть не відносять її до мов програмування як таких. Тому можна зробити висновок, що програмування БД, доступ до яких здійснюється за класичною технологією, вимагає від програміста постійного переналаштування процесу мислення з ООП на SQL, хоча для більшості професійних програмістів це відбувається непомітно, проте початківцям часто це дається важко.

Висновки та перспективи подальших досліджень. Вибір між технологіями (вище розглянутих СКБД чи програмного засобу, що використовує класичний SQL) залежить від багатьох факторів (постановки задачі, кваліфікації виконавця, можливостей апаратного забезпечення тощо), тому однозначної рекомендації, яку із розглянутих вище (чи інших) СКБД слід використовувати, дати неможливо. Проте зазначимо, що розглянуті програмні засоби використовують нову і достатньо прогресивну технологію, яка за багатьма показниками випереджає перевірених часом «гігантів» (MS SQL, MySQL, Oracle тощо), які вже десятки років займають передові позиції на ринку БД. Зважаючи на лавиноподібне зростання користувачів Інтернету взагалі та учасників e-learning зокрема, у зв'язку зі збільшенням навантаження на сховища даних, слід, можливо, переглянути класичні підходи до реалізації баз даних та звернути увагу на нові технології, які пропонує ІТ-спільнота.

ЛИТЕРАТУРА

1. Eric Evans's Weblog [Електронний ресурс]. — Режим доступу до блогу : http://blog.sym-link.com/.
2. Oskarsson Johan «braindump» веб-блог. — Режим доступу до блогу.: http://blog.oskarsson.nu/ .
3. William Rice E «Moodle E-learning Course Development» — Packt Publishing, 2006. — 256 с.
4. Малышев С. Л. Модели и организация новых образовательных технологий в системе дистанционного обучения педагогического вуза. Дис. канд. екон. наук / С. Л. Малышев. — М., 2000 – 134 с.
5. Обзор NoSQL систем [Електронний ресурс] — Режим доступу до ресурсу: http://habrahabr.ru/blogs/nosql/77909/
6. Разработка организационно-методических основ коллективного формирования информационных ресурсов профильных образовательных порталов / [Лунин В. В., Мельников М. Я., Миняйлов В. В., Покровский Б. И.] Сб. научн. ст. «Интернет-порталы: содержание и технологии». Вып. 1. ГНИИ ИТТ «Информика». — М.: Просвещение, 2003. — С. 635–666.
7. Технологія розробки дистанційного курсу : навчальний посібник / [Биков В. Ю., Кухаренко В. М., Сиротенко Н. Г. та ін.], за ред. В. Ю. Бикова, В. М. Кухаренка — К.: Міленіум, 2008. — 324 с.